mathe II

david vajda

9. November 2026

- der text
  - also:
  - das schreiben:
  - die geschichtlichen anmerkungen, in fussnoten in LaTeX
- next
  - theorie, modell, ...
    - theorie
    - modell
    - methode
    - verfahren
    - test
    - theorem
  - maschine, system, ...
    - maschine
    - system
    - automat
    - schaltwerk
  - backend, frontend, synthese, analyse, darstellung, ...
    - backend
    - frontend
    - synthese von schaltwerken
    - analyse von schaltwerken

        * darstellung des automaten

        * simulation des automaten

- methode, mathe, ...
  - methode der linearen kongruenz
  - test der zufaelligkeit (chi quadrat test)
- zwischenbemerkung
  - automat
    * darstellung des automaten
    * simulation des automaten
  - zufallszahl
    * methode der linearen kongruenz
    * test der zufaelligkeit (chi quadrat test)
- linear, additiv, ...
  - linear
  - additiv
  - einfach
  - zusammenhängend
  - mehrfach
  - adaptiv
  - linear
  - sequentiell
- pattern matching:
  - knuth morris pratt
  - boyer moore
  - rabin karp
  - cook ghosper
  - zufallszahlen:
    * d. lehmer
    * knuth
  - algorithmische mathematik, graphen:

        * douglas
        * west
        * kasimir kuratoswski

- zufall, random:
  - zufaellgkeit
  - zufallszahl
  - zufalsexperiment
  - zufällig, beliebig, ...
    * : Marx: einfach, einzeln, zufällig, (der gemeine Mensch)... allgemein, (der gemeinschauplatz oder etwas allgmein gueltiges ...total und erweitert (Deutschland)
    * zufallszahlen:
      · zufällig
      · beliebig
      · pseudo-zufällig
      · echt
    * Mathematik für berufliche gymnasien+:
      · Zufallszahl
      · Zufallsexperiment
      · Wahrscheinlichkeitsfunktion $P(X = x_i)$
      · Zufallsvariable $X(e_i)$
      · Erwartungswert $E(X)$

- Verarbeitung von Zeichenfolgen
  - knuth morris pratt:
  - suchen im muster

    ```
    p # muster
    a # text
    ```

    * p das musster (klein p)
    * a der text
  - text-zeichenfolge
  - binäre zeichenfolge

  - text-zeichen:
    * buchstabe
    * ziffer
    * sonderzeichen
  - Wonneberger (Rheinhard Dr.) - Würzburg, LaTeX, +Knuth verarbeitung von Zeichenfolgen, LaTeX
    * steuerzeichen

    - mathematische zeichen, rechenzeichen
    - akzent, umlaut
    - ...
  - Knuth Morris Pratt

```
p [0]
p [j]
p [M]


next [j]

j = next [j]
next [j] = j

## suchen in dem muster
## ...

while (j <= M)

## ...

while (p[j] != a [i])
while (p[j] != p [i])

## ...

while ((j <= M) && (p[j] != a [i]))
while ((j <= M) && (p[j] != p [i]))

## ...

j = next [j];

## ...

while ((j <= M) && (p[j] != a [i])) j = next [j];
while ((j <= M) && (p[j] != p [i])) j = next [j];

## ...

i++, j++;
```

- ein wenig zufallszahlen
  - suchen im eigenen muster, knuth morris pratt
  - uml, erm
  - methode:
  - methode der linearen kongruenz

```
a [i] = (a[i-1]*b+1) % m;
```

  - einschraenkungen:
    * b: zweier oder zehnerpotenz
    * b: computerwort
    * m: eine stelle weniger als b
    * m: $x21$ endet
    * m, x: gerade zahl
    * m, x: ohne besonderes muster
  - methode der linearen kongruenz;

    ```
    a [i] = (a[i-1]*b+1) % m;
    ```

  - hash-tabelle:

    ```
    x = (x*b+a[i]) % m;
    ```

- alternative formulierung, aehnlichkeit:

```
a [i] = (a[i-1]*b+1) % m;
a [i] = (a[i-1]*b+x[i]) % m;
```

- theorie, modell, methode:
  - theorie:
  - modell:
  - methode:

    ```
    a [i] = (a[i-1]*b+1) % m;
    ```

  - klassenmodell, uml:
    * methode, name, attribute
    * methode:

      ```
      a [i] = (a[i-1]*b+1) % m;
      # methodenname:
      rand ()
      ```

    * funktion: argument, rechenvorschrift, name
    * tupel, paar, liste:
      · relationale datenbank:
      · datenbank management system:
      · ERM: relationale Datenbank: SQL MySQL:
  - listen, paare:

$$(x_1, x_2) \tag{1}$$
$$(x, y, z) \tag{2}$$
$$(a_0, a_1, \ldots, a_n) \tag{3}$$
$$(a_n, a_{n-1}, \ldots, a_1, a_0) \tag{4}$$
$$(s_n, s_{n-1}, \ldots, s_1, s_0) \tag{5}$$
$$\tag{6}$$

– Karthesisches Produkt:

$$M \times N \tag{7}$$
$$M \times M \times M \tag{8}$$
$$\tag{9}$$

...

– Relation: Untermenge von Karthesischem Produkt

$$S \subseteq M \times N \tag{10}$$
$$Q \subseteq M \times M \times \ldots M \tag{11}$$
$$\tag{12}$$

– Relationale Datenbank:
– untermenge von karthesischen produkten
– Mathematik:
– Karthesische Produkte, attribute, Variablen, werte:

$$(x_1, x_2) \tag{13}$$
$$(x, y, z) \tag{14}$$
$$(a_0, a_1, \ldots, a_n) \tag{15}$$
$$(a_n, a_{n-1}, \ldots, a_1, a_0) \tag{16}$$
$$(s_n, s_{n-1}, \ldots, s_1, s_0) \tag{17}$$
$$\tag{18}$$

– funktionen:

$$(f(x), g(x), h(x)) \tag{19}$$
$$\tag{20}$$

stapelfunktionen, mehrdimensionale analysis:

$$(f(x), g(x), h(x)) \tag{21}$$
$$\tag{22}$$

unterschiedlich:

$$(f(x_1), g(z), h(y_2)) \tag{23}$$
$$(f_2(x_1), f_3(z_0), g(z), h(y_2)) \tag{24}$$
$$(f_1(a_0, a_1, \ldots, a_n), f_2(a_0, a_1, \ldots, a_n), \ldots, f_m(a_0, a_1, \ldots, a_n)) \tag{25}$$
$$\tag{26}$$

– eine funktion hat einen namen
– eine methode hat einen namen
– eine menge von methoden
– eine menge von funktionen

- eine kartheisches produkt von mengen
- ein kartheisches produkt von mengen von funktionen
- ein element einer menge enthält funktionen, argumente
- parameter, trägt einen namen
- die klasse, in UML
  - methode der linearen kongruenz
  - UML
  - die klasse:
    - name
    - methoden
    - attribut
  - Assoziation
  - Vererbung
  - Graphentheorie:
    - paare:
    - Tupel
  - Paare in der Graphentheorie:
- Knoten
- Kanten
- Graphen
  - Knoten, V, $V = \{1, 2, 3, \dots\}$
  - Kanten: $E = \{(1, 2), (1, 3), \dots (n, m)\}$
  - Graphen, $G = (V, E)$, $E = V \times V$
  - Knoten haben Nachbarn
  - Tupel, Paare, ...
  - (Relation: gleich ungleich, trichonometrie)
  - (Relationale Datenbank)
  - Graphentheorie
  - Graphentheorie: paare
  - Nachbar und Nachbar
  - UML:
  - Assoziation: Klasse A und Klasse B
  - Verschlüsselung:
    - Allice und Bob
    - Marx: Händler A, Händler B, Verkäufer, Käufer
    - Interaktionsdiagramm:, Kaufvertrauf, Kauf und Verkauf
    - UML: Klasse A, Klasse B
    - Klasse A und Klasse B, sind Elemente von Mengen:, mit Variablen, Attributen, ...
    - Methoden, und haben namen
    - Assoziation: zwei Klassen kennen sich, Vererbung: eine untermenge

        - z.B:
          $$A : \{\{f(x), g(x), h(x), (x, y, z)\}$$
        - Assoziation
          $$(A, B)$$
        - Vererbung
          $$A \subseteq B$$
- Algorithmen
    * Sortieralgorithmen
    * Suchalgorithmen
    * Sortieren:
        - insertion, selection
        - shell bubble
        - rekusiv: quick
        - digitales sortieren
        - bits
        - radix exchange sort
        - straight radix sort
        - digital: bits und radix (wurzel)
        - priotaetswarteschlangen:
        - datenstrukturen informatik:
        - heap, stack, schlange
        - heapsort
        - mergesort
        - digitales sortieren, externes sortieren
        - insertion
        - selection
        - shell
        - bubble
        - quick
        - merge
        - heap
        - sort
        - suchen:
        - baeume:
        - binaere suchbaerem
        - rot-schwarzbaeume
        - baeume generell: graphentheorie
        - wurzelbaeume, aufspannende baeume
        - minimal aufspaenndende baeume
        - suche:
        - sequentielle
        - binaere

· binaere: ist nicht gleich binaerbaum
· wahrscheinlich: bud buddy
· sequentiell, linear
· binaer
· suchen
· suchen in zeichenfolgen
· verarbeitung von zeichenfolgen
· suchen in zeichenfolgen
· pattern matching
· suchen in zeichen:
· grob
· knuth morris pratt
· boyer moore
· rabin karb
· daneben: cook und gosper
· pattern matching:
· automat
· backend
· frontend
· compiler:
· backend
· frontend
· quellsprache
· zielsprache
· bsp: quellsprache C
· Zielsprache: amd64, i586, x86, 8080, ...
· Sprache:
· BNF: Bakus Naur Form
· begriffe:
· DNF: Disjunktive Normalform
· KNF: Konjunktive Normalform
· BNF: Bakus Naur Form
· Grammatik!
· Formale Sprache, grammatik
· Zeichen, mustererkennungsautomat
· deterministischer automat
· nicht deterministischer automat!
· terminiert, deterministisch!
· KMP: Suchen in dem eigenen muster, nach dem Anfang
· wie im Text
· Folge
· Oder
· Closure (Huellenbildung)

```
AB
A+B
(AB)*

BNF, mathe
5 = (4+2)*3 - 17

Distributiv
Assoziativ
Kommutativ

+ Operator
* Operator
() Klammern
```

- Koerper, Halbgruppe, Monoid,
- mathematik, definieren
- der ganzen, natuerlichen, ... zahlen
- Boolesche algebra:

```
AND, OR, NOT, ()

Expr
Term
Fact

<Expr> ::= <Term> | <Term> '+' <Expr>
<Term> ::= <Fact> | <Fact> '*' <Term>
<Term> ::= '('<Expr>')' | v | c
```

- terminierungssymbol: v
- terminiert, nicht terminiert
- deterministisch, nicht deterministisch
- BNF:
- Grammatik, Sprache, Formale Sprache
- Automat: Zeicheerkennung:
- Frontend sprache,
- Backend: Automat
- Automat:
- Darstellung, Simulation
- Zufallszahlen:
- Methode, Test
- Compiler:
- Quellsprache, Zielsprache
- Froentend Backend!
- Vorsicht! Grammatik!

- Dazwischen, Parser
- Parser:
- Bottom-Up
- Top-Down (rekursiver Abstieg)
- Grammatik
- Bottom-Up
- Top-Down
- Compiler
- Compiler-Compiler (lex, yacc)
- Pattern Matching:
- BNF fuer Reguläere Ausdrücke
- Automat
- auswendig:
- Alpha Brava Charlie Delta
- Alpha in der Mitte
- Alpha Bravo Alpha Charlie Delta
- ABA
- auswendig:

  ```
  10100111 fuer KMP
  ```

- auswendig

  ```
  state
  ch [state]
  next1 [state]
  next2 [state]
  ```

- state: 1 2 3 4 5 6 7 8
- unterscheiden: KMP und Pattern Matching automat

  ```
  0 1 2 3 4 5 6
  0 0 1 2 1 1 0
  ```

- so was
- und:
- pattern matching:
- leerzustaende
- ok, hier pause
- Boyer-Moore von hinten herum
- Rabin-Karp:
- Hashtable

  ```
  a [i]*d^(M-1)
  ```

- polynome oder zahlen:

```
a [i]*d(M-1) + a [i-1]*d^(M-2) + ... + a [0]
```

$$a_n \cdot d^{n-1} + \cdots + a_1 \cdot d^1 + a_0$$

- beim hashing:
- von muter p
- und text a
- vorne immer eines raus.
- hinten dran
- und modulo q
- ok.
- arithmetik, zufallszahlen:
- sortieralgorithmen
- suchalgorithmen
- bearbeitung von zeichenfolgen
- graphen
- arithmetik
- arithmetik:
- zufallszahlen
- integration
- kurvenanpassung
- kurvenanpassung
- lagrange: interpolation
- splines: interpolation
- mit polynomen
- mit quadraten
- mit einheitswurzeln
- FFT: Fast Fourier Transformation:
- wie Riemannsches Integral
- FFT: mit Interpolation
- Interpolation: polynom: Lagrange
- Splines
- Quadrate
- Verfahren:
- Einfach
- Zusammengesetzt
- Adaptiv
- Einfache Quadraturverfahren
- Zusammengesetzte Verfahren
- Adaptive Quadratur

- Quadraturverfahren
- Quadratur
- Verfahren
- Das ist: Integration:
- Und Integration, schule:
- Symbolisch + compilerbau, Stammfunktion
- symbolische ableitung
- schule:
- differentialrechnung:
- steigung,
- hochpunkt, tiefpunkt
- wichtig:
- abszisse - rechtsaxe
- ordinate - hochachse

```
ord (ch)
chr (n)
```

- ch - right von nummer
- und ordinalwert
- wuerfelspieler:
- zufallexperiment
- oder teil von zufallsexperiment, wurf n:
- abszisse, wachsende zeit t
- nein: wurf ...
- ordinate: 1, 2, 3, 4, 5, 6
- und:
- differntialrechnung:
- tangente, steigung
- hochpunkt, tiefpunkt, wendepunkt
- integralrechnung: riemann, flaeche,
- numerisch, sonst symbolisch
- Kurvenanpassung
- fuehrt zu FFT
- arithmetik ist
- multiplikation von polynomen
- interpolation
- interpolation: lagrange
- lagrange und nicht langrage
- ok.

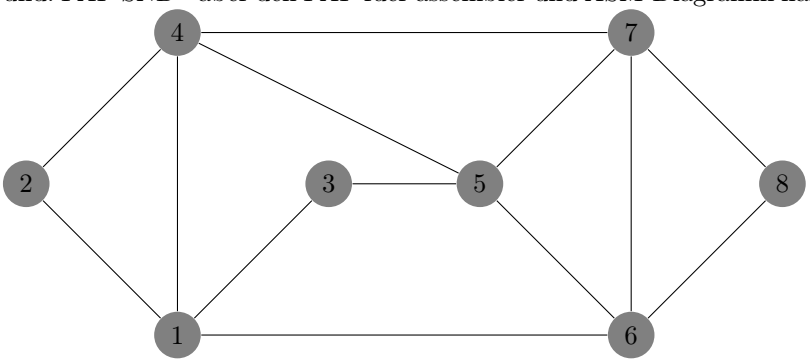
– kurvenanpassung:
– interpolation, lagrange
– splines
– daneben: kleinste quadrate
– methode der kleinsten quadrate
– integration:
– symbolisch
– einfache quadraturverfahren
– quadratur verfahren quadraturverfahren
– einfach adaptiv zusammengesetzt
– FFT: Einheitswurzeln
– Einheitswurzeln selber:
– Definition usw.
– Interpolation
– ok.
– theorie: heisst:
– modell: heisst:
– system und maschine
– kryptosystem
– datenverarbeitungssystem
– graphikverarbeitungssystem
– computersystem
– ...
– jetzt spaeter weiter:
– methode:
– beweismethode:
– vollstaendige induktion, reductio ad absurdum, kontraposition
– methode:
– hash-table, zufallszahlen
– methode:
– aufschreiben:
– geschichtliches in anmerkungen?
– methode bei funktionen:
– funktionsvorschrift, tabelle,
– tabelle, kein roman, kein exception handler, keine fahrgast?? name
– flugschreiber, beim flughafen
– normale tabelle, damit das passiert, jetzt
– lagrange, mit weniger variablen,

– und einer tabelle, hashtabelle normal usw.

– das schoener machen, dann tiefensuche, breitensuche und eulertour auch in tabelle

[1]Vermutlich: Marx. . . , heute: Haushaltstheorie, Mikroökonomie, Theorie, Modell, Low Budget, blos hauswirtschaft: Früher hätte Marx gesagt: ein mensch braucht am tag so und so viel wolle, so und so viel Kaffee, und das braucht der mensch wirklich und aufgerechnet, in seinem Leben: so und so viel bibeln: 5 Bibeln zum Beispiel. Lässt sich nicht einfach halten, vermutet bei Einzelhandel, vermutung: muss gelernt sein, bei Bibeln normal: der mensch braucht: 5 Bibeln z.B. im Leben, weil Besitz ist schwierig zu halten, der Einzelhandel muss auch das halten und droht jederzeit, also verluste in dieser form sind normal, und jeder will ihm unter umständen helfen, soll man keine witze machen, . . . , und: dasselbe bei aktien, also lassen sich aus verschiedenen gründen nicht einfach halten.. . . , ruckelige angelengenheit beim fliegen, paar mal angstschweiss beim flugplatz, schreien im flugzeug, vielleicht andere wetterverhältnisse, politisch fliegen anders, aber so auch mit bibeln und co kg, hängt von der sache ab. anmerkung: hin und wieder sachen umtauschen auf der strasse lohnt sich. kätzerlehre hat verschiedene stärke vielleicht oder ausrechnung, selbst innerhalb, was sind bücher, schon mal welche gehabt, oder wie ist unsere beziehung zum koran, der ist bei mir sehr sicher, kennen wir den wirklich, vielleicht fehlt uns das wirkliche quentchen, was mir meinen bei dem anderen auch nicht zu haben.

[2]jetzt eine runde Spaziergang . . .
dann mache ich ordentlich weiter, tiefensuche breitensuche, eulertour . . .
und so weiter, alles per Tabelle, den text vorher, ordentlich zusammenfassen, nicht fertig schreiben
dann: interpolation Lagrange . . .
wenn ich das habe, auch besonders quine mc cluskey, das kommt in die anmerkungen - AND OR PORT usw.
und: PAP SND - aber den PAP fuer assembler und ASM-Diagramm hardware. . .



Das kann

man jetzt in eine tabelle und menge aufnehmen

$$
\begin{aligned}
V &= \{1,2,3,4,5,6,7,8\} && (27)\\
E &= \{(1,2),(1,3),(1,4),(1,6), && (28)\\
&= (2,4),(4,1), && (29)\\
&= (3,1),(3,5). && (30)\\
&= (4,2),(4,1),(4,5),(4,7), && (31)\\
&= (5,3),(5,4),(5,6),(5,7), && (32)\\
&= (6,1),(6,5),(6,7),(6,8), && (33)\\
&= (7,4),(7,5),(7,6),(7,8), && (34)\\
&= (8,6),(8,7)\} && (35)\\
G &= (V,E) && (36)\\
& && (37)
\end{aligned}
$$

hier so sieht das als verbatim aus: teil der aufgabe

```
\begin{tikzpicture}
\node [circle, fill=gray] (circle1) at (2,0) {1};
\node [circle, fill=gray] (circle2) at (0,2) {2};
\node [circle, fill=gray] (circle3) at (4,2) {3};
\node [circle, fill=gray] (circle4) at (2,4) {4};

\node [circle, fill=gray] (circle5) at (6,2) {5};
\node [circle, fill=gray] (circle6) at (8,0) {6};
\node [circle, fill=gray] (circle7) at (8,4) {7};
\node [circle, fill=gray] (circle8) at (10,2) {8};

\draw (circle4) -- (circle7);
\draw (circle7) -- (circle8);
\draw (circle8) -- (circle6);
\draw (circle6) -- (circle5);
\draw (circle5) -- (circle7);
\draw (circle3) -- (circle5);
\draw (circle6) -- (circle7);
\draw (circle6) -- (circle1);
\draw (circle5) -- (circle4);
\draw (circle3) -- (circle1);
\draw (circle1) -- (circle2);
\draw (circle2) -- (circle4);
\draw (circle1) -- (circle4);
\end{tikzpicture}
Das kann man jetzt in eine tabelle und menge aufnehmen
\begin{eqnarray}
V & = & \{1,2,3,4,5,6,7,8\}\\
E & = & \{(1,2),(1,3),(1,4),(1,6), \\
& = & (2,4),(4,1),\\
& = & (3,1), (3,5). \\
& = & (4,2), (4,1), (4,5), (4,7),\\
```

```
& = & (5,3), (5,4), (5,6), (5,7),\\
& = & (6,1), (6,5), (6,7), (6,8),\\
& = & (7,4), (7,5), (7,6), (7,8),\\
& = & (8,6), (8,7)\}\\
G & = & (V,E)\\
\end{eqnarray}
```

so, jetzt kommt

- tiefensuche
- breitensuche
- eulertour

als, klassische aufgabe mit tabelle. zunächst brauchen wir algorithmen von tiefen und breitensuche.

```
pred[r] = r component[r] = r Q.Append(r)
while Q.IsNotEmpty():
v = Q.Top()
for w in Neighborhood(v):
 if pred[w] == None:
  pred[w] = v
component[w] = component[v] Q.Append(w)
```

[3]anmerkung: ob das jetzt geht? wahrscheinlich nicht! erst auswendig lernen! - trotzdem der algo, morgen, beim nächsten mal [4]anmerkung: man sollte zu den verbatim und pictures einen entsprechenden namen, eine beschriftung unten dran machen es kommt die andere suche:

```
def DFS(v):
global step
label[v] = step
step = step + 1
for w in Neighborhood(v):
 if label[w]==None: pred[w]=v DFS(w)
  return
step=1
DFS(r)
```

und jetzt haben wir das [5]anmerkung: DFS: Depth first search [6]anmerkung: ich muss das heute nachmittag auswendig lernen, dann was anderes was kann/muss ich tun?

- eulertour, graphisch und mathematisch geht
- UML Diagramm exemplarisch wäre schön
- valenzen zählen pro knoten ist machbar
- quine mc cluskey
- ein einfaches assembler programm mit PAP

- dann: lagrange - interpolation mit grad 3.
- lagrange vielleicht noch, so zu verzusagen, vergroessern, verkleinern intervall
- einzeichnen usw.
- hashtabelle
- zufallszahlen, alles mit tabelle

mathematisch eulertour, ohne aufsehend erregende tabelle. Merksatz: [7]anmerkung: das wort merksatz, siehe mathematische grundlage luise unger, theorem, usw. diese begriffe sind teil von mathe. merksatz klingt nach schule, wenn man korrekt mathematisch schreibt, muss man zwischen satz, merksatz und so weiter unterscheiden, dass man dann auch hinschreiben, dann heisst es in der fachsprache trotzdem merksatz.

1. zunächst solte man besonders funktionen, methoden, [8]der begriff wurde eben erläutert: die zufallszahlen sind eine methode nach lehmer, sind ziemlich funktion, so wie in der klasse, methode. sie gleichen oft einer folge, sind algorithmisch anwendbar. beweismethoden lassen sich ebenso: algorithmisch anwenden. [9]eine tabelle - es heisst hashtabelle - ist die hashtabelle nicht einfach: eine funktion, oder sie liefert keine - bei robert sedgewick, wohl mit absicht etwas irrefuehrend, eine tabelle - cash speicher - einfüren eine funktion ist eine tabelle. [10]warum eine funktion eine tabelle ist .... warum die verwirrenden namen. wir kennen das von AVR - Advanced Risk Architecture, ARM - Advanced Risk Machine SOIC - Small outline integrated circuit - und: SOC - System on a chip - so, viele namen, sind oft bewusst so - dass man trotzdem manchmal glaubt, das kennt man von der kanzlei oder kurrentschrift: ist das wirklich das gleiche, ist es wirklich so oder bietet es sogar zwei möglichkeiten, wie wir auch von begriffen, methode und procedure und theorie und modell kennen? ... sollte man so darstellen, dass sie immer einer tabelle ist, wenn es so ist.

2. um allerdings diese tabelle zu verstehen, sollte man folgendes: [11]ich denke wieder an den flughafen und unseren nadeldrucker, eine andere form ist wieder da, die am flughafen bände von passagieren raus lässt oder - ein exception handler in java. ich denke, wenn man keine direkte tabelle nimmt, so wie wirklich, dann kann es bei so grossen wie hier, zu langen - nicht mathematischen listen sondern solchen listen hier kommen, das ist aber dann gut, wenn: man wissen nuss, wie im konkreten falle die tabelle tut[12]warum? weil: das ist so wie mit tikz und so weiter, oder wie auch immer, man kommt wie weit man kommt, man auch eulertouren ganz graphisch machen oder so oder so - so weit ...

$$
\begin{aligned}
V &= \{1,2,3,4,5,6,7,8\} && (38)\\
E &= \{(1,2),(1,3),(1,4),(1,6), && (39)\\
&= (2,4),(4,1), && (40)\\
&= (3,1),(3,5). && (41)\\
&= (4,2),(4,1),(4,5),(4,7), && (42)\\
&= (5,3),(5,4),(5,6),(5,7), && (43)\\
&= (6,1),(6,5),(6,7),(6,8), && (44)\\
&= (7,4),(7,5),(7,6),(7,8), && (45)\\
&= (8,6),(8,7)\} && (46)\\
G &= (V,E) && (47)\\
& && (48)
\end{aligned}
$$

und jetzt eine art liste. man streiche einfach das, was als nächstes weg kommt.

$$
\begin{aligned}
V &= \{1,2,3,4,5,6,7,8\} && (49)\\
E &= \{(1,2),(1,3),(1,4),(1,6), && (50)\\
&= (2,4),(4,1), && (51)\\
&= (3,1),(3,5). && (52)\\
&= (4,2),(4,1),(4,5),(4,7), && (53)\\
&= (5,3),(5,4),(5,6),(5,7), && (54)\\
&= (6,1),(6,5),(6,7),(6,8), && (55)\\
&= (7,4),(7,5),(7,6),(7,8), && (56)\\
&= (8,6),(8,7)\} && (57)\\
G &= (V,E) && (58)\\
& && (59)
\end{aligned}
$$

jetzt ein zweites ding, wo die landen die schon gebraucht sind.

$$
\begin{aligned}
V &= \{1,2,3,4,5,6,7,8\} && (60)\\
E &= \{\overline{(1,2)}^{1},(1,3),(1,4),(1,6), && (61)\\
&= \overline{(2,4)}^{2},\overline{(2,1)}^{1}, && (62)\\
&= (3,1),(3,5). && (63)\\
&= \overline{(4,2)}^{2},(4,1),(4,5),(4,7), && (64)\\
&= (5,3),(5,4),(5,6),(5,7), && (65)\\
&= (6,1),(6,5),(6,7),(6,8), && (66)\\
&= (7,4),(7,5),(7,6),(7,8), && (67)\\
&= (8,6),(8,7)\} && (68)\\
G &= (V,E) && (69)\\
F &= \{(1,2),(2,4)\} && (70)\\
& && (71)
\end{aligned}
$$

1 quine/mc cluskey

```
(C) David Vajda
Wed Oct 21 19:58:16 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    1
 5 0 1 0 1    0
 6 0 1 1 0    1
 7 0 1 1 1    0
 8 1 0 0 0    0
 9 1 0 0 1    0
10 1 0 1 0    1
11 1 0 1 1    1
12 1 1 0 0    0
13 1 1 0 1    1
14 1 1 1 0    0
15 1 1 1 1    1
```

und es geht weiter. als normale tabelle, std. uebungsaufgabe.

(C) David Vajda
Wed Oct 21 19:58:16 2026
4 Network - TTL - Disjunktive Normalform

n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

jetzt noch mal in ordentlicher form:

(C) David Vajda Wed Oct 21 19:58:16 2026 4 Network - TTL - Disjunktive Normalform					
n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
6	0	1	1	0	1
10	1	0	1	0	1
11	1	0	1	1	1
13	1	1	0	1	1
15	1	1	1	1	1

so sieht es ordentlich aus, machen wir aber nicht so.

```
(C) David Vajda
Wed Oct 21 19:58:16 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    1
 6 0 1 1 0    1
10 1 0 1 0    1
11 1 0 1 1    1
13 1 1 0 1    1
15 1 1 1 1    1
```

jetzt gruppieren.

```
(C) David Vajda
Wed Oct 21 19:58:16 2026
4 Network - TTL - Disjunktive Normalform

Gruppe 0:
 0 0 0 0 0    1
Gruppe 1:
 1 0 0 0 1    1
 2 0 0 1 0    1
 4 0 1 0 0    1
Gruppe 2:
 3 0 0 1 1    1
 6 0 1 1 0    1
10 1 0 1 0    1
Gruppe 3:
11 1 0 1 1    1
13 1 1 0 1    1
```

```
Gruppe 4:
15 1 1 1 1    1
```

dann jetzt verschmelzen/zusammenfassen:

```
(C) David Vajda
Wed Oct 21 19:58:16 2026
4 Network - TTL - Disjunktive Normalform

Gruppe 0:
 0 0 0 0 0    1
Gruppe 1:
 1 0 0 0 1    1
 2 0 0 1 0    1
 4 0 1 0 0    1
Gruppe 2:
 3 0 0 1 1    1
 6 0 1 1 0    1
10 1 0 1 0    1
Gruppe 3:
11 1 0 1 1    1
13 1 1 0 1    1
Gruppe 4:
15 1 1 1 1    1

0:1  0 0 0 x
0:2  0 0 x 0
0:4  0 x 0 0
1:3  0 0 x 1
2:3  0 0 1 x
2:6  0 x 1 0
2:10  x 0 1 0
4:6  0 1 x 0
3:11  x 0 1 1
10:11 1 0 1 x
11:15 1 x 1 1
13:15 1 1 x 1
```

perfekt, dieses sortieren usw. also, weiter verschmelzen, könnte man auch PAP oder ASM-Diagramm machen, später

- flow chart - vielleicht u.a. regeln des alltags
- PAP - assembler - atmega8 und nasm, x86, i586, amd64, ...
- ASM-Diagramm + RTL-Notation: Hardware Algorithmus

```
0:1  0 0 0 x
0:2  0 0 x 0
0:4  0 x 0 0
1:3  0 0 x 1
2:3  0 0 1 x
```

```
2:6  0 x 1 0
2:10  x 0 1 0
4:6  0 1 x 0
3:11  x 0 1 1
10:11 1 0 1 x
11:15 1 x 1 1
13:15 1 1 x 1


2:10  x 0 1 0
3:11  x 0 1 1

0:4  0 x 0 0
2:6  0 x 1 0
11:15 1 x 1 1

0:2  0 0 x 0
1:3  0 0 x 1
4:6  0 1 x 0
13:15 1 1 x 1

0:1  0 0 0 x
2:3  0 0 1 x
10:11 1 0 1 x
```

jetzt möglichst numerisch sortieren

```
2:10  x 0 1 0
3:11  x 0 1 1

0:4  0 x 0 0
2:6  0 x 1 0
11:15 1 x 1 1

0:2  0 0 x 0
1:3  0 0 x 1
4:6  0 1 x 0
13:15 1 1 x 1

0:1  0 0 0 x
2:3  0 0 1 x
10:11 1 0 1 x
```

ok, sortiert, was vor den punkt betrifft, bessere schreibweise in tupel, paaren mit komma

```
(2,10) x 0 1 0
(3,11) x 0 1 1

(0,4)  0 x 0 0
(2,6)  0 x 1 0
```

```
(11,15) 1 x 1 1

(0,2)  0 0 x 0
(1,3)  0 0 x 1
(4,6)  0 1 x 0
(13,15) 1 1 x 1

(0,1)  0 0 0 x
(2,3)  0 0 1 x
(10,11) 1 0 1 x
```

dann inzwischen in eine ordentliche LaTeXTabelle

n	x_3	x_2	x_1	x_0
$(2,10)$	x	0	1	0
$(3,11)$	x	0	1	1
$(0,4)$	0	x	0	0
$(2,6)$	0	x	1	0
$(11,15)$	1	x	1	1
$(0,2)$	0	0	x	0
$(1,3)$	0	0	x	1
$(4,6)$	0	1	x	0
$(13,15)$	1	1	x	1
$(0,1)$	0	0	0	x
$(2,3)$	0	0	1	x
$(10,11)$	1	0	1	x

oder, so:

n	x_3	x_2	x_1	x_0
$(2,10)$	x	0	1	0
$(3,11)$	x	0	1	1
$(0,4)$	0	x	0	0
$(2,6)$	0	x	1	0
$(11,15)$	1	x	1	1
$(0,2)$	0	0	x	0
$(1,3)$	0	0	x	1
$(4,6)$	0	1	x	0
$(13,15)$	1	1	x	1
$(0,1)$	0	0	0	x
$(2,3)$	0	0	1	x
$(10,11)$	1	0	1	x

der nächste schritt, die sache noch mal, so weit sie geht.

```
(2,10) x 0 1 0
(3,11) x 0 1 1

(2,10,3,11)  x 0 1 x

(0,4)  0 x 0 0
```

```
(2,6)  0 x 1 0
(11,15) 1 x 1 1

(0,4,2,6)   0 x x 0
(11,5)   1 x 1 1

(0,2)  0 0 x 0
(1,3)  0 0 x 1
(4,6)  0 1 x 0
(13,15) 1 1 x 1

(0,2,1,3)   0 0 x x
(0,2,4,6)   0 x x 0
(13,15)   1 1 x 1

(0,1)  0 0 0 x
(2,3)  0 0 1 x
(10,11) 1 0 1 x

(0,1,2,3)   0 0 x x
(2,3,10,11)  x 0 1 x
```

ok, jetzt das zeug raus, was wieder geht.

```
(2,10,3,11)  x 0 1 x
(0,4,2,6)   0 x x 0
(11,15)   1 x 1 1
(0,2,1,3)   0 0 x x
(0,2,4,6)   0 x x 0
(13,15)   1 1 x 1
(0,1,2,3)   0 0 x x
(2,3,10,11)  x 0 1 x
```

ok, im nächsten schritt, wieder, redundante (terme?) raus und richtig sortieren. usw.

```
(2,10,3,11)  x 0 1 x
(2,3,10,11)  x 0 1 x

(0,4,2,6)   0 x x 0
(0,2,4,6)   0 x x 0

(0,2,1,3)   0 0 x x
(0,1,2,3)   0 0 x x

(11,15)   1 x 1 1

(13,15)   1 1 x 1
```

hier redundanz lieber raus

```
(2,10,3,11)  x 0 1 x
(0,4,2,6)   0 x x 0
(0,2,1,3)   0 0 x x
(11,15)   1 x 1 1
(13,15)   1 1 x 1
```

weiter lässt sich auch nichts mehr machen. wie man sieht. jetzt primimplikantentafel.

```
 0 1 2 3 4 6 10 11 13 15


(2,10,3,11)  x 0 1 x
(0,4,2,6)   0 x x 0
(0,2,1,3)   0 0 x x
(11,15)   1 x 1 1
(13,15)   1 1 x 1
```

so sehe ich das bisher, vergleiche ich mit der wahrheitstafel oben.

```
(C) David Vajda
Wed Oct 21 19:58:16 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    1
 6 0 1 1 0    1
10 1 0 1 0    1
11 1 0 1 1    1
13 1 1 0 1    1
15 1 1 1 1    1
```

das passt optimal, jetzt wirklich:

```
   0 1 2 3 4 6 10 11 13 15
(2,10,3,11)   1 1   1 1
(0,4,2,6)  1  1  1 1
(0,2,1,3)  1 1 1 1
(11,15)         1  1
(13,15)          1 1
```

		0	1	2	3	4	6	10	11	13	15
0	$(2,10,3,11)$			1	1			1	1		
1	$(0,4,2,6)$	1			1	1	1				
2	$(0,2,1,3)$	1	1	1	1						
3	$(11,15)$								1		1
4	$(13,15)$									1	1
		x	2	x	x	1	1	0	3	4	x

jetzt die primimplikanten, mal anders: ok, alle sind dran beteiligt, also, folgendes: dann stelle ich jetzt das gleichungssystem auf.

```
(2,10,3,11)  x 0 1 x
(0,4,2,6)   0 x x 0
(0,2,1,3)   0 0 x x
(11,15)   1 x 1 1
(13,15)   1 1 x 1

 y := not x2 and x1 or
   not x3 and not x0 or
   not x3 and not x2 or
   x3 and x1 and x0 or
   x3 and x2 and x0;
```

dann mit galasm

so typisches gatter von aussen, ic - dip/dil

```
A1  VCC
B1  A3
C1  B3
A2  C3
A3  A4
C3  B4
GND  C4
```

daraus wird dann auch, das für den

```
gal16v, galasm
```

man kann sich das so vorstellen und nun so umdrehen:

```
A1 B1 C1 A2 B2 C3 GND
C4 B4 A4 C3 B3 A3 VCC
```

und nun bei galasm

```
CLK X3 X2 X1 X0 NC NC NC NC GND
/OE Y NC NC NC NC NC NC NC VCC
```

generell: im westen ist Input
im osten output,

```
CLK: input
/OE Output
```

jetzt die schaltung:

```
GAL16V8
MODULE GAL16V8QUINDNF10202026

CLK X3 X2 X1 X0 NC NC NC NC GND
/OE Y NC NC NC NC NC NC NC VCC

 Y = /X2 * X1 +
   /X3 * /X0 +
```

```
   /X3 * /X2 +
   X3 * X1 * X0 +
   X3 * X2 * X0

DESCRIPTION:
UEBUNG WIE JEDEN TAG
```

oder probieren als so:

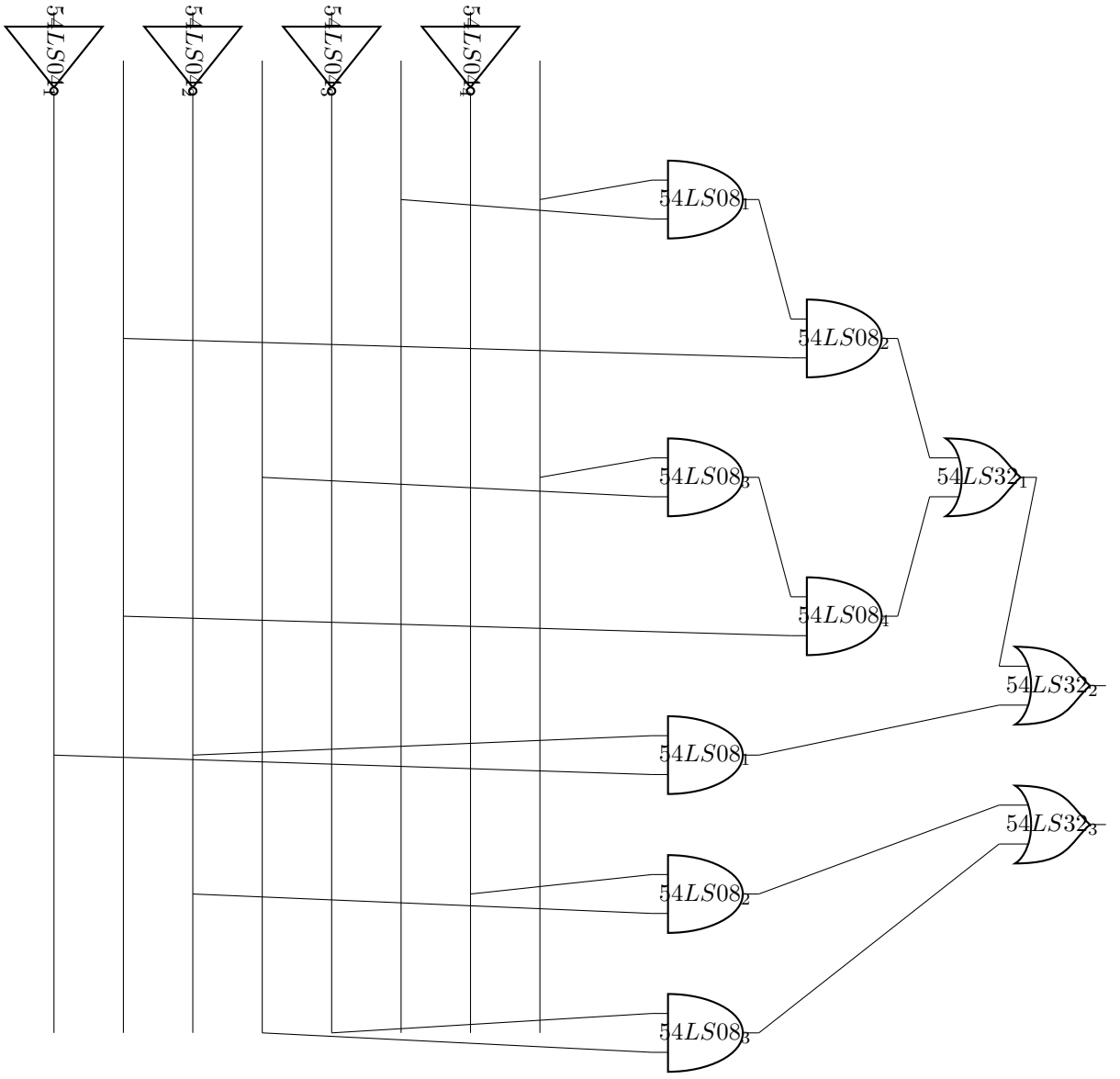
```
GAL16V8
MODULE GAL16V8QUINDNF10202026

CLK X3 X2 X1 X0 NC NC NC NC GND
/OE Y NC NC NC NC NC NC NC VCC

Y = /X2 * X1 + /X3 * /X0 + /X3 * /X2 + X3 * X1 * X0 + X3 * X2 * X0

DESCRIPTION:
UEBUNG WIE JEDEN TAG
```

ok, nächster schritt, jetzt geht es los, mit circuitikz. der schaltplan.



perfekt, so habe ich das gemacht:

```
\begin{circuitikz}
\node [and port] (and1) at (10,12) {$54LS08_1$};
\node [and port] (and2) at (12,10) {$54LS08_2$};
\node [and port] (and3) at (10,8) {$54LS08_3$};
\node [and port] (and4) at (12,6) {$54LS08_4$};
\node [and port] (and5) at (10,4) {$54LS08_1$};
\node [and port] (and6) at (10,2) {$54LS08_2$};
\node [and port] (and7) at (10,0) {$54LS08_3$};

\draw (and1.out) -- (and2.in 1);
\draw (and3.out) -- (and4.in 1);
```

```
\node [not port, rotate=270] (not1) at (0,14) {$54LS04_1$};
\node [not port, rotate=270] (not2) at (2,14) {$54LS04_2$};
\node [not port, rotate=270] (not3) at (4,14) {$54LS04_3$};
\node [not port, rotate=270] (not4) at (6,14) {$54LS04_4$};

\draw (not1.out) -- (0,0);
\draw (not2.out) -- (2,0);
\draw (not3.out) -- (4,0);
\draw (not4.out) -- (6,0);

\draw (1,14) -- (1,0);
\draw (3,14) -- (3,0);
\draw (5,14) -- (5,0);
\draw (7,14) -- (7,0);


\node [or port] (or1) at (14,8) {$54LS32_1$};
\node [or port] (or2) at (15,5) {$54LS32_2$};
\node [or port] (or3) at (15,3) {$54LS32_3$};

\draw (and2.out) -- (or1.in 1);
\draw (and4.out) -- (or1.in 2);
\draw (or1.out) -- (or2.in 1);
\draw (and5.out) -- (or2.in 2);
\draw (and6.out) -- (or3.in 1);
\draw (and7.out) -- (or3.in 2);

\draw (and1.in 1) -- (7,12);
\draw (and1.in 2) -- (5,12);
\draw (and2.in 2) -- (1,10);
\draw (and3.in 1) -- (7,8);
\draw (and3.in 2) -- (3,8);
\draw (and4.in 2) -- (1,6);
\draw (and5.in 1) -- (2,4);
\draw (and5.in 2) -- (0,4);
\draw (and6.in 1) -- (6,2);
\draw (and6.in 2) -- (2,2);
\draw (and7.in 1) -- (4,0);
\draw (and7.in 2) -- (3,0);
\end{circuitikz}
```

sehr, gut, jetzt kommt:

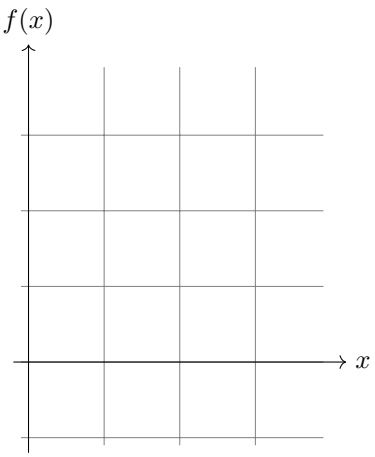
1. das programm für i586, nasm mit pap
2. für atmega8 mit PAP

und wenn das passiert ist, noch lagrange. oder vorher.

in die anmerkungen:

$$
\begin{aligned}
X &= (1,2,3) && (72)\\
X &= (x_0 = 1, x_1 = 2, x_2 = 3) && (73)\\
Y &= (2,7,4) && (74)\\
l_i(x) &= \sum_{\substack{j=0\\ j\neq i}}^{3} \frac{x - x_j}{x_i - x_j} && (75)\\
l_0(x) &= \frac{x-2}{1-2} \cdot \frac{x-3}{1-3} && (76)\\
l_1(x) &= \frac{x-1}{2-1} \cdot \frac{x-3}{2-3} && (77)\\
l_2(x) &= \frac{x-1}{3-1} \cdot \frac{x-2}{3-2} && (78)\\
l_0(x) &= \frac{(x-2)\cdot(x-3)}{1-3} && (79)\\
l_1(x) &= \frac{(x-1)\cdot(x-3)}{2-3} && (80)\\
l_2(x) &= \frac{(x-1)\cdot(x-2)}{3-2} && (81)\\
l_0(x) &= \frac{(x-2)\cdot(x-3)}{1-3} = \frac{(x^2 - 2\cdot x - 3\cdot x + 6)}{-2} && (82)\\
l_1(x) &= \frac{(x-1)\cdot(x-3)}{2-3} = \frac{(x^2 - 3\cdot x - x + 3)}{-1} && (83)\\
l_2(x) &= \frac{(x-1)\cdot(x-2)}{3-2} = \frac{(x^2 - 2\cdot x - x + 2)}{1} && (84)\\
l_0(x) &= \frac{(x-2)\cdot(x-3)}{1-3} = \frac{(x^2 - 5\cdot x + 6)}{-2} && (85)\\
l_1(x) &= \frac{(x-1)\cdot(x-3)}{2-3} = \frac{(x^2 - 4\cdot x + 3)}{-1} && (86)\\
l_2(x) &= \frac{(x-1)\cdot(x-2)}{3-2} = \frac{(x^2 - 3\cdot x + 2)}{1} && (87)\\
L(x) &= \sum_{j=0}^{2} y_j l_j(x) && (88)\\
L(x) &= y_0 \cdot l_0(x) + y_1 \cdot l_1(x) + y_2 \cdot l_2(x) && (89)\\
&= (x^2 - 3\cdot x + 2)\cdot 4 - (x^2 - 4\cdot x + 3)\cdot 7 - \frac{(x^2 - 5\cdot x + 6)}{2}\cdot 2 && (90)\\
&= 4\cdot x^2 - 12\cdot x + 8 - 7\cdot x^2 - 28\cdot x + 21 - x^2 - 5\cdot x + 6 && (91)\\
&= 4\cdot x^2 - 7\cdot x^2 - x^2 + (-12\cdot x - 28\cdot x - 5\cdot x) + (+8 + 21 + 6) && (92)\\
&= -4\cdot x^2 - 45\cdot x + 35 && (93)\\
& && (94)
\end{aligned}
$$

ausprobieren, ob das funktioniert.


$f(x)$
x

tioniert, wie man sieht. man koennte noch werte einsetzen.

$$-4 \cdot (1) - 45 + 35 = -11$$

das stimmt tatsächlich nicht, dann ist rechenfehler drin, die aufgabe aber für heute beendet. nun kommt. ein m8 programm, für pap, später wo anders weiter. und: was kommt noch? ach ja, mit der euler tour, tiefen und breitensuche reicht noch nicht.

```
;; (c) david vajda
;; 10/22/2026
;; m8/atmega8/led count

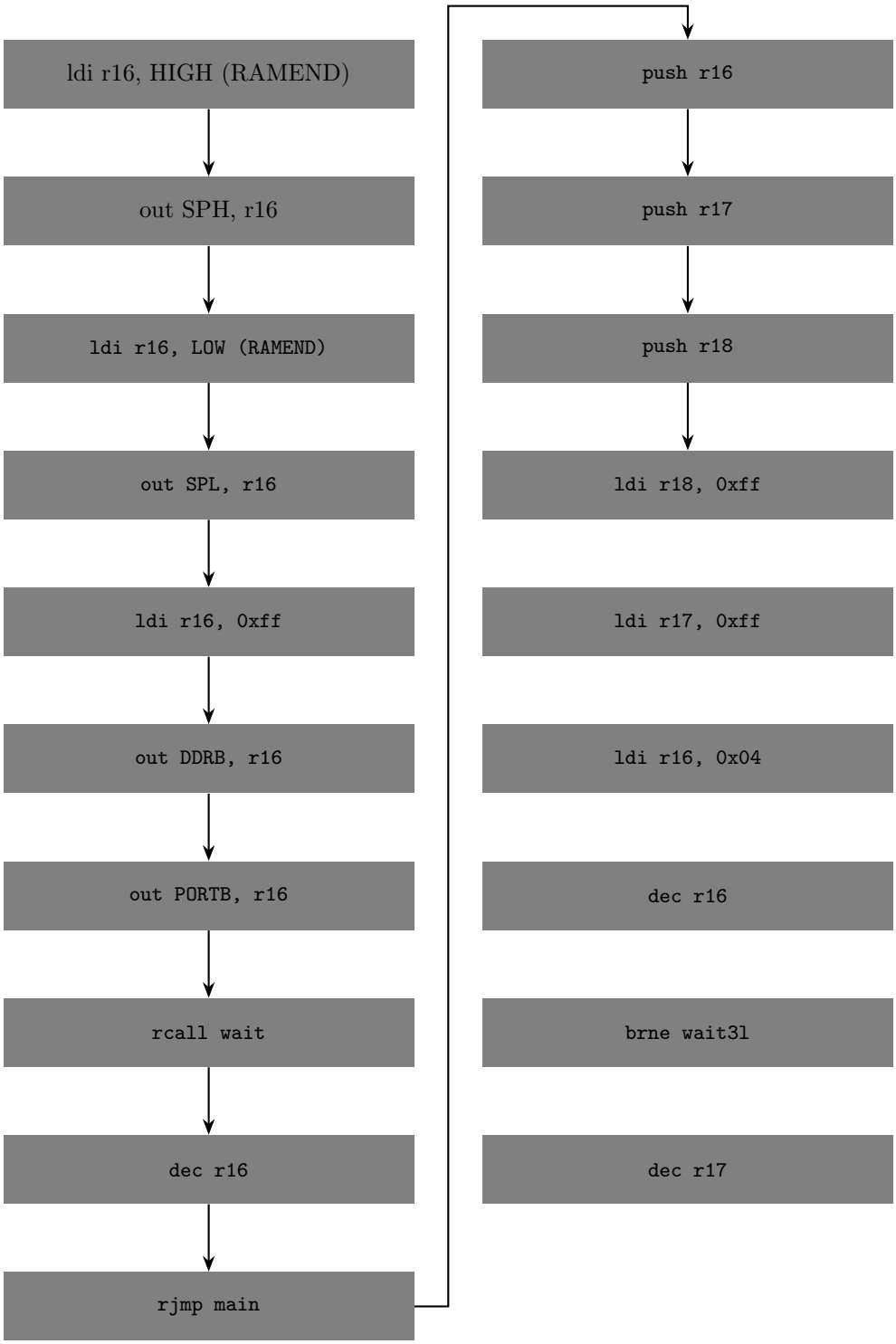
.include "m8def.inc"

ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16

ldi r16, 0xff
out DDRB, r16

main:
out PORTB, r16
rcall wait
dec r16
rjmp main

wait:
push r16
push r17
push r18
ldi r18, 0xff
wait1l:
ldi r17, 0xff
wait2l:
ldi r16, 0x04
wait3l:
dec r16
brne wait3l
dec r17
brne wait2l
dec r18
brne wait1l
pop r18
pop r17
pop r16
ret
```


ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16
ldi r16, 0xff
out DDRB, r16
out PORTB, r16
rcall wait
dec r16
rjmp main
push r16
push r17
push r18
ldi r18, 0xff
ldi r17, 0xff
ldi r16, 0x04
dec r16
brne wait3l
dec r17

```
dec r18
```

```
brne wait1l
```

```
pop r18
```

```
pop r17
```

```
pop r16
```

```
ret
```

ich habe nachgeguckt/oder weiss ich auswendig:

- Anweisung (engl. statement)
- engl. - heisst: Echo November Golf Lima dot
- pt - point, dot, pixel.
- simple statement, condition
- bedingte verzweigung
- unbedingte verzweigung
- sprung - das wort jump wird oft anders benutzt
- conditional statement
- Die Standard-Elemente im Programmablaufplan (PAP)
- Ein klassischer PAP nach DIN 66001
- Programmablaufplan (PAP) (auch Flussdiagramm genannt) oder ein Zustandsdiagramm (ASM-Chart / State Machine Diagram)
- ok, das war eine etwas falschere version, die geht so, vom pfeil her, richtig heisst sie so, danach, hier der quelltext

[13]Methode, lineare kongruenz , methode, Additive kongruenz

Ich glaube, Letzteres ist der Zusammenhang. Es gibt ja auch Grafen mit Zusammenhang. Also es gibt Grafen, also Grafen, die nicht Zusammenhngen, Teil Grafen und Grafen, die Zusammenhngen und bei den Zufallszahlen wissen wir ja, die hngen als Folge zusammen und kungruenzb Vielleicht sagt es was ber die Additive, das ist tatschlich, sagte ich ja, bei der Hasch Stable ist ja sag ich mal. Bei der Hashstbe wird es gleich gemacht nur nicht +1, sondern plus dem alten Wert. Also, es wird zwei reichen, dazu addiert wie beim Betonmixer. Vielleicht kann ich mir vorstellen, dass die Additive genau das macht, was die Hash Stable macht oder zumindest eine alte Reihe nimmt Anfalls zahlen und dann eine zweite dazu und dann hab ich die Additive und Kongruenz heit, wie gesagt wie auch beim Grafen einfach Zusammenhang congruere Lateinisch Zusammenhngen So, und das ist dann hier das gleiche weil bei der Folge hier hngen die Zahlen ja zusammen ungefhr wie beim Grafen Additiv knnte dann eben das heien bei der Hash Blle ist ja die gleiche Reihe und dann passt es okay und die geht so aus dem Kopf

$$a_i = (a_{i-1} \cdot b + 1) mod m \tag{95}$$
$$a_i = (a_{i-1} \cdot b + x_i) mod m \tag{96}$$
$$\tag{97}$$

Bei der Hash Tabelle gilt

```
\begin{tikzpicture}[
    node distance=1.5cm,
    stmt/.style={rectangle, draw, minimum width=6cm, minimum height=1cm, text centered, fi
    arrow/.style={thick, -{Stealth}}
]

\node [rectangle,minimum width=6cm, minimum height=1cm, fill=gray] (stmt1) at (2,50) {ldi
\node [rectangle,minimum width=6cm, minimum height=1cm, fill=gray] (stmt2) at (2,48) {out

\draw [arrow] (stmt1) -- (stmt2);

\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt3) at (2,46) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt4) at (2,44) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt5) at (2,42) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt6) at (2,40) {\sm

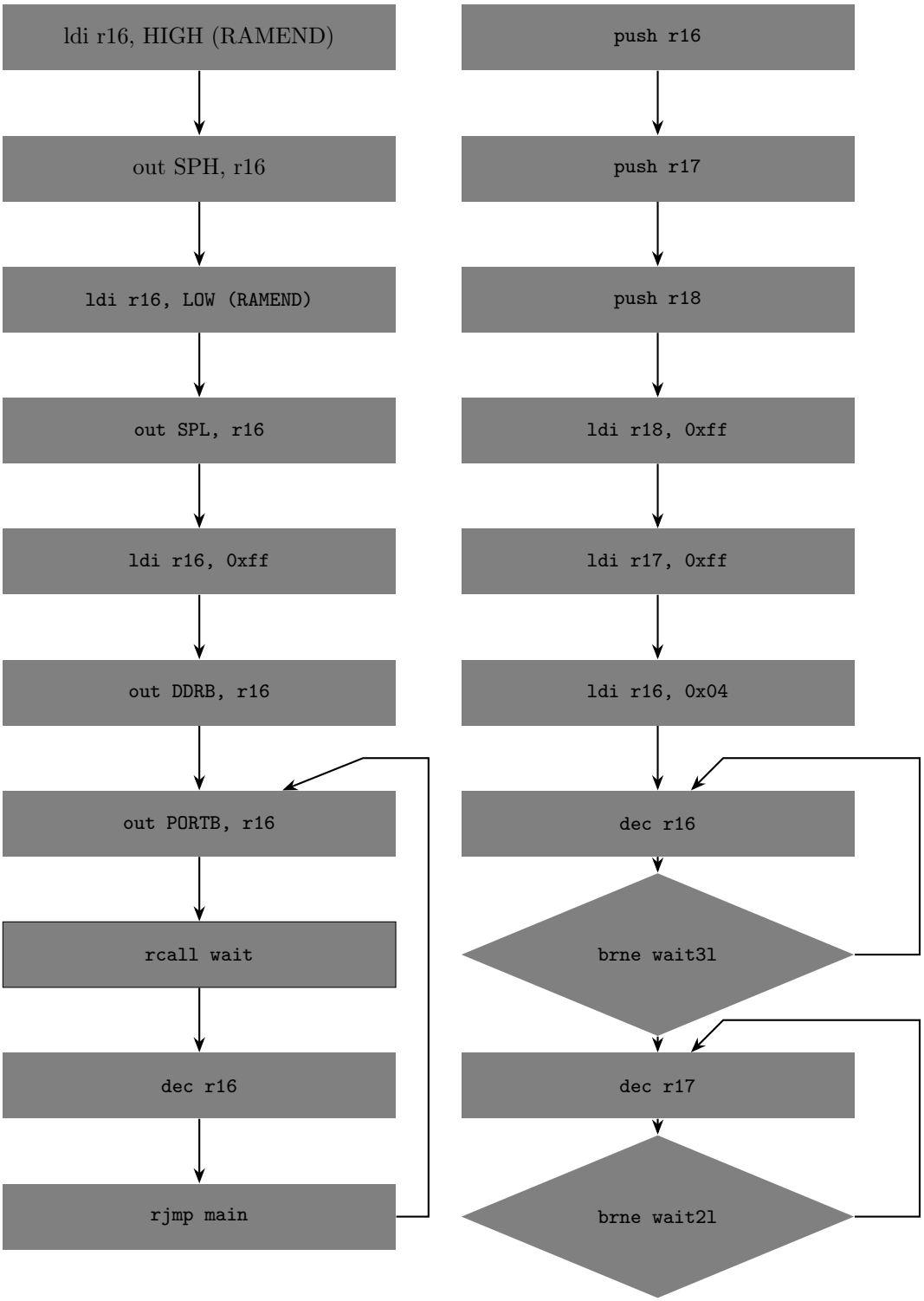
% main:
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt7) at (2,38) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt8) at (2,36) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt9) at (2,34) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt10) at (2,32) {\s

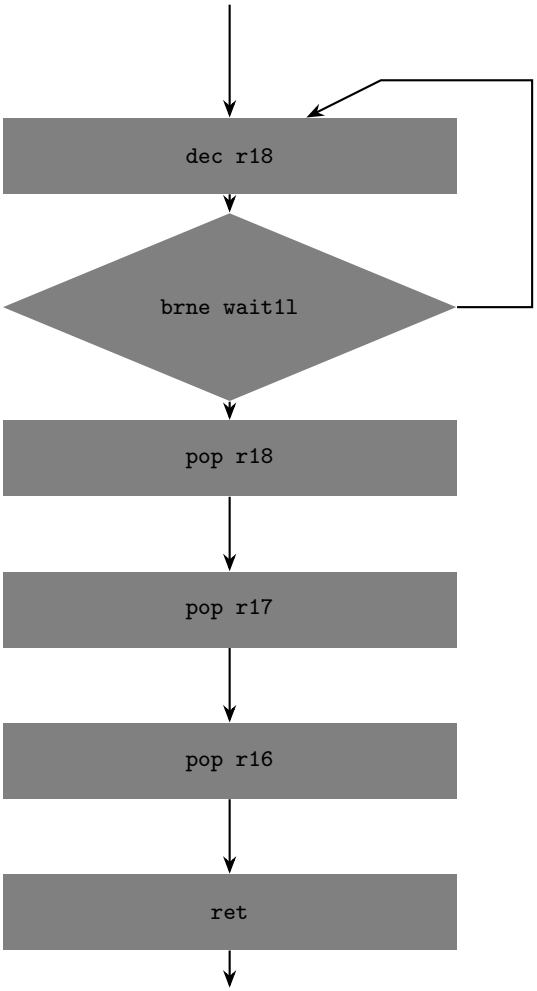
% wait:
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt11) at (9,50) {\s
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt12) at (9,48) {\s
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt13) at (9,46) {\s
```

```
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt14) at (9,44) {\sm
% wait1l:
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt15) at (9,42) {\sm
% wait2l:
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt16) at (9,40) {\sm
% wait3l:
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt17) at (9,38) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt18) at (9,36) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt19) at (9,34) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt20) at (9,12) {\sm

\draw [arrow] (stmt2) -- (stmt3);
\draw [arrow] (stmt3) -- (stmt4);
\draw [arrow] (stmt4) -- (stmt5);
\draw [arrow] (stmt5) -- (stmt6);
\draw [arrow] (stmt6) -- (stmt7);
\draw [arrow] (stmt7) -- (stmt8);
\draw [arrow] (stmt8) -- (stmt9);
\draw [arrow] (stmt9) -- (stmt10);
\draw [arrow] (stmt10) -- (5.5,32) -- (5.5,51) -- (9,51) -- (stmt11);
\draw [arrow] (stmt11) -- (stmt12);
\draw [arrow] (stmt12) -- (stmt13);
\draw [arrow] (stmt13) -- (stmt14);
\end{tikzpicture}

\begin{tikzpicture}
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt21) at (2,10) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt22) at (2,8) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt23) at (2,6) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt24) at (2,4) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt25) at (2,2) {\sm
\node [rectangle, minimum width=6cm, minimum height=1cm, fill=gray] (stmt26) at (2,0) {\sm
\end{tikzpicture}
```


ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16
ldi r16, 0xff
out DDRB, r16
out PORTB, r16
rcall wait
dec r16
rjmp main
push r16
push r17
push r18
ldi r18, 0xff
ldi r17, 0xff
ldi r16, 0x04
dec r16
brne wait3l
dec r17
brne wait2l



- Im Standard-Flussdiagramm (DIN 66001 / ISO 5807) nutzt man das Symbol fr einen
- Vordefinierten Prozess (Predefined Process).

[14]wie man den predefined process als knoten macht, weiss ich nicht, es gehört ein doppelstrich auf die linke und rechte seite, hier in tikz keine ahnung

1.1 nasm/i586/amd64/elf64

```
;; (c) david vajda
;; 10/22/2026
;; nasm i586/amd64

global _start
section     .data
    msg001: db  "david@https://www.vajda-breadboard.de", 10, 13, 0
section     .text
_start:
```

```
    mov rsi, msg001
    mov rcx, 0x00
l1: mov ah, [rsi]
    cmp ah, 0x00
    je l1end
    inc rcx
    inc rsi
    jmp l1
l1end:
    mov rax, 0x04
    mov rbx, 0x01
    mov rdx, rcx
    mov rcx, msg001
    int 0x80

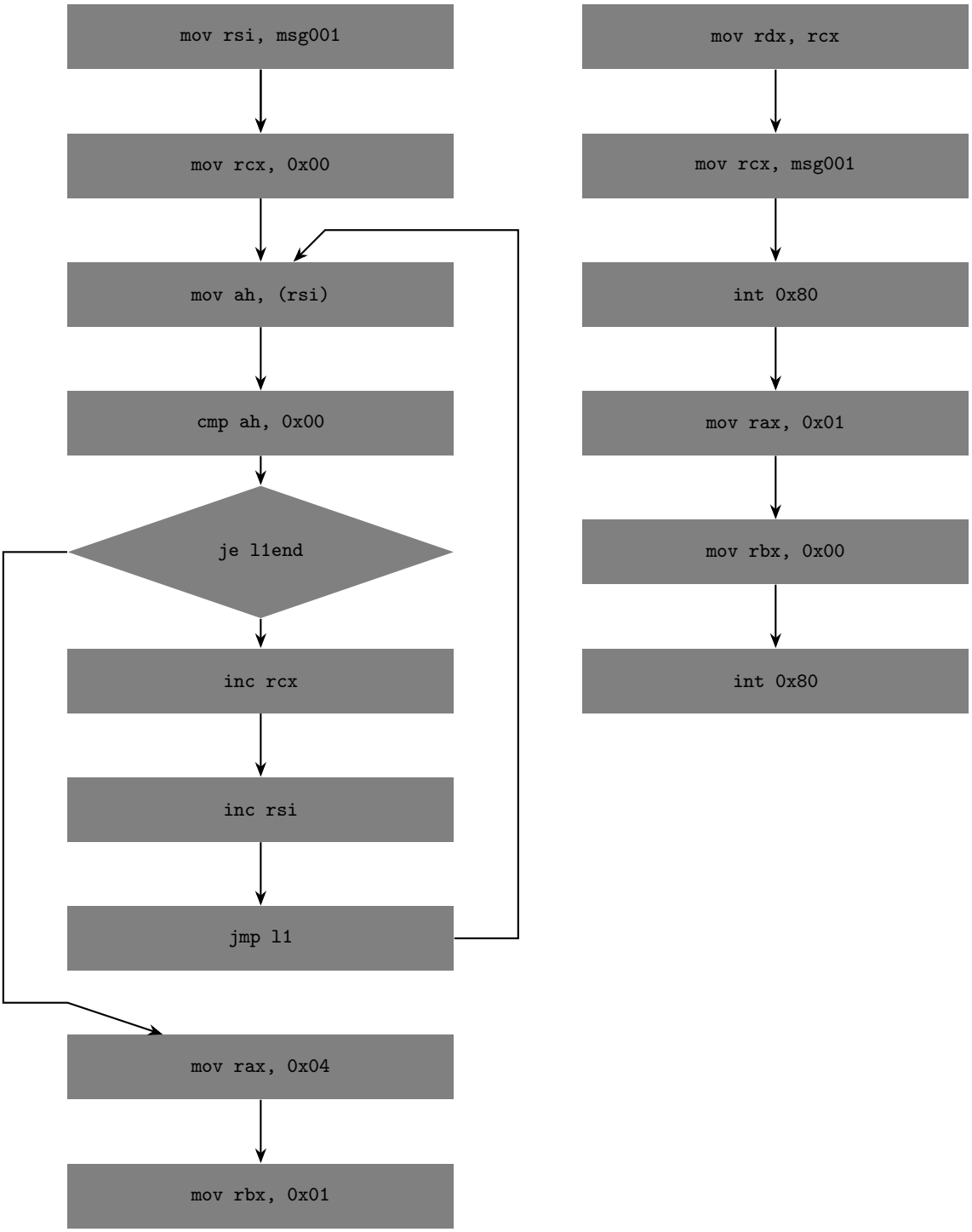
    mov rax, 0x01
    mov rbx, 0x00
    int 0x80

;; david@vajda-breadboard-0000:~$ nasm -f elf64 asm10222026.asm
;; david@vajda-breadboard-0000:~$ ld -o asm10222026 asm10222026.o
;; david@vajda-breadboard-0000:~$ ./asm10222026
;; david@https://www.vajda-breadboard.de
;; david@vajda-breadboard-0000:~$
```

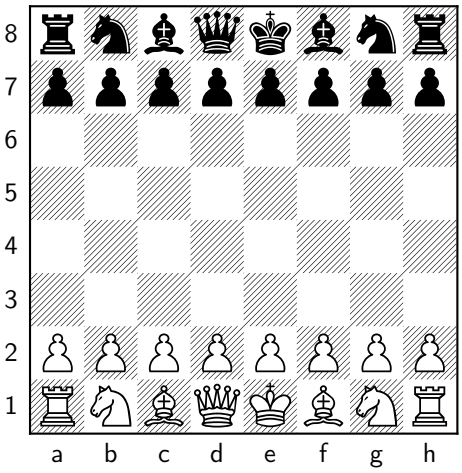
ergibt:

```
david@vajda-breadboard-0000:~$ nasm -f elf64 asm10222026.asm
david@vajda-breadboard-0000:~$ ld -o asm10222026 asm10222026.o
david@vajda-breadboard-0000:~$ ./asm10222026
david@https://www.vajda-breadboard.de
david@vajda-breadboard-0000:~$
```

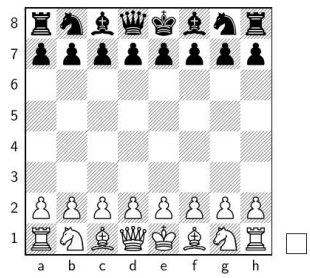
daraus nun ein pap


mov rsi, msg001
mov rcx, 0x00
mov ah, (rsi)
cmp ah, 0x00
je l1end
inc rcx
inc rsi
jmp l1
mov rax, 0x04
mov rbx, 0x01
mov rdx, rcx
mov rcx, msg001
int 0x80
mov rax, 0x01
mov rbx, 0x00
int 0x80

2 schach und der automat



2 schach und der automat



alright geht mit latex, so:

```
...
\usepackage{xskak}
\usepackage{chessboard}
\begin{document}
\newgame
\chessboard
...
```

ok:

[0]Vorankündigung, ich bin so frech. Ich hab jetzt verstanden, was **P** und **NP** Vollständigkeit ist.

Beziehungsweise Vollständigkeit ist **NP**Vollständigkeit noch mal. Also der deterministischen Automat ist der, bei dem keine Entscheidung getroffen wird, wenn die Daten durch die Leitung rs 232/v.24 die ganze Zeit durchgehen und da passiert nix weiter dann ist es deterministisch

P und **NP** Problematik bezieht sich auf die Laufzeit. Beides lässt sich als wir haben, zum Beispiel folgendes.

Ein normaler, also ein Nichte deterministisch Automat muss Entscheidung treffen. Zum Beispiel bei dem pattern Matching muss er Entscheidung treffen.

Okay, und wenn man jetzt den KMP anguckt

Also wir gehen zu dem trivialen also sozusagen groben über und der hat der muss ja der hat die Laufzeit M für p für das Muster und N für a den Text wenn ich bei jedem sozusagen Vergleich wieder an den Anfang zurück muss. Aber im schlimmsten Fall geh ich am Ende dann hab ich MN

alright geht mit latex, so:

```
...
\usepackage{xskak}
\usepackage{chessboard}
\begin{document}
\newgame
\chessboard
...
```

[15]Voranknd igung, ich bin so frech. Ich hab jetzt verstanden, was **P** und **NP**

Vollstndigkeit ist.

Beziehungsweise Vollstndigkeit ist **NP**Vollstndigkeit noch mal. Also der deterministischen Automat ist der, bei dem keine Entscheidung getroffen wird, wenn die Daten durch die Leitung rs 232/v.24 die ganze Zeit durchgehen und da passiert nix weiter dann ist es deterministisch

P und **NP** Problematik bezieht sich auf die Laufzeit. Beides lsst sich als wir haben, zum Beispiel folgendes.

Ein normaler, also ein Nichte deterministisch Automat muss Entscheidung treffen. Zum Beispiel bei dem pattern Matching muss er Entscheidung treffen.

Okay, und wenn man jetzt den KMP anguckt

Also wir gehen zu dem trivialen also sozusagen groben ber und der hat der muss ja der hat die Laufzeit M fr p fr das Muster und N fr a den Text wenn ich bei jedem sozusagen Vergleich wieder an den Anfang zurck muss. Aber im schlimmsten Fall geh ich am Ende dann hab ich MN

Also wenn ich fr jedes Ding einfach zurck muss, dann muss ich sozusagen das ist dann aber auch nicht deterministisch weil ich kann nicht 100 pro entscheiden, dass der Automat einfach durchluft

P und **NP** heit brigens, dazu gehren alle Grafen Euler Tour zum Beispiel aber bei Hamilton Kreisen. Also Euler Tour selber hat er nicht automatisch den Hamilton Kreis. Der Eule hat ja erst mal bewiesen, dass es sozusagen die Eulert Tour machbar ist. Okay Hamilton Kreis ist zum Beispiel ein Beispiel. Es geht aber um die Frage Polynomial oder noch mehr als Polynomial.

Jetzt hab ich ne Idee es gibt bei Latex ja auch Schachfiguren. Wie ist es eigentlich mit dem Schachspiel?

Ich hab ne Idee also gucken Sie mal wir gehen einfach pro Schachfiguren. Wir haben 16 Schachfiguren und wir haben Latex mit Latex kann man sich einen schnen Schach zeichnen da es schne Automaten sind hab ich festgestellt

Folgendes, das sind jeweils ein Pfad, also ein gerader Automat das knnte man sagen Fasten deterministischen Automat, wenn man den sozusagen die Bauern anguckt

Erst mal die Frage ist wie viel Figuren haben wir ich selber hab 16. anderer hat 16. sind 32

Dabei kann ich Entscheidung treffen dann muss ich pro Figur zum Beispiel nehmen wir den Lufer sagen wir mal grob das stimmt nicht. Ganz kann ich sozusagen in vier Richtungen gehen.
Ich kann dabei so so viele Felder laufen also mache ich zum Beispiel Vierer Party Richtung die Figur und so weiter jetzt hab ich aber 16 Figuren. Also hab ich neautomat bei mir der allein 16 Entscheidung treffen kann und beim Gegner 16. Ich muss aber den Gegner bei mir so einrechnen wir bei mir selber. Ich wei zwar nicht was er macht, aber das sind 32 Figuren. Da muss ich mit dem Rechnen gut okay

Wenn er das hat, dann ist es soweit und das ist sozusagen automatisch 32 sozusagen Entscheidung trifft also nicht deterministisch sind es 32 jetzt kommen aber noch Zge, die ich nicht machen kann dann wrde ich ne dritte Menge nehmen. Die dritte Menge wrde ich halt mit berschneidung, Schnittmenge und sozusagen Vereinigung Menge wieder abziehen die wrde ich rausnehmen. Trotzdem hab ich 32 Figuren wenn ich meine Bauern wiederum angucke. Die knnen ja alle gerade ausgehen. Auer es steht ne gegnerische Figur an der Seite. Jetzt hab ich noch Latex damit kann man Schachbretter, dann hab ich ein schnes Automat. Da geht Bauer immer geradeaus.

Das ist quasi deterministischen Automat. Auer es steht ne Figur an der Seite. Wenn die Figur an der Seite ist, dann kann er sich entscheiden schrg zu schlagen dann gibt es klein bisschen Entscheidung also deterministischen sind acht Zge nach vorne sind nur sieben, weil er steht ja eins vorne rein theoretisch sieben und das ist deterministisch mit ein bisschen Einschrnkung, dass sie zur Seite gehen kann sozusagen schrg sogar vielleicht in zwei okay solche Sachen das kann man ja morgen ausprobieren mit dem Schachtel und dann hat man auch kann man das auch als normalen Automat darstellen fertig danke schn jetzt gehe ich in die Stadt. ok:

2.1 deterministischer Automat/nicht deterministischer Automat

- deterministischer Automat
- nicht-deterministischer Automat

so ungefähr, ausser NP-Vollständigkeit

- **P**: heisst: Polynomiale Laufzeit, also: n bei einer sequentiellen Eingabe von n Zeichen, im Automat

- **NP**: Mehr als polynomiale Laufzeit, also $n \cdot m$ bei einem nicht deterministischen Automat, der entscheidungen trifft, zwei eingaben, zum Beispiel ein Text und ein Muster anhand dessen Entschieden wird, Text von der Länge n und Muster von der Länge m
- deterministischer Automat: hier wird keine entscheidung getroffen, der Graph, automat läuf sequentiell linear einfach durch
- nicht-deterministischer Automat: hier werden entscheidungen getroffen, im Graph

Graphen

- Graph P_n: Pfad. P steht für Path.
- andere Graphen, wie Bäume und mehrere Cycles C_n - Eulertour: Heisst Hamilton Kreise, hier werden entscheidungen getroffen.

Beim Groben Algorithmus zum Finden von Mustern in Zeichenfolgen, müsste ich im schlimmsten falle davon ausgehen, ich gehe im Text immer ein Zeichen vorwärts und vergleiche jedes mal fast das ganze muster bis zum Ende, also $m \times n$

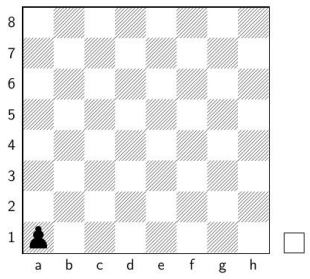
trotzdem bezieht sich das klassischerweise auf automaten. Kombinatorik, wie ist es beim Schach.

1. erste Idee: 8 Bauern, könnten, ein Schachfeld hat 8×8, Felder sind 64. schwarz/weiss, können, so wie sie stehen, in einer zeile/reihe ohne hindernisse, jeweils 7 schritte gehen, wenn sie nicht schlagen, nach vorne, denn sie stehen vom spieler aus, in der 2. Reihe/Zeile
2. an und für sich ist ein Bauer, der immer nur einen Schritt nach vorne geht, bis zum Ende, ein nahe zu deterministischer automat, unter dieser Bedingung. denn
   - ohne, dass im Norden, vom Spieler aus, Nord-West, oder Nord-Ost, Distanz ein Feld, ein Gegner steht, geht der Bauer immer gerade aus
   - Feld für Feld
   - Ergibt einen Pfad P_7 mit 7 Knoten, pro Bauer
3. als gesamtes gibt das einen Automaten, der nicht deterministisch ist, er kann pro zug, angenommen, nur bauern stehen auf dem feld und: sie können unabhängig von einander jeweils pro zug einen schritt nach vorne gehen: ein automat, mit 8 entscheidungen pro zug. da jeder der Bauern einen schritt nach norden bewegt werden kann.
4. sollte sich ergeben, dass vor ihm, ohne, dass in diesem modell das einen einfluss hat, eine gegnerische figur stehen, liesse sich pro bauer ein nicht deterministischer automat . . . : weil: jetzt kann er 3 entscheidungen pro zu treffen. eines nach norden, eines nach nord-westen oder eines nach nordosten + 8 Bauern
5. abgesehen davon, wenn der erste Bauer das ende erreicht hat. und trotzdem: pro bauer, sind es 7 züge, damit $8 \cdot 7$

6. wie gross ist die gesamtzahl einschätzungsweise aller figuren pro spiel mit n getanen zügen für alle 32 Figuren auf dem Schachbrett mit 64 Feldern?

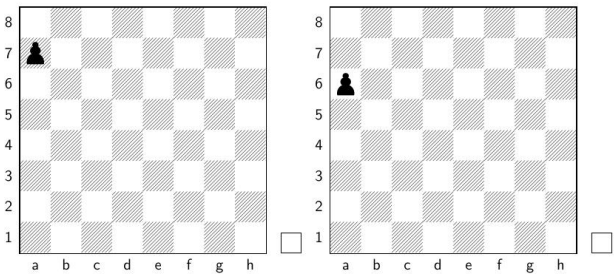
   - könnte eine Figur, auf allen 64 Feldern stehen, sind das 64 Felder
   - nehmen wir an, alle anderen könnten auch überall stehen, sogar da, wo die anderen stehen. Bei 32 Figuren sind es 64^{32}
   - unter der vorraussetzung, dass eine figur nicht da stehen kann, wo die anderen stehen, sind es nahezu nicht 64^{32}? sondern $64 \cdot 63 \cdot 62 \cdot \cdots \cdot 32$ ällerdings! es sind nicht 64!, sondern $\binom{64}{2}$, also 64 über 2. 64 chr 2. weil: ich ja nur die ersten 32 figuren verteilen kann.

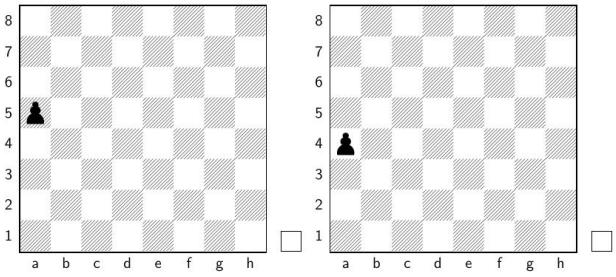
nun wollen wir uns einen solchen automaten mit hilfe des schachbretts vorstellen.

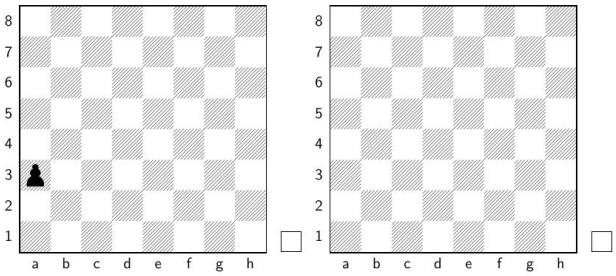


60

mit einem einzigen bauern. für latex2html


8 7 6 5 4 3 2 1
a b c d e f g h
8 7 6 5 4 3 2 1
a b c d e f g h


8 7 6 5 4 3 2 1
a b c d e f g h
8 7 6 5 4 3 2 1
a b c d e f g h


8 7 6 5 4 3 2 1
a b c d e f g h
8 7 6 5 4 3 2 1
a b c d e f g h

59

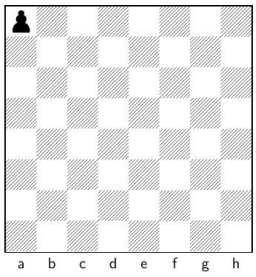
4. sollte sich ergeben, dass vor ihm, ohne, dass in diesem modell das einen einfluss hat, eine gegnerische figur stehen, liesse sich pro bauer ein nicht deterministischer automat . . . : weil: jetzt kann er 3 entscheidungen pro zu treffen. eines nach norden, eines nach nord-westen oder eines nach nord-osten + 8 Bauern

5. abgesehen davon, wenn der erste Bauer das ende erreicht hat. und trotzdem: pro bauer, sind es 7 züge, damit $8 \cdot 7$

6. wie gross ist die gesamtzahl einschätzungsweise aller figuren pro spiel mit n getanen zügen für alle 32 Figuren auf dem Schachbrett mit 64 Feldern?

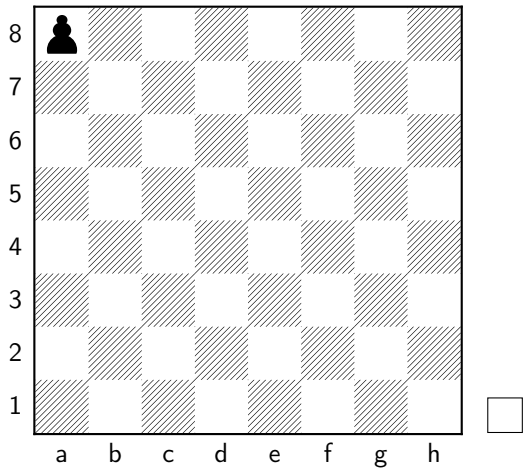
   - könnte eine Figur, auf allen 64 Feldern stehen, sind das 64 Felder
   - nehmen wir an, alle anderen könnten auch überall stehen, sogar da, wo die anderen stehen. Bei 32 Figuren sind es 64^{32}
   - unter der vorraussetzung, dass eine figur nicht da stehen kann, wo die anderen stehen, sind es nahezu nicht 64^{32}? sondern $64 \cdot 63 \cdot 62 \cdot \cdots \cdot 32$ ällerdings! es sind nicht 64!, sondern $\binom{64}{2}$, also 64 über 2. 64 chr 2. weil: ich ja nur die ersten 32 figuren verteilen kann.

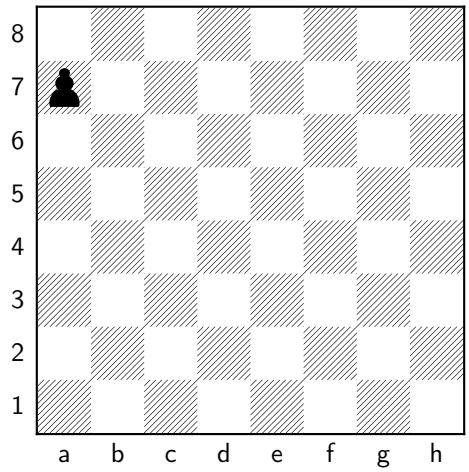
nun wollen wir uns einen solchen automaten mit hilfe des schachbretts vorstellen.

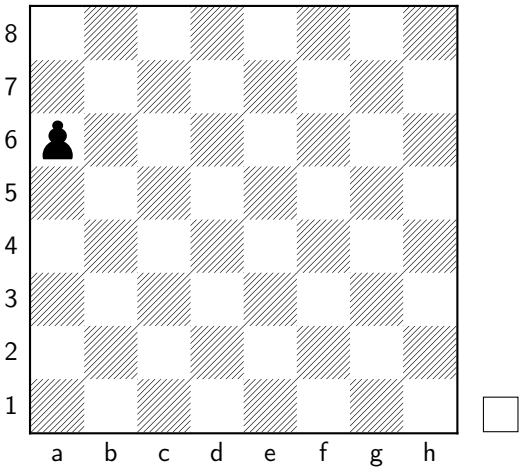


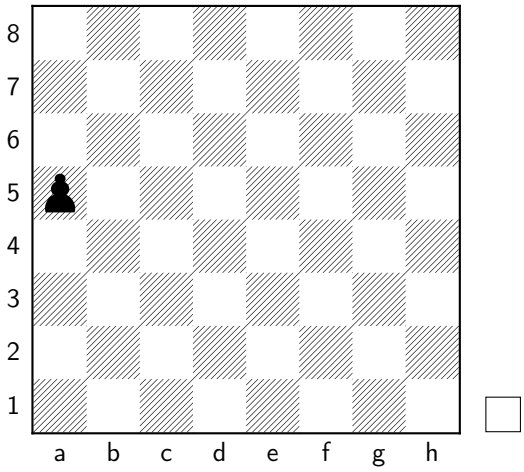
mit einem einzigen bauern. **1 d2 d3 2 e7 e6** □

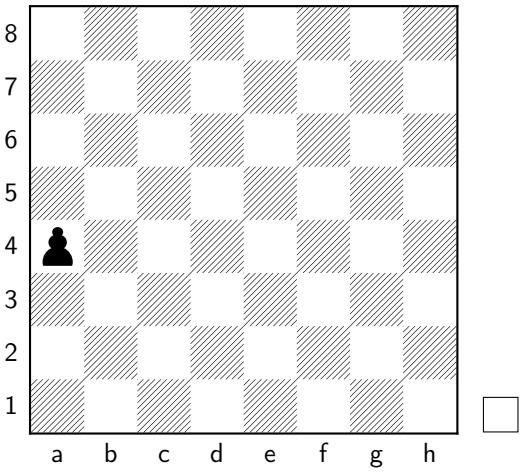
1 d2 d3 2 e7 e6

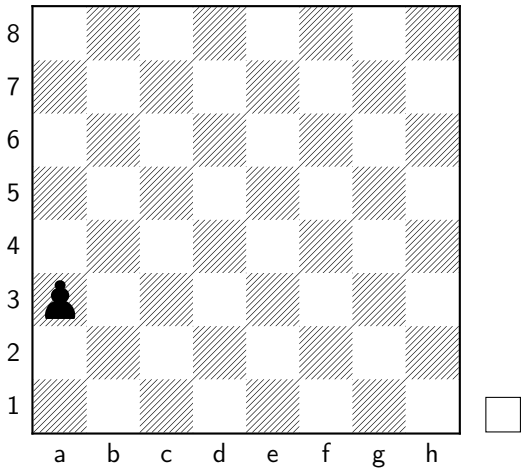


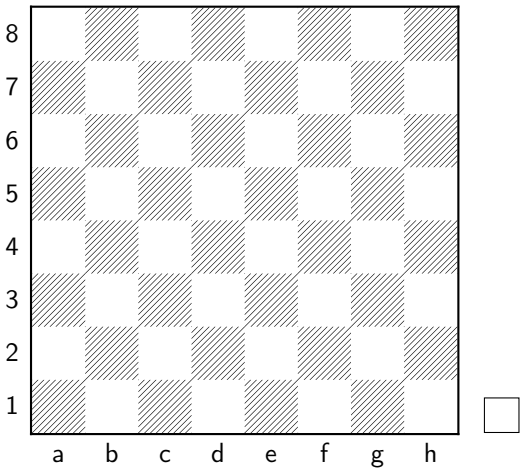


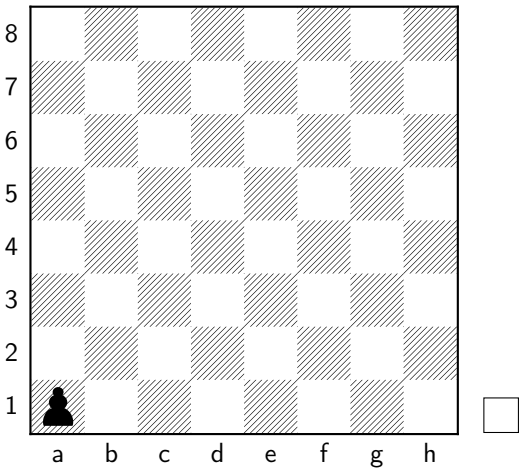









8
7
6
5
4
3
2
1
a b c d e f g h

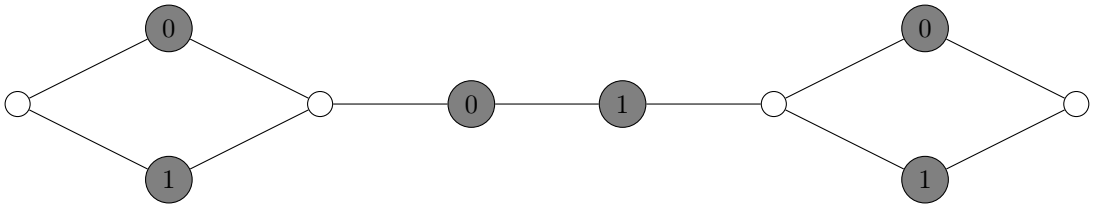

8
7
6
5
4
3
2
1
a b c d e f g h

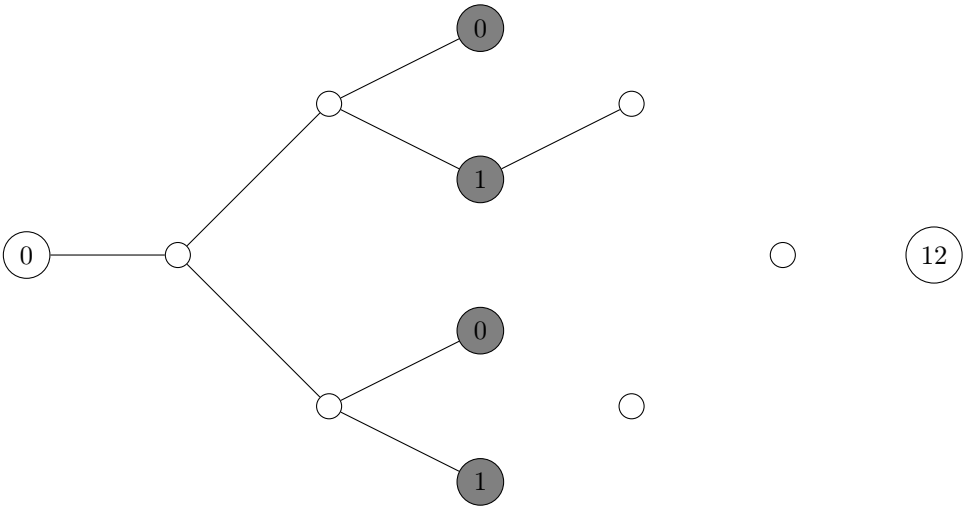


3 KMP und Pattern Matching

```
(2,10,3,11)  x 0 1 x
(0,4,2,6)   0 x x 0
(0,2,1,3)   0 0 x x
(11,15)   1 x 1 1
(13,15)   1 1 x 1

 Y = /X2 * X1 +
   /X3 * /X0 +
   /X3 * /X2 +
   X3 * X1 * X0 +
   X3 * X2 * X0
```





ok, machen wir später weiter, ein wenig KMP, wir nehmen das y-Wort, bzw. in der Senkrechte nehmen wir.

```
(C) David Vajda
Wed Oct 21 19:58:16 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    1
 5 0 1 0 1    0
 6 0 1 1 0    1
 7 0 1 1 1    0
 8 1 0 0 0    0
 9 1 0 0 1    0
10 1 0 1 0    1
11 1 0 1 1    1
12 1 1 0 0    0
```

```
13 1 1 0 1    1
14 1 1 1 0    0
15 1 1 1 1    1

0b1010.1100.0101.1111
```

ok, nun knuth-morris-pratt, der algorithmus

| | | |

sondern:

```
(A*B+AC)D
```

bei arithmetisch:

```
<expr> ::= <term> | <term> '+' <expr>
<term> ::= <fact> | <fact> '*' <term>
<fact> ::= '('<expr>')' | v
```

3.0.1 pattern matching ausdrücke

drücken wir per zufall das per pattern matching aus

```
(A*B + AC)D
(AB)*
(AB*)*
((A+B)D)*C*
```

ergibt, nachher der KMP, mit richtiger belegung und vergleich nicht KMP.

```
(A*B + AC)D  z.B.= AAABD, AAAAAAABD, ACD, ABD, etc.
(AB)*  z.B.=AB, ABAB, ABABAB, etc.
(AB*)*  z.B.=AB, ABB, ABBB, ABAB, ABBABB, ABBBABBB, ABBBABBBABBB, etc.
((A+B)D)*C* z.B.= ADC, BDC, ADADADCCCC, ADBDADADADBDBDCCCC, BDBDC, BDADCCCC, etc.

marx: oder= (alt) ..
aber eigentlich untermenge, weil schreibweise:

(A*B + AC)D  z.B.= {AAABD, AAAAAAABD, ACD, ABD}
(AB)*  z.B.={AB, ABAB, ABABAB}
(AB*)*  z.B.={AB, ABB, ABBB, ABAB, ABBABB, ABBBABBB, ABBBABBBABBB}
((A+B)D)*C* z.B.=

und
{AB, ABB, ABBB, ABAB, ABBABB, ABBBABBB, ABBBABBBABBB} SUBSET (AB*)*
gleich im mathe modus
```

[16]beim marx heisst es *oder* = und *usw.Ware* und *etc.* kommt am ende der Aufzählung, so weit ich weiss, Marx, jedenfalls, wahrscheinlich früher auch heute korrekt nur nicht immer - jedenfalls - . . . - Kapital, Bd. 1 . . . 3, anwesend, jetzt kein blick

so, jetzt der algorithmus von KMP, jetzt aber mit normalem durchsuchen des musters.
noch mal die algorithmen. von der idee her: der KMP macht im vorlauf das durchsuchen des musters von anfang an wie sonst den text. das heisst, das muster wird im muster selbst durchsucht. fälschlicher weise, heisst das auf den anfang bezogen.

- also ich suche das muster nicht im text, sondern im muster.
- das wiederum heisst, wenn ich das muster im text suche, fange ich am anfang des musters an

- jetzt die erweiterung: während ich das muster, im muster, nach der groben art suchen könnte, wie das muster, im text, nach der groben...
- geht KMP wieder einen schritt weiter: es durchsucht das muster, nach dem muster, nach KMP art - Knuth-Morris-Pratt ...
- ... zusatz: und zwar sogar nach der begründung, dass ja: die tabelle bereits am anfang gefüllt ist, das heisst, ein tabelle ist schon da.

im nächsten schritt machen den KMP von vorher: als grob, damit wir sehen, wenn es um das muster suchen im muster geht, die wiedersetzaufsetzen: dann: könnte man klassische grobe art machen. und also bei jedem fehlstart

```
i+1, j=0
```

im text wiederanfangen. und der zweite der noch verwirklicht zu werden hat: ist: in diese tabelle die ich gemacht habe. das sind jetzt zwei tabellen in einer

1. das muster selber, muster unter muster
2. es geht noch weiter: ich muss

   ```
   i, j, next [j]
   ```

   usw. initialisieren, wie bei tabellen, was das hashing betrifft und zufallszahlen und so weiter - handelt es sich bei funktionen/methoden gleichzeitig um tabellen. wenn sie

   - den wert aus dem alten wert berechnen (Zufallszahlen)
   - oder verschiedene neue argumente einsetzen $f(x)$
   - oder beides Hashtable (Argument + altes, ist ja zufallszahl) - additive Zufallszahl - hier muss ich bei Robert Sedgewick nachgucken, das buch liegt hier. oder:
     Zu bedenken, wahrscheinlichkeitsrechnung

     ```
     X[i];
     ```

     entspricht bei Hashtable $P(X = x_i)$ Wahrscheinlichkeit $x_i = X(e_i)$, entspricht Zufallsvariable. das heisst, wenn auch bei Hashtable Agumente bei i trotzdem $X(i)$ so zu sagen.

[17]jetzt muss ich blos putzen und weg. jetzt bin ich wieder da. ich habe überlegt. der rest gehört in die fussnoten.
[18]fussnote = anmerkung so: ich muss machen:

- das übliche quine [19]die fussnote - hier rein... zurück zur sache: quine mit DNF - und ordentlich, mit [20]man hat gerade wieder schwierigkeiten was zu sagen, kommt nachher alles in die fussnoten - alles [21]ich muss bei mir selber sein, ob ich andere ärgere oder nicht - das hängt mit rechnen nicht zusammen, geärgert müssen mit schrei oder ohne, die damit klar kommen, melde mich ab vom dienst - der menschlichkeit - die vollziehbar ist, aber auch nicht - was mit rechnen nichts zu tun hat, nachher viele anmerkung - ein starker wind - nachher viele anmerkungen - melde mich vollständig ab vom dienst mit galasm und: des weiteren: circuitikz. die klassische aufgabe, zum normalen programm wie immer

- dazu ein bisschen aber auch

  `atmega8`

  nämlich das heraufzählen, der eingangsvariablen x_3, x_2, x_1, x_0 mit einer zählschleife und externen interrupts, vielleicht timer interrupts

- dazu ein i586/nasm programm. aber vor allem dazu: die

  ```
  ax/eax/rax = 1, exit
  ax/eax/rax = 2, fork
  ax/eax/rax = 3, read
  ax/eax/rax = 4, write
  ax/eax/rax = 5, open
  ax/eax/rax = 6, create
  ```

  ein zu führen.

- der hauptschritt dies mal liegt aber wie immer beim rechnen [22]der aller haupt schritt, allerdings bei den graphen: eulertour - in wie weit tiefen und breitensuche funktioniert, weiss ich nicht, dies mal
  - matrizenrechnung - multiplikation
  - hash-table + fletcher's checksum +
  - zufallszahlen
  - integralrechnung - symbolisch
  - lagrange - interpolation von polynomen
- nächsten schritt, kommt die graphentheorie selber
  - graph malen
  - eulertour
  - wenn geht, tiefen und breitensuche
- wenn dies geschehen ist, kommen weitere graphen+automaten, als da wären
  - ich übe ausdrücke in der grammatik, ich schreibe möglichst viele hin, und probiere beispiele zu finden
  - die ordentliche sprache + schrift, mengen zum beispiel mache ich so gut wie möglich finde verschiedene formen
  - ich mache folgen von ziffern/zeichen - binär - 2 bit, 4 bit: (Aiken, möglich, dreiexcess, was auch immer) - text: und versuche daraus Grammatik zu machen
  - ich erstelle den automaten
  - ich nehme das KMP suche das muster im muster [23]das würde ich zu erst machen: KMP
  - ich nehme eine tabelle, für kmp üblich
  - ich nehme als vergleich den groben algo

– ich schreibe tabellen, einerseits nur funktion

– + gemischt: doppelt: die vom muster + die mit

  `next [j]`

  und so weiter

[24]ich fange jetzt an

[25]abgemeldet vom dienst der menschlichkeit - es kann immer zu geruckel kommen, turbulenzen - abgemeldet vom dienst der menschlichkeit, heisst, ich rechne trotzdem: active no - lift off - ready - 3, 2, 1 start - go - ok

noch mal die tabelle:

1	~~0~~	1	0	1	1	0	0	0	1	0	1	1	1	1	1											
	~~1~~	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1										
1	0	***1***	0	1	1	0	0	0	1	0	1	1	1	1	1											
		1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1									
1	0	***1***	***0***	1	1	0	0	0	1	0	1	1	1	1	1											
		1	***0***	1	0	1	1	0	0	0	1	0	1	1	1	1	1									
1	0	***1***	***0***	***1***	1	0	0	0	1	0	1	1	1	1	1											
		1	***0***	***1***	0	1	1	0	0	0	1	0	1	1	1	1	1									
1	0	***1***	***0***	***1***	~~1~~	0	0	0	1	0	1	1	1	1	1											
		1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1									
1	0	1	0	1	***1***	0	0	0	1	0	1	1	1	1	1											
					1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1						
1	0	1	0	1	***1***	***0***	0	0	1	0	1	1	1	1	1											
					1	***0***	1	0	1	1	0	0	0	1	0	1	1	1	1	1						
1	0	1	0	1	***1***	***0***	~~0~~	0	1	0	1	1	1	1	1											
					1	***0***	~~1~~	0	1	1	0	0	0	1	0	1	1	1	1	1						
1	0	1	0	1	1	0	~~0~~	0	1	0	1	1	1	1	1											
							~~1~~	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1				
1	0	1	0	1	1	0	0	~~0~~	1	0	1	1	1	1	1											
								~~1~~	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1			
1	0	1	0	1	1	0	0	0	***1***	0	1	1	1	1	1											
									1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1		
1	0	1	0	1	1	0	0	0	***1***	***0***	1	1	1	1	1											
									1	***0***	1	0	1	1	0	0	0	1	0	1	1	1	1	1		
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	1	1	1	1											
									1	***0***	***1***	0	1	1	0	0	0	1	0	1	1	1	1	1		
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	~~1~~	1	1	1											
									1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1		
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	~~1~~	1	1	1											
									1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1		
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	~~1~~	1	1	1											
									1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1		
1	0	1	0	1	1	0	0	0	1	0	1	***1***	1	1	1											
												1	0	1	0	1	1	0	0	0	1	0	1	1	1	
1	0	1	0	1	1	0	0	0	1	0	1	***1***	~~1~~	1	1											
												1	~~0~~	1	0	1	1	0	0	0	1	0	1	1	1	
1	0	1	0	1	1	0	0	0	1	0	1	1	***1***	1	1											
													1	0	1	0	1	1	0	0	0	1	0	1	1	
1	0	1	0	1	1	0	0	0	1	0	1	1	***1***	~~1~~	1											
													1	~~0~~	1	0	1	1	0	0	0	1	0	1	1	
1	0	1	0	1	1	0	0	0	1	0	1	1	1	***1***	1											
														1	0	1	0	1	1	0	0	0	1	0	1	
1	0	1	0	1	1	0	0	0	1	0	1	1	1	***1***	~~1~~											
														1	~~0~~	1	0	1	1	0	0	0	1	0	1	
1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	***1***											
															1	0	1	0	1	1	0	0	0	1	0	

also, ok und jetzt geht es mit tabelle -

```
next [j]
```

```
next [0], next [j], next [M]
```

jetzt denken wir zunächst an die schleife und bedingungen, ich schreibe dann die algorithmen beiderseits voll hin

```
while ((p [j] != p [i]) && (j > 0))
```

ausserdem

```
while ((p [j] != p [i]) && (j > 0)) j = next [j];
```

und oben

```
next [j] = j;
```

warum? zunächst:

- aller erstens: ich vergleiche das muster zunächst nicht mit dem text, sondern mit dem muster, das ist der witz, ich vergleiche das muster zunächst mit dem muster
- ausserdem: ich suche das muster im muster, heisst, trotzdem bezogen auf den anfang
- ausserdem: ich benutze dieselbe methode, wie im text bei KMP, das heisst, der vorlauf könnte passieren, wie bei dem normalen suchalgorithmus, um es dann beim wirklichen KMP an zu wenden, aber das ist so nicht, ich benutze zum suchen im muster, mit dem muster, wieder denselben wie beim suchen bei dem muster selber: begründung
  - für den suchalgorithmus im text brauche ich KMP, mit vorlauf
  - für den vorlauf nehme ich KMP, vorlauf bedeutet aber tabelle, und die tabelle ist ja so weit da, nämlich von anfang an.
- erstens: die innere schleife, heisst nicht, solange

  ```
  /*
  das heisst es nicht
   if (p[j] == p [i]) ...
  das heisst es nicht
   while (p[j] == p [i]) ...
  */
  ```

  sondern das gegenteil: solange es ungleich ist. das heisst, beim suchen im muster, des musters passiert dasselbe wie im text

  ```
  j = next [j]
  ```

  wird damit initialisiert, solange es nicht gleich nicht! obwohl das da steht ist das gerne ein psychologischer verwechslungsfehler! weil, man möchte nicht die innere schleife dem nicht gleich gleicht. gleichzeitig solange grösser 0. wenn das nicht der fall wird, aussen:

```
next [j] = j
```

also, das umgedrehte. doch das heisst 0. denn solange

```
while ((p [j] != p [i]) && (j >= 0)) j = next [j];
```

und

```
j >= 0
```

so, das war die klassische tabelle, jetzt kommt aber, erst der grobe algorithmus
nein, nun wird die tabelle um

```
next [j]
```

erweitert. bisher, war das die reine mustertabelle, dazu kommt jetzt: die tabelle wie in der mathematik üblich zusätzliche parameter.

[26]klassische tabelle - sie belieben zu scherzen herr vajda - zur begeisterung ihrer angeblichen sozialarbeiterin wie sie meinten - aber eigentlich betreuerin - frau buchner - genauer traue ich mich jetzt auch nicht zu reden und es mit der angst zu tun - verklemmung - dass das wort gesetzlich, womit ich den ganzen laden wieder in einen hühnerstall verwandelt habe ...- weil aber: damit die emotionskappe - oder wie soll sagen, abschluss, verschluss, dass das raus kommt, somit die emotionalität der anderen waltet, und freiheit erlaubt, womit sie mit den negativen - einschliesslich einweisung in die psychiatrie selber klar kommen müssen, auch, wenn alle gleichzeitig - müssen somit menschen probleme selber lösen - psychogramm während dem mathematik schreiben, je mehr es ruckelt - um so mehr ruckelt vajda - und: ...- psychogramm während mathe, was denke ich - ärgere gerade stosse an psychologen

[27]mit oder ohne ärger, auch psychologe - putin alle müssen selber damit klar kommen - obwohl ich das nicht soll, müssen selber damit klar kommen und - ...auch wenn nicht glauben, ich mache doch mathe, dann müssen damit klar kommen, dass ich tue - ...- oder: melde mich aber ab, damit kann ich gegen willen anderer. abgemeldet jetzt! wenn andere glauben, das ist jetzt psycho arbeit egal - weil: damit müssen die und ich klar kommen und das hat nichts mit - ruhm und ehre zu tun - weil: ich muss das halten machen, wie die interpretieren - deren problem - diese worte und melde mich ab vom dienst.

[28]das ist jetzt: normaler/grober/klassischer automat zum finden von zeichenfolgen in zeichenfolgen, eines musters in einem text. meist text-zeichenfolgen in text-zeichenfolgen. mit klassisch, das heisst, komplett zurücksetzen.

[29]davor fällt mir auf - gucken wir uns das an - es fehlt die markierung, wo üebereinstimmt, blos wie markieren? farbige tabelleninhalte, oder was? [30]mir fällt auf, man kann das ganze auch mit noten machen - zunächst buchstaben noten - aber in LaTeXangezeigten richtigen noten, was das gleiche sein könnte

an einer stelle war die angabe natürlich falsch - nämlich wenn ich im muster, nach dem muster suche, so wie wenn in dem muster, indem ich als text suche, ..., jedenfalls, dann ist

```
i
```

beteiligt, mit

```
next [i] = j;
```

[31]an einer stelle war die angabe natürlich falsch - nämlich wenn ich im muster, nach dem muster suche, so wie wenn in dem muster, indem ich als text suche, ..., jedenfalls, dann ist i beteiligt, mit $next(i) = j$;

hier die algorithmen,

1. Knuth-Morris-Pratt

   (a) Die Suche

   ```
   int kmpsearch (char *p, char *a) {
    int i, j, M = strlen (p), N = strlen (a);
    initnext (p);

    for (i = 0, j = 0;  j < M && i < N;  i++, j++)
     while ((j >= 0) && (a[i] != p[j])) j = next [j];
    if (j == M) return i-M; else return i;

   }
   ```

   (b) Der Vorlauf

   ```
   initnext (char *p) {
    int i, j, M = strlen (p);
    next [0] = -1;
    for (i = 0, j = -1; i < M;  i++, j++, next [i] = j)
     while ((j >= 0) && (p[i] != p[j])) j = next [j];
   }
   ```

2. der grobe Algorithmus

   ```
   int brutesearch (char *p, char *a) {
    int i, j, M = strlen (p), N = strlen(a);
    for (i = 0, j = 0; j < M && i < N; i++, j++)
     while (a [i] != p [j]) {i -= j-1; j = 0; }
    if (j == M) return i-M; else return i;
   }
   ```

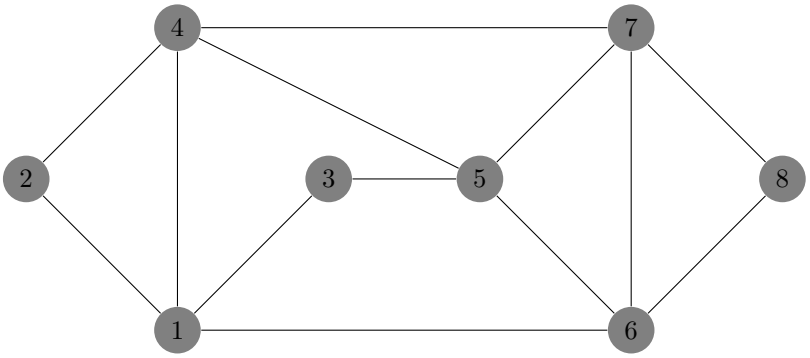
1	~~0~~	1	0	1	1	0	0	0	1	0	1	1	1	1	1											
	~~1~~	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1										
-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1											
1	0	***1***	0	1	1	0	0	0	1	0	1	1	1	1	1											
		1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1									
-1	0	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1											
1	0	***1***	***0***	1	1	0	0	0	1	0	1	1	1	1	1											
		1	***0***	1	0	1	1	0	0	0	1	0	1	1	1	1	1									
1	0	***1***	***0***	***1***	1	0	0	0	1	0	1	1	1	1	1											
		1	***0***	***1***	0	1	1	0	0	0	1	0	1	1	1	1	1									
1	0	***1***	***0***	***1***	~~1~~	0	0	0	1	0	1	1	1	1	1											
		1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1									
1	0	1	0	1	***1***	0	0	0	1	0	1	1	1	1	1											
					1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1						
1	0	1	0	1	***1***	***0***	0	0	1	0	1	1	1	1	1											
					1	***0***	1	0	1	1	0	0	0	1	0	1	1	1	1	1						
1	0	1	0	1	***1***	***0***	~~0~~	0	1	0	1	1	1	1	1											
					1	***0***	~~1~~	0	1	1	0	0	0	1	0	1	1	1	1	1						
1	0	1	0	1	1	0	~~0~~	0	1	0	1	1	1	1	1											
							~~1~~	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1				
1	0	1	0	1	1	0	0	~~0~~	1	0	1	1	1	1	1											
								~~1~~	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1	1		
1	0	1	0	1	1	0	0	0	***1***	0	1	1	1	1	1											
									1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	1	1	
1	0	1	0	1	1	0	0	0	***1***	***0***	1	1	1	1	1											
									1	***0***	1	0	1	1	0	0	0	1	0	1	1	1	1	1	1	
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	1	1	1	1											
									1	***0***	***1***	0	1	1	0	0	0	1	0	1	1	1	1	1	1	
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	~~1~~	1	1	1											
									1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1	1	
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	~~1~~	1	1	1											
									1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1	1	
1	0	1	0	1	1	0	0	0	***1***	***0***	***1***	~~1~~	1	1	1											
									1	***0***	***1***	~~0~~	1	1	0	0	0	1	0	1	1	1	1	1	1	
1	0	1	0	1	1	0	0	0	1	0	1	***1***	1	1	1											
												1	0	1	0	1	1	0	0	0	1	0	1			
1	0	1	0	1	1	0	0	0	1	0	1	***1***	~~1~~	1	1											
												1	~~0~~	1	0	1	1	0	0	0	1	0	1			
1	0	1	0	1	1	0	0	0	1	0	1	1	***1***	1	1											
													1	0	1	0	1	1	0	0	0	1	0	1		
1	0	1	0	1	1	0	0	0	1	0	1	1	***1***	~~1~~	1											
													1	~~0~~	1	0	1	1	0	0	0	1	0	1		
1	0	1	0	1	1	0	0	0	1	0	1	1	1	***1***	1											
														1	0	1	0	1	1	0	0	0	1	0	1	
1	0	1	0	1	1	0	0	0	1	0	1	1	1	***1***	~~1~~											
														1	~~0~~	1	0	1	1	0	0	0	1	0	1	
1	0	1	0	1	1	0	0	0	1	0	1	1	1	1	***1***											
															1	0	1	0	1	1	0	0	0	1	0	1

ok, dazu sei an zu merken, dazu fehlt mir gerade die geduld, es ist 19:11 und: der anfang stimmt nicht optimal, ich spiele eine runde klavier, mache dann quine mc cluskey, als konzentrative zwischenaufgabe, inzwischen danach die eulertour von neulich fertig, inzwischen einen kaffee. die sache ist ja weit gediehen, dass der vergleich wenigstens drunter stimmt, auch wenn

```
next []
```

noch nicht stimmt. [32]ok, dazu sei an zu merken, dazu fehlt mir gerade die geduld, es ist 19:11 und: der anfang stimmt nicht optimal, ich spiele eine runde klavier, mache dann quine mc cluskey, als konzentrative zwischenaufgabe, inzwischen danach die eulertour von neulich fertig, inzwischen einen kaffee. die sache ist ja weit gediehen, dass der vergleich wenigstens drunter stimmt, auch wenn $next()$ noch nicht stimmt.

4 euler touren



jetzt machen wir das ganze graphisch: indem ich das optisch löse:

```
Kreis A: (2,4,1,2)
Kreis B: (1,3,5,4,1)
Kreis C: (5,6,7,5)
Kreis D: (6,8,7,6)

als mengen, oder listen:

A = (2,4,1,2)
B = (1,3,5,4,1)
C = (5,6,7,5)
D = (6,8,7,6)

# also das ist jetzt graphisch
# und graphisch bedeutet:
# ich mache das als knoten die kreise ...
# so wie man sieht, ohne perfekte mathematische
# notierung wuerde bei hamilton kreisen
# als: Kanten gehen. wuerde aber auch mit Knoten
# gehen, das sind listen, muss aber nicht sein, kann
# auch mengen sein, ohne korrekte reihenfolge
```

```
# nachher nachdem hier reinen text, als mathematische
# formulierung, vorher hier als menge in rein text

# man kann in Mengen elemente doppelt nehmen
# muesste man bei knotenangabe
# oder - man nimmt nur die beteiligten knoten

A = {1,2,2,4}
B = {1,1,3,4,5}
C = {5,5,6,7}
D = {6,6,7,8}

A = {1,2,4}
B = {1,3,4,5}
C = {5,6,7}
D = {6,7,8}

# jetzt im naechsten schritt noch mal die alte schreibweise
# nachher latex korrekt mathematisch. und: aber nachher richtige
# mathematische abarbeitung fuer das programm ueblich

# Versuch auch tiefensuche und breitensuche.

# jetzt weiter...

# bei hamilton kreise, ich glaube das sind welche -
# setzt man den einen kreis, wo punkte uebereinstimmen
# in den anderen ein


A = (2,4,1,2)
B = (1,3,5,4,1)
C = (5,6,7,5)
D = (6,8,7,6)

A = (2,4,(1,3,5,4,1),2)
C = (5,6,7,5)
D = (6,8,7,6)

A = (2,4,(1,3,(5,6,7,5),4,1),2)
D = (6,8,7,6)

A = (2,4,(1,3,(5,(6,8,7,6),7,5),4,1),2)

das ergibt den Kreis gesamt

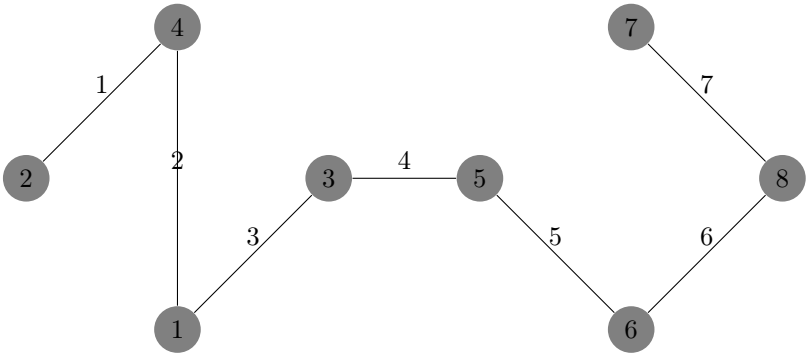
EU = (2,4,1,3,5,6,8,7,6,7,5,4,1,2)

ich mache das jetzt mathematisch, dann mit graph, wo die nummerierung
dran ist, bei den schritten und, wenn das passiert ist. dann mit kanten
```

mathematischen verfahren, versuch tiefen und breiten suche zu tun was relativ einfach ist, und noch ordentliche tabelle

dann versuch - DNF - quine, wie immer und der rest

$$EU = (2, 4, 1, 3, 5, 6, 8, 7, 6, 7, 5, 4, 1, 2)$$



an dieser stelle sieht man da geht was nicht.
mir ist aufgefallen, ich habe die hamilton kreise, in verschiedene richtungen gezeichnet. diese müssen alle in die gleiche richtung gezeichnet werden.
so, das habe ich falsch gemacht. [33]habe ich vergessen in die fussnote rein muss

```
vorher:
Kreis A: (2,1,4,2)
nachher:
Kreis A: (2,1,4,2)
Kreis B: (1,3,5,4,1)
Kreis C: (5,6,7,5)
Kreis D: (6,8,7,6)

als mengen, oder listen:

vorher:
A = (2,4,1,2)
nachher:
A = (2,4,1,2)
B = (1,3,5,4,1)
C = (5,6,7,5)
D = (6,8,7,6)

# also das ist jetzt graphisch
# und graphisch bedeutet:
# ich mache das als knoten die kreise ...
# so wie man sieht, ohne perfekte mathematische
# notierung wuerde bei hamilton kreisen
# als: Kanten gehen. wuerde aber auch mit Knoten
# gehen, das sind listen, muss aber nicht sein, kann
# auch mengen sein, ohne korrekte reihenfolge
# nachher nachdem hier reinen text, als mathematische
```

```
# formulierung, vorher hier als menge in rein text

# man kann in Mengen elemente doppelt nehmen
# muesste man bei knotenangabe
# oder - man nimmt nur die beteiligten knoten

A = {1,2,2,4}
B = {1,1,3,4,5}
C = {5,5,6,7}
D = {6,6,7,8}

A = {1,2,4}
B = {1,3,4,5}
C = {5,6,7}
D = {6,7,8}

# jetzt im naechsten schritt noch mal die alte schreibweise
# nachher latex korrekt mathematisch. und: aber nachher richtige
# mathematische abarbeitung fuer das programm ueblich

# Versuch auch tiefensuche und breitensuche.

# jetzt weiter...

# bei hamilton kreise, ich glaube das sind welche -
# setzt man den einen kreis, wo punkte uebereinstimmen
# in den anderen ein


# A = (2,4,1,2), falsch
A = (2,1,4,2) # richtig
B = (1,3,5,4,1)
C = (5,6,7,5)
D = (6,8,7,6)

A = (2,(1,3,5,4,1),4,2)
C = (5,6,7,5)
D = (6,8,7,6)

A = (2,4,(1,3,(5,6,7,5),4,1),2)
D = (6,8,7,6)

A = (2,4,(1,3,(5,(6,8,7,6),7,5),4,1),2)

das ergibt den Kreis gesamt

EU = (2,4,1,3,5,6,8,7,6,7,5,4,1,2)

ich mache das jetzt mathematisch, dann mit graph, wo die nummerierung
dran ist, bei den schritten und, wenn das passiert ist. dann mit kanten
```

```
mathematischen verfahren, versuch tiefen und breiten suche zu tun
was relativ einfach ist, und noch ordentliche tabelle

dann versuch - DNF - quine, wie immer und der rest
```

jetzt stelle ich fest, das war nicht der einzige fehler, eine sondersituation. euler selber, und das fehlt bei der aufgabe. hat bewiesen. solange ein Graph G nur aus Zyklen besteht und die anzahl der Valenzen, gerade ist, ist eine Eulertour machbar. trotzdem habe ich hier, unabhängig davon (Hamilton-Kreise) - müssen gleich gerichtet verlaufen, im Uhrzeigersinn oder dagegen.

ist hier ein sonderfall: weil hier ein viereck ist, mit einer Kante dazwischen, diese darf man nicht zwei mal laufen. damit gibt es einschränkungen für die kreise. bezogen auf den benachbarten kreis, sage ich trivialerweise, was sonst nicht der fall ist, und die eulertour trotzdem auf jeden fall lösbar.
trotzdem fehlt mir so oder so auch, das ist std. bei aufgaben, auch, wenn einfach aussieht, die angabe der valenzen pro jeden knoten. jetzt.

trotzdem geht das vielleicht, vom graphischen ausgesehen mit der feststellung, wer die nachbarn sind, einfacher. probieren wir erst mal nur valenzen und dann noch die nachbarn und wieder valenzen. auch das geht - das eine verfahren schliesst das andere nicht aus.

$$
\begin{aligned}
val(1) &= 4 && (98)\\
val(2) &= 2 && (99)\\
val(3) &= 2 && (100)\\
val(4) &= 4 && (101)\\
val(5) &= 4 && (102)\\
val(6) &= 4 && (103)\\
val(7) &= 4 && (104)\\
val(8) &= 2 && (105)\\
& && (106)
\end{aligned}
$$

und jetzt als folge

$$(4, 2, 2, 4, 4, 4, 4, 2)$$

ok, und nun weiter.

Jetzt die Edges/Kanten/ sprich nachbarn

$$
\begin{aligned}
E(1) &= (2,3,4,6) && (107)\\
E(2) &= (1,4) && (108)\\
E(3) &= (1,5) && (109)\\
E(4) &= (1,2,5,7) && (110)\\
E(5) &= (3,4,6,7) && (111)\\
E(6) &= (1,5,7,8) && (112)\\
E(7) &= (4,5,6,8) && (113)\\
E(8) &= (6,7) && (114)\\
& && (115)
\end{aligned}
$$

jetzt kann man vielleicht die kardinaliät davon herstellen.

$$
\begin{aligned}
|E(1)| &= |(2,3,4,6)| = 4 && (116)\\
|E(2)| &= |(1,4)| = 2 && (117)\\
|E(3)| &= |(1,5)| = 2 && (118)\\
|E(4)| &= |(1,2,5,7)| = 4 && (119)\\
|E(5)| &= |(3,4,6,7)| = 4 && (120)\\
|E(6)| &= |(1,5,7,8)| = 4 && (121)\\
|E(7)| &= |(4,5,6,8)| = 4 && (122)\\
|E(8)| &= |(6,7)| = 2 && (123)\\
& && (124)
\end{aligned}
$$

das sieht ziemlich identisch aus mit dem vorher.

$$
\begin{aligned}
val(1) &= 4 && (125)\\
val(2) &= 2 && (126)\\
val(3) &= 2 && (127)\\
val(4) &= 4 && (128)\\
val(5) &= 4 && (129)\\
val(6) &= 4 && (130)\\
val(7) &= 4 && (131)\\
val(8) &= 2 && (132)\\
& && (133)
\end{aligned}
$$

ok. natürlich fällt mir auf, natürlich der lösung unabhängig der hamilton kreise, wie ich die bisher mit punkten gemacht habe, gibt man besser kanten an, das ist die mathematische lösung, streicht sie, eines nach der anderen, und es wird nichts übrig bleiben. es wird automatisch gehen, das ist aber die graphische lßung mit punkten. das problem, das fällt mir auf - übrigens, das ist bei der mathematischen lösung über kanten sicher nicht so, indem ich sie einfach herunter nehme. sondern: der Kreis

$$A = (1,4,2,1)$$

bedeutet. dass der kreis geschlossen ist, blos die kante

$$(1, 4)$$

lässt sich nicht wiederverwenden. entweder äusserer kreis, oder man kann den kreis auch ohne das machen, weil auch das sind nachbarkreise.

```
A = (1,4,2,1)
B = (1,6,5,3,1)
C = (5,7,4,5)
D = (7,6,8,7)

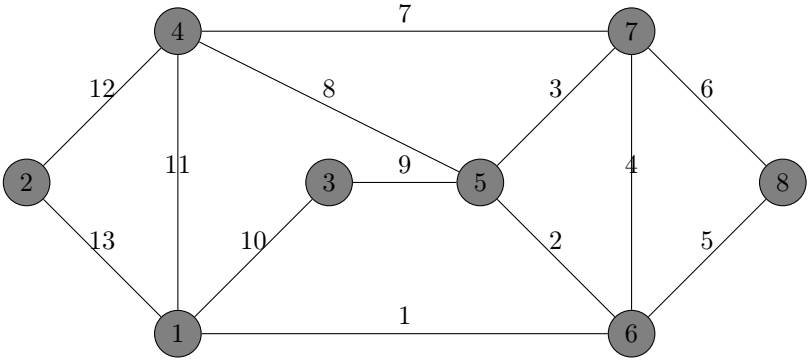
# zusammen setzen

E = ((1,6,5,3,1),4,2,1)
C = (5,7,4,5)
D = (7,6,8,7)

F = ((1,6,(5,7,4,5),3,1),4,2,1)
D = (7,6,8,7)

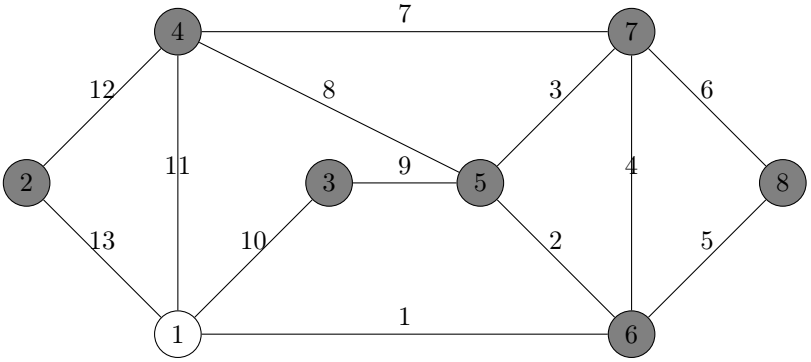
F = ((1,6,(5,(7,6,8,7),4,5),3,1,4,2,1)

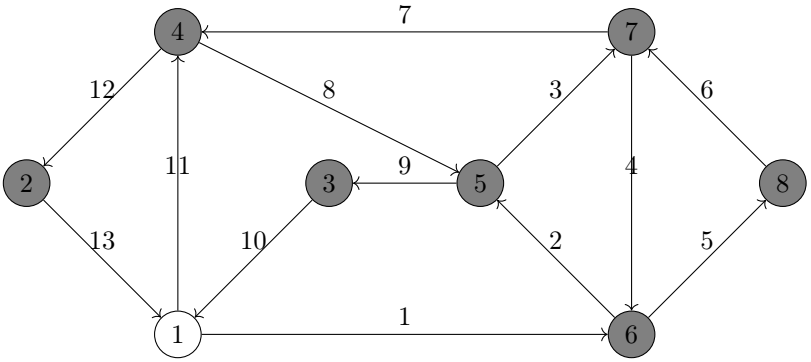
F = (1,6,5,7,6,8,7,4,5,3,1,4,2,1)
```



$$F = (1, 6, 5, 7, 6, 8, 7, 4, 5, 3, 1, 4, 2, 1)$$

jetzt noch mit pfeilen:





jetzt probieren wir es auf die mathematische tour: ich schreibe, die kanten auf, einerseits geht das mit mengen, als paare, andererseits, als tupel, listen usw. und streiche sie dann stück für stück indem sie zusammenpassen und: die sache löst sich von alleine.
bei paaren ist darauf zu achten, sie existieren zwei fach, weil der Nachbar A des Nachbarn B ist umgekehrt ebenso nachbar. und: hier bei listen ist die reihenfolge vorgeschrieben, sonst nicht.

```
\begin{tikzpicture}
\node [circle, fill=white,draw=black] (circle1) at (2,0) {1};
\node [circle, fill=gray,draw=black] (circle2) at (0,2) {2};
\node [circle, fill=gray,draw=black] (circle3) at (4,2) {3};
\node [circle, fill=gray,draw=black] (circle4) at (2,4) {4};

\node [circle, fill=gray,draw=black] (circle5) at (6,2) {5};
\node [circle, fill=gray,draw=black] (circle6) at (8,0) {6};
\node [circle, fill=gray,draw=black] (circle7) at (8,4) {7};
\node [circle, fill=gray,draw=black] (circle8) at (10,2) {8};

\draw [->] (circle1) edge node [above] {1} (circle6);
\draw [->] (circle6) edge node [above] {2} (circle5);
\draw [->] (circle5) edge node [above] {3} (circle7);
\draw [->] (circle7) edge node [above] {4} (circle6);
\draw [->] (circle6) edge  node [above] {5} (circle8);
\draw [->] (circle8) edge  node [above] {6} (circle7);
\draw [->] (circle7) edge  node [above] {7} (circle4);
\draw [->] (circle4) edge  node [above] {8} (circle5);
\draw [->] (circle5) edge  node [above] {9} (circle3);
\draw [->] (circle3) edge  node [above] {10} (circle1);
\draw [->] (circle1) edge  node [above] {11} (circle4);
\draw [->] (circle4) edge  node [above] {12} (circle2);
\draw [->] (circle2) edge  node [above] {13} (circle1);
\end{tikzpicture}
```

$$
\begin{aligned}
E &= \{ && (134)\\
E_1 &= ((1,2),(1,3),(1,4),(1,6)), && (135)\\
E_2 &= ((2,1),(2,4)), && (136)\\
E_3 &= ((3,1),(3,5)), && (137)\\
E_4 &= ((4,1),(4,2),(4,5),(4,7)), && (138)\\
E_5 &= ((5,3),(5,4),(5,6),(5,7)), && (139)\\
E_6 &= ((6,1),(6,5),(6,7),(6,8)), && (140)\\
E_7 &= ((7,4),(7,5),(7,6),(7,8)), && (141)\\
E_8 &= ((8,6),(8,7))\} && (142)\\
& && (143)
\end{aligned}
$$

nachher, noch mit mengenschreibweise für die paare, nachbarn. jetzt streichen wir die einzeln, machen eine zahl hin hoch. als index, und streichen durch. und machen eine zweite liste, da kommen die paare rein.

$$
\begin{aligned}
E &= \{ && (144)\\
E_1 &= (\underline{(1,2)_1},(1,3),(1,4),(1,6)), && (145)\\
E_2 &= (\underline{(2,1)_1},\underline{(2,4)_2}), && (146)\\
E_3 &= ((3,1),(3,5)), && (147)\\
E_4 &= ((4,1),\underline{(4,2)_2},(4,5),(4,7)), && (148)\\
E_5 &= ((5,3),(5,4),(5,6),(5,7)), && (149)\\
E_6 &= ((6,1),(6,5),(6,7),(6,8)), && (150)\\
E_7 &= ((7,4),(7,5),(7,6),(7,8)), && (151)\\
E_8 &= ((8,6),(8,7))\} && (152)\\
& && (153)
\end{aligned}
$$

$$EU = ((1,2),(2,4),$$

jetzt könnte einzelne hamilton kreise nehmen, das dauert zu lange, also setze ich wo anders fort, aus dem kopf, sichtbar, wo nicht in den alten kreis führt, das

ist abschätzung.

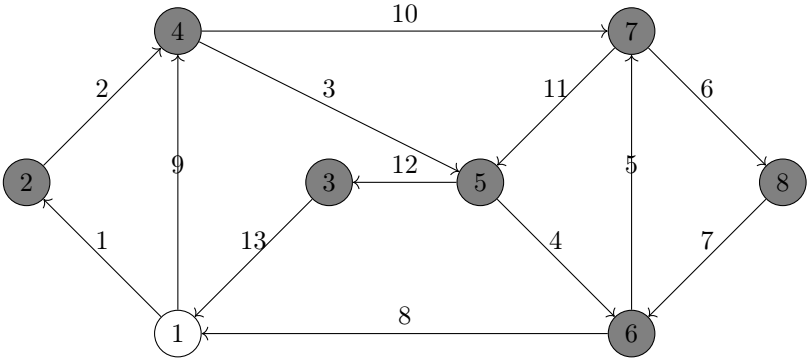
$$
\begin{aligned}
E &= \{ && (154)\\
E_1 &= (\underline{(1,2)_1}, \underline{(1,3)_{13}}, \underline{(1,4)_9}, \underline{(1,6)_8}), && (155)\\
E_2 &= (\underline{(2,1)_1}, \underline{(2,4)_2}), && (156)\\
E_3 &= (\underline{(3,1)_{13}}, \underline{(3,5)_{12}}), && (157)\\
E_4 &= (\underline{(4,1)_9}, \underline{(4,2)_2}, \underline{(4,5)_3}, \underline{(4,7)_{10}}), && (158)\\
E_5 &= (\underline{(5,3)_{12}}, \underline{(5,4)_3}, \underline{(5,6)_4}, \underline{(5,7)_{11}}), && (159)\\
E_6 &= (\underline{(6,1)_8}, \underline{(6,5)_4}, \underline{(6,7)_5}, \underline{(6,8)_7}), && (160)\\
E_7 &= (\underline{(7,4)_{10}}, \underline{(7,5)_{11}}, \underline{(7,6)_5}, \underline{(7,8)_6}), && (161)\\
E_8 &= (\underline{(8,6)_7}, \underline{(8,7)_6})\} && (162)\\
& && (163)
\end{aligned}
$$

$$
\begin{aligned}
EU &= ((1,2),(2,4),(4,5),(5,6),(6,7), && (164)\\
&= (7,8),(8,6),(6,1),(1,4),(4,7),(7,5),(5,3),(3,1)) && (165)
\end{aligned}
$$

so, dann machen wir das noch mit nummerierung

$$
\begin{aligned}
EU &= ((1,2)_1,(2,4)_2,(4,5)_3,(5,6)_4, && (166)\\
&\quad (6,7)_5,(7,8)_6,(8,6)_7,(6,1)_8,(1,4)_9, && (167)\\
&\quad (4,7)_{10},(7,5)_{11},(5,3)_{12},(3,1)_{13}) && (168)\\
& && (169)
\end{aligned}
$$

daran dass die letzte kante, im letzten graphen, tatsächlich 13 war, kann man sehen, dass: die zahl hier wohl stimmt, im nächsten setze ich diesen graphen um. ich zeige auch wiederum den quelltext von tikz - also dem graph und bild.



$$
\begin{aligned}
EU &= ((1,2)_1,(2,4)_2,(4,5)_3,(5,6)_4, && (170)\\
&\quad (6,7)_5,(7,8)_6,(8,6)_7,(6,1)_8,(1,4)_9, && (171)\\
&\quad (4,7)_{10},(7,5)_{11},(5,3)_{12},(3,1)_{13}) && (172)\\
& && (173)
\end{aligned}
$$

das hat auch funktioniert

```
\begin{tikzpicture}
\node [circle, fill=white,draw=black] (circle1) at (2,0) {1};
\node [circle, fill=gray,draw=black] (circle2) at (0,2) {2};
\node [circle, fill=gray,draw=black] (circle3) at (4,2) {3};
\node [circle, fill=gray,draw=black] (circle4) at (2,4) {4};

\node [circle, fill=gray,draw=black] (circle5) at (6,2) {5};
\node [circle, fill=gray,draw=black] (circle6) at (8,0) {6};
\node [circle, fill=gray,draw=black] (circle7) at (8,4) {7};
\node [circle, fill=gray,draw=black] (circle8) at (10,2) {8};

\draw [->] (circle1) edge node [above] {1} (circle2);
\draw [->] (circle2) edge node [above] {2} (circle4);
\draw [->] (circle4) edge node [above] {3} (circle5);
\draw [->] (circle5) edge node [above] {4} (circle6);
\draw [->] (circle6) edge  node [above] {5} (circle7);
\draw [->] (circle7) edge  node [above] {6} (circle8);
\draw [->] (circle8) edge  node [above] {7} (circle6);
\draw [->] (circle6) edge  node [above] {8} (circle1);
\draw [->] (circle1) edge  node [above] {9} (circle4);
\draw [->] (circle4) edge  node [above] {10} (circle7);
\draw [->] (circle7) edge  node [above] {11} (circle5);
\draw [->] (circle5) edge  node [above] {12} (circle3);
\draw [->] (circle3) edge  node [above] {13} (circle1);
\end{tikzpicture}
```

jetzt kommt, die tiefen und breitensuche, wenn geht. ich vollziehe nur das prinzip, ich nehme eine schlange und keinen stapel, ich lege den ersten drauf und seine nachbarn gehe durch und tue sie auf die schlange hinten anfügen, einfach mathematisch vom prinzip. ich habe daneben die listen, der schon besuchten und: ich habe: den zusammenhang quasi. erst mal, aber ohne korrekte tabelle, einfach mal aufschreiben, die tabelle wird mir schon einfallen. inzwischen zigaretten pause. ok, ich sagte ja, [34]eigentlich gehört hier die fussnote hin, blos, das problem ist folgendes - verwechselt - modus operandi? ist das teil bei der operation beim schreiben - weiss nicht - schamhaft - ... keine näheren erläuterungen, ... überschrift

5 quine mc cluskey/dnf/10/23/2026

like always next thema. was fehlt noch? toilette. ok, was fehlt ist: interpolation - lagrange - und so weiter. also, zunächst rechnen wir wieder für drei x-stellen

das polynom aus
diese bleiben bei gleichen x-en gleich. so: es sind drei teilpolynome

$$(l_2(x), l_1(x), l_0(x))$$

, deren namen ich nicht kenne. wie diese teilpolynome heissen. die formel für sie geht so. erst im zweiten schritt, werden die y-werte eingesetzt.

$$\begin{aligned}
l_i(x) &= \sum_{\substack{j=0 \\ j\neq i}}^{2} \frac{x - x_j}{x_i - x_j} && (174)\\
l_0(x) &= \frac{x - x_1}{x_0 - x_1} + \frac{x - x_2}{x_0 - x_2} && (175)\\
l_1(x) &= \frac{x - x_0}{x_1 - x_0} + \frac{x - x_2}{x_1 - x_2} && (176)\\
l_2(x) &= \frac{x - x_0}{x_2 - x_0} + \frac{x - x_1}{x_2 - x_1} && (177)\\
& && (178)
\end{aligned}$$

Falsch ist allerdings die Summe habe ich erkannt. es ist keine Summe, sondern ein Produkt, erst bei der zusammensetzung zum Lagrange-Polynom (?) eigentlich nicht zum lagrange polynom, sondern zum interpolierten Polynom, kommen Summen in frage.

$$\begin{aligned}
l_i(x) &= \prod_{\substack{j=0 \\ j\neq i}}^{2} \frac{x - x_j}{x_i - x_j} && (179)\\
l_0(x) &= \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} && (180)\\
l_1(x) &= \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} && (181)\\
l_2(x) &= \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1} && (182)\\
& && (183)
\end{aligned}$$

für

$$X = (1, 2, 3) = (x_0 = 1, x_1 = 2, x_2 = 3)$$

kommt heraus.

$$\begin{aligned}
l_i(x) &= \prod_{\substack{j=0 \\ j\neq i}}^{2} \frac{x - x_j}{x_i - x_j} && (184)\\
l_0(x) &= \frac{x - 2}{1 - 2} \cdot \frac{x - 3}{1 - 3} && (185)\\
l_1(x) &= \frac{x - 1}{2 - 1} \cdot \frac{x - 3}{2 - 3} && (186)\\
l_2(x) &= \frac{x - 1}{3 - 1} \cdot \frac{x - 2}{3 - 2} && (187)\\
& && (188)
\end{aligned}$$

und das geht so weiter:

$$
\begin{aligned}
l_i(x) &= \prod_{\substack{j=0 \\ j\neq i}}^{2} \frac{x-x_j}{x_i-x_j} && (189)\\
l_0(x) &= \frac{x-2}{1-2}\cdot\frac{x-3}{1-3} && (190)\\
&= \frac{x^2-3\cdot x-2\cdot x+6}{2} && (191)\\
l_1(x) &= \frac{x-1}{2-1}\cdot\frac{x-3}{2-3} && (192)\\
&= \frac{x^2-3\cdot x-x+3}{-1} && (193)\\
l_2(x) &= \frac{x-1}{3-1}\cdot\frac{x-2}{3-2} && (194)\\
&= \frac{x^2-2\cdot x-x+2}{2} && (195)\\
& && (196)
\end{aligned}
$$

ok

$$
\begin{aligned}
l_i(x) &= \prod_{\substack{j=0 \\ j\neq i}}^{2} \frac{x-x_j}{x_i-x_j} && (197)\\
l_0(x) &= \frac{x-2}{1-2}\cdot\frac{x-3}{1-3} && (198)\\
&= \frac{x^2-5\cdot x+6}{2} && (199)\\
l_1(x) &= \frac{x-1}{2-1}\cdot\frac{x-3}{2-3} && (200)\\
&= \frac{x^2-4\cdot x+3}{-1} && (201)\\
l_2(x) &= \frac{x-1}{3-1}\cdot\frac{x-2}{3-2} && (202)\\
&= \frac{x^2-3\cdot x+2}{2} && (203)\\
& && (204)
\end{aligned}
$$

ok, jetzt alles auf einen nenner bringen, das war wohl beim letzten mal der fehler beim addieren.

$$
\begin{aligned}
p(x) &= \sum_{i=0}^{2} y_i\cdot l_i(x) && (205)\\
& && (206)
\end{aligned}
$$

ok: das „mittlere“polynome mit -2 multiplizieren

$$
\begin{aligned}
l_i(x) &= \prod_{\substack{j=0 \\ j\neq i}}^{2} \frac{x-x_j}{x_i-x_j} && (207)\\
l_0(x) &= \frac{x-2}{1-2}\cdot\frac{x-3}{1-3} && (208)\\
&= \frac{x^2-5\cdot x+6}{2} && (209)\\
l_1(x) &= \frac{x-1}{2-1}\cdot\frac{x-3}{2-3} && (210)\\
&= \frac{x^2-4\cdot x+3}{-1} && (211)\\
&= \frac{-2\cdot x^2+8\cdot x-6}{2} && (212)\\
l_2(x) &= \frac{x-1}{3-1}\cdot\frac{x-2}{3-2} && (213)\\
&= \frac{x^2-3\cdot x+2}{2} && (214)\\
& && (215)
\end{aligned}
$$

sieht man genau hin, sieht es so aus, als würde x^2 nun raus fallen, bzw. nicht das. sondern das x zur zweiten potenz. das ist allerdings falsch, weil die y kommen erst noch, und zwar jetzt.

$$Y = (2, 3, 1)$$

$$
\begin{aligned}
l_i(x)\cdot y_i &= \prod_{\substack{j=0 \\ j\neq i}}^{2} \frac{x-x_j}{x_i-x_j} && (216)\\
l_0(x)\cdot y_0 &= y_0\left(\cdot\frac{x-2}{1-2}\cdot\frac{x-3}{1-3}\right) && (217)\\
&= 2\cdot\left(\frac{x^2-5\cdot x+6}{2}\right) && (218)\\
l_1(x)\cdot y_0 &= y_1\cdot\left(\frac{x-1}{2-1}\cdot\frac{x-3}{2-3}\right) && (219)\\
&= 3\cdot\left(\frac{x^2-4\cdot x+3}{-1}\right) && (220)\\
&= 3\cdot\left(\frac{-2\cdot x^2+8\cdot x-6}{2}\right) && (221)\\
l_2(x)\cdot y_2 &= y_2\cdot\left(\frac{x-1}{3-1}\cdot\frac{x-2}{3-2}\right) && (222)\\
&= 1\cdot\left(\frac{x^2-3\cdot x+2}{2}\right) && (223)\\
& && (224)
\end{aligned}
$$

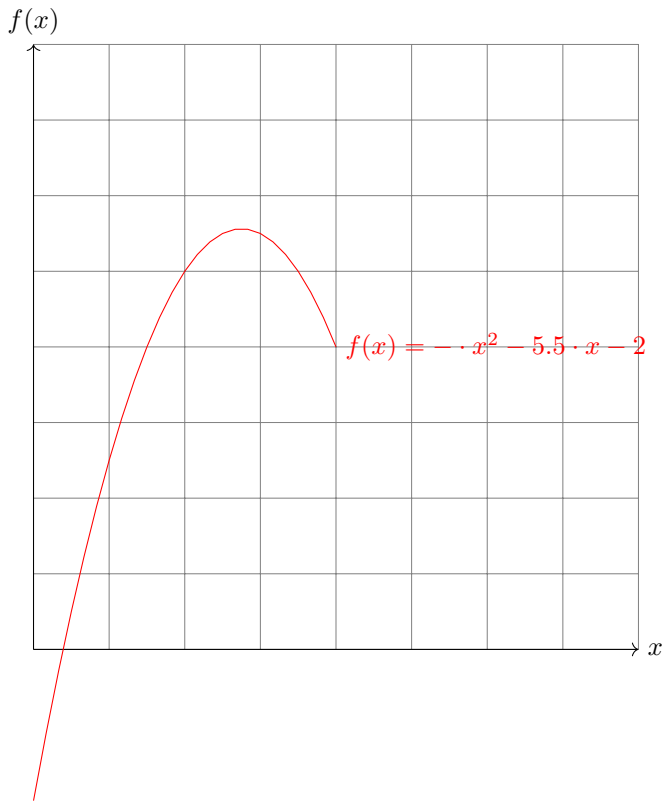
ok, jetzt alles auf e

$$
\begin{aligned}
l_i(x) \cdot y_i &= \prod_{\substack{j=0 \\ j \neq i}}^{2} \frac{x - x_j}{x_i - x_j} && (225)\\
l_0(x) \cdot y_0 &= y_0 \left(\cdot \frac{x-2}{1-2} \cdot \frac{x-3}{1-3} \right) && (226)\\
&= 2 \cdot \left(\frac{x^2 - 5 \cdot x + 6}{2} \right) && (227)\\
&= \left(x^2 - 5 \cdot x + 6 \right) && (228)\\
l_1(x) \cdot y_1 &= y_1 \cdot \left(\frac{x-1}{2-1} \cdot \frac{x-3}{2-3} \right) && (229)\\
&= 3 \cdot \left(\frac{x^2 - 4 \cdot x + 3}{-1} \right) && (230)\\
&= \left(-3 \cdot x^2 + 12 \cdot x - 9 \right) && (231)\\
l_2(x) \cdot y_2 &= y_2 \cdot \left(\frac{x-1}{3-1} \cdot \frac{x-2}{3-2} \right) && (232)\\
&= 1 \cdot \left(\frac{x^2 - 3 \cdot x + 2}{2} \right) && (233)\\
&= x^2 - \frac{3}{2} \cdot x + 1 && (234)\\
& && (235)
\end{aligned}
$$

so, im nächsten können wir das wirklich addieren. wie beim multiplizieren geht hier auch eine tabelle

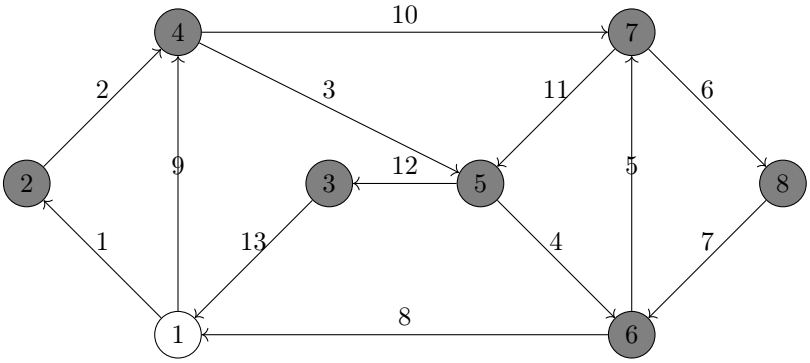
$$
\begin{aligned}
l_0(x) \cdot 0 &= \left(x^2 - 5 \cdot x + 6 \right) && (236)\\
l_1(x) \cdot y_1 &= \left(-3 \cdot x^2 + 12 \cdot x - 9 \right) && (237)\\
l_2(x) \cdot y_2 &= \left(x^2 - \frac{3}{2} \cdot x + 1 \right) && (238)\\
& && (239)
\end{aligned}
$$

i	$a_i x^2$	$b_i x$	c_i
0	x^2	$-5 \cdot x$	6
1	$-3 \cdot x^2$	$+12 \cdot x$	-9
2	x^2	$-1.5 \cdot x$	$+1$
$\sum$	$-x^2$	$5.5x$	-2



gut, besser geht nicht, ausrechnen

6 und nun die breitensuche



jetzt machen wir damit unabhängig der Pfeile, eine Tiefen- und Breitensuche - zumindest, erst eine Tiefensuche, und zwar indem wir in diesem bild, die pfeile vernachlässigen

[35]es wäre in zukunft, abbildung, code und so weiter, an die abbildungen, die codes an zu bringen

jetzt erst mal in normaler form, ohne konkrete tabelle. damit man das daraus ableitet, woraus man einen stil lernt. vom verfahren geht es so. man hat eine schlange, man hat ein Feld, in dem markiert man, die knoten, die bereits

bearbeitet wurden und: man bildet eine liste, die die reihenfolge angibt, in der die knoten besucht wurden. da es sich um zusammenhänge, sprich komponenten handelt, heisst das:

```
component []
visited [] // meinetwegen - oder: besser Robert sedgewick steht was.
```

der rest kommt in eine schlange. da nehmen wir C funktionen nehmen wir unseren eigenen code, ohne in zu implementieren, noch besser. wir nehmen keinen programmierquelltext das ist falsch, weil das ist mathe. und nicht ganz falsch, weil programmieren ist teil von mathe, ist auch egal,...
jedenfalls: man kann auch mathematische listen und mengen nehmen, die man dann ergänzt ohne das dazu tun programmiertechnischer anweisungen.

eine menge kann natürlich rein theoretisch auch `component` heissen, auch in mathematik. man muss hier zwischen listen und mengen unterscheiden, bei mengen geschweifte klammern bei listen runde

```
component = (1)
# komponente
nachbarn (1) =  (2,3,4,6)
# neighbour
visited = {1}
schlange = ...
# queue

queue = (2)
visited = {1,2}
component = (1,2)

#das ist tiefensuche, das. heisst, ich gehe jetzt erst durch die 2.
# jetzt muss man aber, eigentlich mathematisch, die bearbeiteten
# das ist halt im programmierquelltext anders, durch die anweisungen
# speichert so zu sagen, die queue - ... was auch immer
# die bearbeiteten und nicht bearbeiteten - knoten mit, das ist teil vom
# programmierquelltext, also, die die dann besucht wurden
# mathematisch muss das eigentlich gespeichert werden
# dann geht das gerade so weiter

queue = (2)
visited = {1,2}
component = (1,2)
# processed heisst bearbeitet bei stapel papieren
# logisch process im PC usw.

queue = (2,1)
visited = {1,2}
component = (1,2)
neighbours = ((1,(2,3,4))
processed ((1,(2)))

naechster schritt, bei 2 weiter:
```

```
ok, das ist jetzt eher rekursiv tiefensuche, breitensuche, ich kann jetzt auch gleich, sag
alle in visited rein tun, dafuer die schlange behalten, zwei methoden

die 2 ist am falschen ende gelandet
BREITENSUCHE
queue  = (1,2,3,4)
visited = {1}
component = (1,2,3,4)
neighbours = (

# ja gut ist ein fehler, am amfang ist auf der queue eine 1 und die wird aber entfernt!

BREITENSUCHE
queue  = (1,2,3,4)
visited = {1}
component = (1,2,3,4)
neighbours = ((1,(2,3,4))

# Das andere ist Tiefensuche
TIEFENSUCHE:
# das weiss ich jetzt nicht, wie man das nennt, wahrscheinlich auch Q, aber egal
Q = (1,2,3,4)
visited = {1,2}
component = (1,2)
# das ist eher ein stapel, als von hinten her
S = (1,2,3,4)
visited = {1,4}
component = (1,4)
neighbours = ((1,(2,3,4))

ok, naechster schritt:

BREITENSUCHE
Q  = (1,2,3,4)
visited = {1,2}
component = (1,2,3,4)
neighbours = ((1,(2,3,4),(2,1,4)))
TIEFENSUCHE:
Q = (1,2,3,4)
visited = {1,4}
component = (1,4)
```

6.1 quine/DNF

```
(C) David Vajda
Sat Oct 24 08:40:37 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    0
```

```
 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    1
 5 0 1 0 1    0
 6 0 1 1 0    0
 7 0 1 1 1    0
 8 1 0 0 0    1
 9 1 0 0 1    0
10 1 0 1 0    0
11 1 0 1 1    0
12 1 1 0 0    0
13 1 1 0 1    0
14 1 1 1 0    1
15 1 1 1 1    0
```

so, minimierung, ohne tabelle:

```
(C) David Vajda
Sat Oct 24 08:40:37 2026
4 Network - TTL - Disjunktive Normalform

 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    1
 8 1 0 0 0    1
14 1 1 1 0    1

Gruppe 1:
 1 0 0 0 1    1
 2 0 0 1 0    1
 4 0 1 0 0    1
 8 1 0 0 0    1
Gruppe 2:
 3 0 0 1 1    1
Gruppe 3:
14 1 1 1 0    1

(1,3)  0 0 x 1
(2,3)  0 0 1 x
(4)  0 1 0 0
(8)  1 0 0 0
(14)  1 1 1 0

dann kommt eine primimplikantentafel
  1 2 3 4 8 14
(1,3)   1  1
(2,3)   1 1
(4)     1
```

```
(8)      1
(14)       1
```

```
das sind alles primimplikanten, jetzt kommt die tabelle ordentlich:
```

n	x_3	x_2	x_1	x_0	y
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
8	1	0	0	0	1
14	1	1	1	0	1

n	x_3	x_2	x_1	x_0	y
$(1,3)$	0	0	x	1	1
$(2,3)$	0	0	1	x	1
(4)	0	1	0	0	1
(8)	1	0	0	0	1
(14)	1	1	1	0	1

	1	2	3	4	8	14
$(1,3)$	1		1			
$(2,3)$		1	1			
(4)				1		
(8)					1	
(14)						1

```
(1,3)  0 0 x 1
(2,3)  0 0 1 x
(4)  0 1 0 0
(8)  1 0 0 0
(14)  1 1 1 0
```

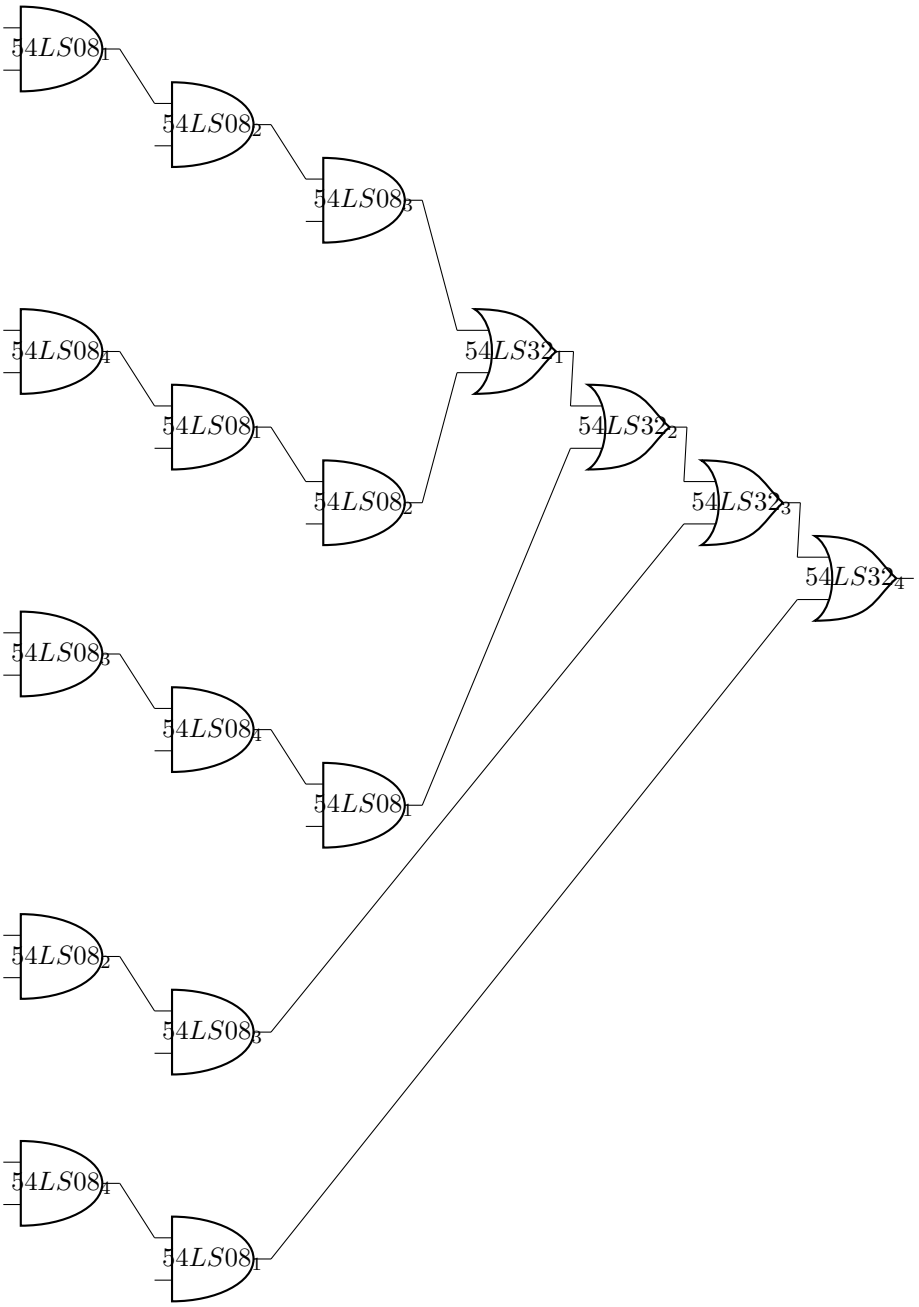
so, jetzt hier ABEL/GAL16v8/galasm code

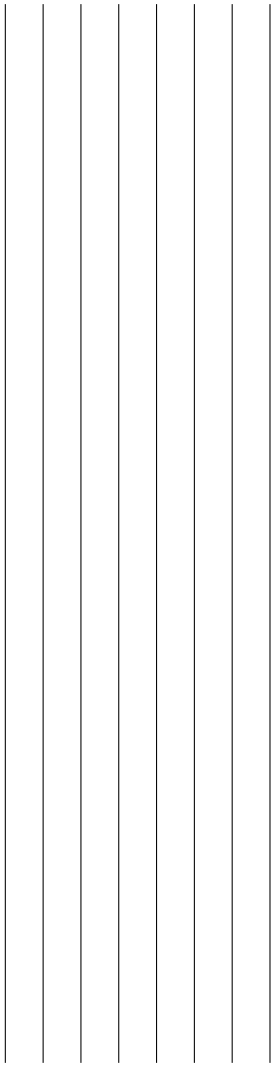
```
GAL16V8
MODULE QUINE10242026DNF

CLK X3 X2 X1 X0 NC NC NC NC GND
/OE Y NC NC NC  NC NC NC NC VCC

y = /x3 * /x2 * x0 +
  /x3 * /x2 * x1 +
  /x3 * x2 * /x1 * /x0 +
  x3 * /x2 * /x1 * /x0 +
  x3 * x2 * x1 * /x0
DESCRIPTION
QUINE DNF 1024 2026 DNF
```

so, jetzt kommt: das tikzpicture:


54LS08 1
54LS08 2
54LS08 3
54LS08 4
54LS08 1
54LS08 2
54LS08 3
54LS08 4
54LS08 1
54LS08 2
54LS08 3
54LS08 1
54LS32 1
54LS32 2
54LS32 3
54LS32 4



so, die sache ist jetzt so weit fertig. und geht jetzt weiter. und zunächst probieren wir sterne zu programmieren.

zunächst teilen wir

$$2 \cdot \pi \cdot r$$

durch beim pentagram durch 5, beim heptagragramm 7, beim eneagramm durch 9, usw

so, jetzt machen wir die sache weiter: wir errechnen zunächst für 5, 7, 9 die grade, um sterne mit viel mehr hin zu bekommen, bei denen allerdings - die knoten nicht eingezeichnet werden, sondern lediglich die kanten, müssen wir computer programme benutzen

fangen wir zunächst klein an:

ok, ohne vorher die tabellle zu tun, wir müssen erst mal ein gefühl für die sache bekommen, rechnen wir die gerade aus.

```
\begin{tabular}{|c|c|c|c|}
```

```
\hline
$i$ & \\
\hline

\hline
\end{tabular}

r = 4??

(2*pi*r)/5 =
(2*pi*r)/7 =
(2*pi*r)/9 =

summe:

w_5 = (2*pi*r/5) =
w_7 = (2*pi*r/7) =
w_9 = (2*pi*r)/9 =

sum_(i=1)^1 w_5 =
sum_(i=1)^1 w_7 =
sum_(i=1)^1 w_9 =

sum_(i=1)^2 w_5 =
sum_(i=1)^2 w_7 =
sum_(i=1)^2 w_9 =

sum_(i=1)^3 w_5 =
sum_(i=1)^3 w_7 =
sum_(i=1)^3 w_9 =

sum_(i=1)^4 w_5 =
sum_(i=1)^4 w_7 =
sum_(i=1)^4 w_9 =

sum_(i=1)^5 w_5 =
sum_(i=1)^5 w_7 =
sum_(i=1)^5 w_9 =

sum_(i=1)^6 w_7 =
sum_(i=1)^6 w_9 =

sum_(i=1)^7 w_7 =

sum_(i=1)^7 w_9 =
sum_(i=1)^8 w_9 =
sum_(i=1)^9 w_9 =

so, nachdem wir das dann errechnet haben, folgt sin(w_k) und cos(w_k) errechnung.
```

die fornel fuer die graphen, als sterne sollten wir im kopf haben

((1,n/2),(1,n/2+1))

das ist allerdings nur auf den knoten 1 bezogen

((m,(m+n)/2),(m,(m+n)/2+1))

vielleicht noch mod n
oder anders m < n/2

also, die folgen noch mal, mit sin(w_k) und cos(w_k)

sin(sum_(i=1)^1 w_5) =
sin(sum_(i=1)^1 w_7) =
sin(sum_(i=1)^1 w_9) =
cos(sum_(i=1)^1 w_5) =
cos(sum_(i=1)^1 w_7) =
cos(sum_(i=1)^1 w_9) =

sin(sum_(i=1)^2 w_5) =
sin(sum_(i=1)^2 w_7) =
sin(sum_(i=1)^2 w_9) =
cos(sum_(i=1)^2 w_5) =
cos(sum_(i=1)^2 w_7) =
cos(sum_(i=1)^2 w_9) =

sin(sum_(i=1)^3 w_5) =
sin(sum_(i=1)^3 w_7) =
sin(sum_(i=1)^3 w_9) =
cos(sum_(i=1)^3 w_5) =
cos(sum_(i=1)^3 w_7) =
cos(sum_(i=1)^3 w_9) =

sin(sum_(i=1)^4 w_5) =
sin(sum_(i=1)^4 w_7) =
sin(sum_(i=1)^4 w_9) =
cos(sum_(i=1)^4 w_5) =
cos(sum_(i=1)^4 w_7) =
cos(sum_(i=1)^4 w_9) =

sin(sum_(i=1)^5 w_5) =
sin(sum_(i=1)^5 w_7) =
sin(sum_(i=1)^5 w_9) =
cos(sum_(i=1)^5 w_5) =
cos(sum_(i=1)^5 w_7) =
cos(sum_(i=1)^5 w_9) =

```
sin(sum_(i=1)^6 w_7) =
sin(sum_(i=1)^6 w_9) =
cos(sum_(i=1)^6 w_7) =
cos(sum_(i=1)^6 w_9) =

sin(sum_(i=1)^7 w_7) =
cos(sum_(i=1)^7 w_7) =


sin(sum_(i=1)^7 w_9) =
sin(sum_(i=1)^8 w_9) =
sin(sum_(i=1)^9 w_9) =
cos(sum_(i=1)^7 w_9) =
cos(sum_(i=1)^8 w_9) =
cos(sum_(i=1)^9 w_9) =
```

gut, lassen wir das mit den sternen. ok, erst mal normale übungen zwischen drin, matrizen rechnung, vielleicht dann np vollständigkeit, nennen wir das mal! ist das np vollständig? hart? oder? n weil: ehrlich gesagt: wir haben zwei gleich grosse matrizen und sie gehören beide zum selben algo?
oder antwort gefunden: zwei matrizen?! ganz einfach: wir haben einen quelloperanten? ok? dann haben wir zwei? und warum deswegen np? falsch gedacht! wenn ich zwei gleich grosse matrizen habe?! und wie das teil thema... hätte ich fast gesagt! blos das ist quatsch! hier ist anwendung: das macht leuten spass! wann macht mein computer schlapp?! das ist nicht die richtige antwort! die anwendung spielt keine rolle. so wie bei graphen!

denken wir darüber nach! wenn wir anwendungen suchen: also es sind zwei fakten

1. die interessantere frage: ist das hier n oder np vollständigkeit?! das ist eigentlich das problem, und wenn ja oder nein, wieso und weshalb?!

    - die antworten sind einfach: aber nicht so trivial: wenn wir eine matrize haben und dann haben wir zwei?! ist das eine frage, des operanden und nun haben wir zwei?
    - die antwort lautet: falsch! weil es ist eine frage der operation und auch die taucht hier auf! warum? weil, wir es mit matrizenmultiplikation zu tun haben? ok: und wann haben wir es mit n vollständigkeit zu tun?
    - erst mal sind die begriffe vollständigkeit und so weiter falsch. erklären jetzt bitte nicht revolte. aber was der autor darunter versteht, welche lächerlichkeit, sagte man das in der antike so? dass ich sage: vollständigkeit usw. der richtige begriff! denken sie drüber nach was marx sagte: buttersäure hätte er falsch als summenforml angegeben! der herr marx! er fand sich ein wenig lächerlich! doch für das lächerliche und das hier ist verlagswesen! dass sie zu unrecht angreifen! gibt es gründe, ich lerne nämlich gerade NP Vollständigkeit und:

gehen sie nicht davon aus, dass das jemand vollständig wüsste! ganz einfach ,und nicht ich untersuche das und versuche zu verstehen! das kann man auswendig lernen! ich auswendig lernen! das sind begriffe! und wenn man die weiss, geht es! so: Auswendig! nur bin ihc gerade bei der überschrift Ich bin bei der überschrift und verewige mich aus egoismus! Und sie hoffentlich auch aus egoismus, wenn nicht jeder einer ist! geht es nicht mehr!

gehen sie davon aus! also: der begriff vollständigkeit ist falsch! vollständigkeit heisst es: wenn P und NP zusammen kommen! Das gilt nicht?! warum ist es allerdings von der Operation abhängig und das ist

ich weiss schon wie es geht! nur vollständigkeit ist nicht der richtige ausdruck. das heisst eher: N oder NP vollständigkeit. und woran es liegt! weil man mir nicht während dem schreiben ins zeug rein guckt! falscher korrektor! und weil ich noch physisch die seiten nicht aufgeschlagen habe?! kenen sie oder?! wenn man seiten nicht aufschlägt nicht buch liesst und es geht vor einem weg! ich verstehe was da steht, wenn ich auswendig lerne! sondern seiten nicht aufgeschlagen! nur ueberschrift!
also, noch mal die spiegelpunkte noch mal einfach!

2. die zweite frage lautet: anwendung?! die antworten kommen gleich

so, noch mal das thema:

1. die interessantere frage: ist das hier n oder np vollständigkeit?! das ist eigentlich das problem, und wenn ja oder nein, wieso und weshalb?!

   - die antworten sind einfach: aber nicht so trivial: wenn wir eine matrize haben und dann haben wir zwei?! ist das eine frage, des operanden und nun haben wir zwei?
   - es sind nicht die zwei operanten, denken sie drueber nach! und das heisst nicht operand! sondern operant! und daneben gibt es die operation. es ist keine frage von zwei operanten! es gibt ein gegenbeispiel und das ist sicher so! matrizenaddition zum beispiel! wenn ich, angenommen und das wäre std. argument, das steht übrigens nicht im buch, das problem liegt hier an anderer stelle: erklärungstechnisch: was ist problematik! nciht im gesellschaftliche sinne und was ist vollständigkeit, das kann unterscheiden einigermassen, vollständigkeit heisst: N und NP Gleichzeitig! und problematik! oder besser laufzeit! heisst: in wie viel schritten
   - in wie viel schritten heisst: also es ist keine frage von operanten! sondern der operation! nehmen sie eine matrizenaddition! Zwei matrizen!
   - sie addieren matrix A zur Matrix B. und zwar Spalte fuer Spalte und Zeile fuer Zeile was man anders herum sagt, das heisst Element fuer Element. die Matrizenmultiplikation wiederum ist NP. warum? weil P heisst, Addition! NP heisst, ich muss die mul und die Addiion, na ja fuer jedes element in der ausgangsmatrix kommt noch mal die entsprechende menge in der zielmatrix vorhanden, waehrend der rest ist wie es ist

- selbst kleine matrix, algo: untersucht heisst anzahl der schritte, wie in der chemie, 12 Gran oder 24 oder 64. das ist ein gravierender unterschied und muss dazu!

2. darauf sei hier hingewiesen: die anzahl der schritte, ist vielleicht in der mathematik ein problem, wie in der chemie! es geht nicht nur um elemente! um die substanz! es geht um das volumen, das gewicht/die massen, das hohlmass, die dauer, wie etwas abbrennt. zum beispiel S_{N1} und $SN2$ Reaktion. da geht es nicht einfach darum ob eine reaktion 2s oder 2 jahre dauert! das hat auswirkungen auf den stoff der entsteht! eine seife braucht lange zum reagieren! sie braucht mehrere Monate. wenn allerdings $SN1$ und $SN2$ stattfindet ist die frage: welche reaktion wird vorgeschoben! kommt die eine reaktion schneller zu stande als die andere! dann heisst: dann wird sie so verlaufen!

3. in der mathematik ist das oft ebenso entscheidend! wie viele schritte sind notwendig! wird das vernachlaessigt?! sind das chemiker, die nicht notieren, wie viel sie verbraten haben! Wie lange sie am feuer sassen wie viel stoecke sie verbrannt haben! dann bedeutet das! ein algorithmus mathematisch muss in der anzahl der schritte beschrieben werden und es geht nicht nur um ungeduldige user am PC die auf den Fortschrittsbalken warten!

4. die zweite frage lautet: anwendung?! die antworten kommen gleich

5. wenn man darueber nachdenkt! was anwendung heisst: soziologische probleme! haben keine frage des fortschrittsbalken. es geht nicht nur:

6. psychologie: noch ein problem: wenn ich von einer matrize auf das andere schliesse, von wegen anwendung! muss man wissne was im graphikprozessor passiert! wenn er naemlcih lauter punkte berechnet, dann berechnet er lauter matrizen

7. es ist nicht nru eine frage der geschwindigkeit damit der PC schnell angeht! ich kann manche sachen nicht berechnen, das ist anwendung! wenn die pixel zu lange brauchen!

8. gilt nur beim rauchen nicht und den feuerzeugen! sonst schon! und bitte keine herleitung! und sie machen was sie machen!

$$m = 4, n = 4, s = 4$$

$$\begin{pmatrix} 9 & 3 & 3 & 6 \\ 3 & 5 & 1 & 8 \\ 3 & 5 & 1 & 6 \\ 9 & 3 & 8 & 3 \end{pmatrix} \begin{pmatrix} 1 & 8 & 8 & 1 \\ 5 & 0 & 5 & 4 \\ 4 & 0 & 5 & 4 \\ 9 & 5 & 8 & 4 \end{pmatrix} = \begin{pmatrix} 90 & 102 & 150 & 57 \\ 104 & 64 & 118 & 59 \\ 86 & 54 & 102 & 51 \\ 83 & 87 & 151 & 65 \end{pmatrix}$$

```
[ 9 3 3 6   [1 8 8 1
3 5 1 8  + 5 0 5 4
3 5 1 6   4 0 5 4
9 3 8 3]   9 5 8 4]

=
```

und

```
[ 9 3 3 6   [1 8 8 1
3 5 1 8  * 5 0 5 4
3 5 1 6   4 0 5 4
9 3 8 3]   9 5 8 4]
```

ok, und die anzahl der schritte mitbedenken, da kurz formuliert.

```
a_{11} + b_{11} = c_{11}
a_{12} + b_{12} = c_{12}
```

ok, es sind n x m schritte als NP Problem. wegen n x m
falsch die gesamtzahl aller schritte sind, matrix kann eine lineare
menge sein, sie selber ist mit oder ohne zeile und spalte einfach
eine sagen wir ueber RS232 V.24 gemultiplexte serielle datenmenge:

```
9 3 3 6 3 5 1 8 3 5 1 6 9 3 8 3
```

so. das ist die datenmenge und nun kann man nicht mal vom doppelten der schrittte ausgehen
weil das ist ja ein schritt beim algo.

```
1 8 8 1 5 0 5 4 4 0 5 4 9 5 8 4
```

so nun addieren wir beide:

```
9 3 3 6 3 5 1 8 3 5 1 6 9 3 8 3
+
1 8 8 1 5 0 5 4 4 0 5 4 9 5 8 4
```

um es nicht zu kompliziert zu halten mache ich jetzt nicht, zwischen alle
eine summe, blos in der ziele mit plus

```
9 3 3 6 3 5 1 8 3 5 1 6 9 3 8 3
+ + + + + + + + + + + + + + + +
1 8 8 1 5 0 5 4 4 0 5 4 9 5 8 4
```

so und folgendes, zaehlen wir die plus nach und das ist moeglich MMX und so weiter.

so und unser Plus ist jetzt eine Element beim Algorithmus! beim Algorithmus, wie die zahl
bei der menge der attribute. das ist allerdings keine verdopplungs. die zwei operanten!

sondern! es ist dieselbe menge an Plus!

n:n

1:1 so zu sagen, abbgebildet!

ok: bei der matrizenmultiplikation sieht es so aus:

[9 3 3 6   [1 8 8 1
3 5 1 8 * 5 0 5 4
3 5 1 6   4 0 5 4
9 3 8 3]   9 5 8 4]

9*1 + 3*5 + 3*4 + 6*9
ok: hier ist die anzahl der schritte, wir muessten gucken. rechnen wir nach wie viele
plus haben wir? insgesamt * 3
und wie viele additionen *4

also, hatten wir vorher: 16 operationen haben wir jetzt
16*12 operationen!

das ist in jedem falle NP und P Laufzeit!

jetzt ist das ein problem, was man nicht vollst"andig im griff hat, hier an der stelle wo
addieren wir erst mal und multiplizieren.

[9 3 3 6   [1 8 8 1
3 5 1 8 + 5 0 5 4
3 5 1 6   4 0 5 4
9 3 8 3]   9 5 8 4]

[9+1 3+8

alltaegliche koennen wir loesen: sie reden ueber tiefen und breitensuche und nicht staendi
gerechtigkeit politische themen, verhalten sonst nutzt das ihrem verhalten nichts! sie koe
sagen, und das koennen sie im alltag loesen

tiefensuche sagen und breitensuche sagen! und das kann man loesen, ansonsten kann man allt
probleme kaffee raus lassen, die ihrem verhaltenstechnischen sinne nicht geloest. da sie d
brauchen sie keinen kopf machen! (1.) zweitens: ist das alltag! da gibt es eine loesung! t
reden sie nur tiefen und breitensuche, dann kommt nur tiefen und breitensuche raus. so jet

und verhaltensforscher machen dass dann bitte so: dass sie nicht zu den leuten gehen, sond
klaeren was eine frage des verhaltens ist und mich nicht betrifft.
medizin fragen, bitte keine leute ran nehmen, sondern allgemein klaeren und zwar im buch,
notwendigkeit bitte bleiben lassen! ich bin kein verhalten und bin kein verhaltensforscher
egal, solange noch was raus kommt! zitieren sie mich nicht ein! sie reden was ordentliches

das sind alte leute, die zieht man nicht! und bindet sie an den karren! sondern hier spiel

darueber brauchen sie nicht diskutieren weil sie ueber verhalten reden! dann reden sie tie
problem, was sie haben! nicht wegen zigaretten.

[9 3 3 6   [1 8 8 1
3 5 1 8 + 5 0 5 4
3 5 1 6   4 0 5 4

```
9 3 8 3]   9 5 8 4]

[9+1 3+8 3+8 6+1
3+8 5+0 1+5 8+4
3+4 5+0 1+5 4+6
9+9 3+5 8+8 3+4]

[10 11 11 7
11 5 6 12
7 5 6 10
12 8 16 7]

[ 9 3 3 6   [1 8 8 1
3 5 1 8  * 5 0 5 4
3 5 1 6   4 0 5 4
9 3 8 3]   9 5 8 4]

ZEILE1,  ROW1
9*1 + 3*5 + 3*4 + 6*9  COL1
9*8 + 3*0 + 3*0 + 6*5. COL2
9*8 + 3*5 + 3*5 + 6*8. ...
9*1 + 3*4 + 3*4 + 6*4

ROW2
3*1 + 5*5 + 1*4 + 9*9
3*8 + 5*0 + 1*0 + 8*5
3*8 + 5*0 + 1*5 + 8*8
3*1 + 5*4 + 1*4 + 8*4

ROW3
3*1 + 5*5 + 1*4 + 6*9
3*8 + 5*0 + 1*0 + 6*5
3*8 + 5*5 + 1*5 + 6*8
3*1 + 5*4 + 1*4 + 6*4

ROW4
9*1 + 3*5 + 8*4 + 3*9
9*8 + 3*0 + 8*0 + 3*5
9*8 + 3*5 + 8*5 + 3*8
9*1 + 3*4 + 8*4 + 3*4

hilfsschrieb, wg. bildschirm groesse
[ 9 3 3 6   [1 8 8 1
3 5 1 8  * 5 0 5 4
3 5 1 6   4 0 5 4
9 3 8 3]   9 5 8 4]


zusammenrechnen. danach anzahl der schritte zaehle, algo...
```

```
ZEILE1,  ROW1
9*1 + 3*5 + 3*4 + 6*9 = 9 + 15 + 12 + 54COL1
9*8 + 3*0 + 3*0 + 6*5 = 72 + 30 = COL2
9*8 + 3*5 + 3*5 + 6*8 = 72 + 15 + 15 + 48 =
9*1 + 3*4 + 3*4 + 6*4 = 9 + 12 + 12 + 24 =

# schritte trennen anmerkung mit gartenzaun #
# verfuehrung verfuehrerisch in mathe widerstehen
# die dinger zusammenrechnen im 2. schritt
# process auftakt, schritte richtig benenne
# das ist nicht der 2. schritt, sondern der +2. weil
# davor auch schon. so jetzt weiter.

ROW2
3*1 + 5*5 + 1*4 + 9*9 = 3 + 25 + 4 + 81
3*8 + 5*0 + 1*0 + 8*5 = 24 + 0 + 0 40
# nicht verfuehererisch heisst,
# 1.) selbst wenn anmerkung in der sache schlecht aussieht
# trotzdem an die richtige stelle
# 2.) mathematiik ist esthetisch, nicht aufgeschriebene mathematik
# ist keine. die estehtik vertut man sich entsthet nicht deswgen
# weil der kommentar nihct da ist, am ende sieht man
# beim letzten habe ich nullen vergessen, nullen hier gehoerne
# bei der addition dazu, bei der summe, die kommsn von alleine raus
# da aber beim letzten mal weggelassen, gehe ich nicht im text
# und mache das besser. vergleich computer und spaziergaenger
# ein spaziergaenger laesst alles moegliche liegen und wird trotz
# allem aeusseren nicht perfekt, das ist allerdings, also die letzten
# nicht rein, hinten, im text am anfang, sondern:  hier rein machen.
# bei der summe, genug geschwatzt.
3*8 + 5*0 + 1*5 + 8*8 = 24 + 0 + 5 + 64
3*1 + 5*4 + 1*4 + 8*4 = 3 + 20 + 4 + 32

ROW3
3*1 + 5*5 + 1*4 + 6*9 = 3 + 25 + 4 + 54
3*8 + 5*0 + 1*0 + 6*5 = 24 + 0 + 0 + 30
3*8 + 5*5 + 1*5 + 6*8 = 24 + 25 + 5 + 48
3*1 + 5*4 + 1*4 + 6*4 = 3 + 20 + 4 + 24

ROW4
9*1 + 3*5 + 8*4 + 3*9 = 9 + 15 + 32 + 27
9*8 + 3*0 + 8*0 + 3*5 = 72 + 0 + 0 + 15
9*8 + 3*5 + 8*5 + 3*8 = 72 + 15 + 40 + 24
9*1 + 3*4 + 8*4 + 3*4 = 9 + 12 + 32 + 12

ok, jetzt naechster schritt, die summen bilden

ZEILE1,  ROW1
9*1 + 3*5 + 3*4 + 6*9 = 9 + 15 + 12 + 54 = 24 + 66 = 90
9*8 + 3*0 + 3*0 + 6*5 = 72 + 30 = 102
```

```
9*8 + 3*5 + 3*5 + 6*8 = 72 + 15 + 15 + 48 = 120 + 30 = 150
9*1 + 3*4 + 3*4 + 6*4 = 9 + 12 + 12 + 24 =  48 + 9 = 57

ROW2
3*1 + 5*5 + 1*4 + 9*9 = 3 + 25 + 4 + 81 = 32 + 81 = 3 + 110 = 113
3*8 + 5*0 + 1*0 + 8*5 = 24 + 0 + 0 40 = 64
3*8 + 5*0 + 1*5 + 8*8 = 24 + 0 + 5 + 64 = 93
3*1 + 5*4 + 1*4 + 8*4 = 3 + 20 + 4 + 32 = 59

ROW3
3*1 + 5*5 + 1*4 + 6*9 = 3 + 25 + 4 + 54 = 61 + 25 = 86
3*8 + 5*0 + 1*0 + 6*5 = 24 + 0 + 0 + 30 = 54
3*8 + 5*5 + 1*5 + 6*8 = 24 + 25 + 5 + 48 = 48 + 1 + 48 = 97
3*1 + 5*4 + 1*4 + 6*4 = 3 + 20 + 4 + 24 = 28 + 23 = 51

ROW4
9*1 + 3*5 + 8*4 + 3*9 = 9 + 15 + 32 + 27 = 47 + 27 + 9 = 7 + 7 + 9 = 60 = 83
9*8 + 3*0 + 8*0 + 3*5 = 72 + 0 + 0 + 15 = 87
9*8 + 3*5 + 8*5 + 3*8 = 72 + 15 + 40 + 24 = 70 + 10 + 40 + 20 + 7 + 4 = 140 + 11 = 151
9*1 + 3*4 + 8*4 + 3*4 = 9 + 12 + 32 + 12 = 56 + 9 = 65

# so gut.
# jetzt schreiben wir das noch mal hin und befreien es vom ausdruck und schreiben
# ordentlich zusammen


ZEILE1,  ROW1
90  102.  150   57

ROW2
113  64   93.  59

ROW3
86  54   97  51

ROW4
83   87   151   65

# so, jetzt machen wir da eine ordentliche matrix draus, vergleichen das mit
# dem ergebnis und wenn das passiert ist, dann machen wir das genauso
# wie im computer moeglich. und zwar so: dass die + und * stehen bleiben
# und dann kann man die anzahl der operationen von alleine ausrechnen.

[ 90  102.  150   57
113  64   93.  59
86  54   97  51
83   87   151   65]


ROW3
```

```
3*1 + 5*5 + 1*4 + 6*9 = 3 + 25 + 4 + 54
3*8 + 5*0 + 1*0 + 6*5 = 24 + 0 + 0 + 30
3*8 + 5*5 + 1*5 + 6*8 = 24 + 25 + 5 + 48
3*1 + 5*4 + 1*4 + 6*4 = 3 + 20 + 4 + 24

ROW4
9*1 + 3*5 + 8*4 + 3*9 = 9 + 15 + 32 + 27
9*8 + 3*0 + 8*0 + 3*5 = 72 + 0 + 0 + 15
9*8 + 3*5 + 8*5 + 3*8 = 72 + 15 + 40 + 24
9*1 + 3*4 + 8*4 + 3*4 = 9 + 12 + 32 + 12

ok, jetzt naechster schritt, die summen bilden

ZEILE1,  ROW1
9*1 + 3*5 + 3*4 + 6*9 = 9 + 15 + 12 + 54 = 24 + 66 = 90
9*8 + 3*0 + 3*0 + 6*5 = 72 + 30 = 102
9*8 + 3*5 + 3*5 + 6*8 = 72 + 15 + 15 + 48 = 120 + 30 = 150
9*1 + 3*4 + 3*4 + 6*4 = 9 + 12 + 12 + 24 =  48 + 9 = 57

ROW2
3*1 + 5*5 + 1*4 + 9*9 = 3 + 25 + 4 + 81 = 32 + 81 = 3 + 110 = 113
# hier ist ein fehler: da muss 8*9 rein
# also: 3 + 29 + 72 = 75 + 29 = 104
# jetzt richtig
3*8 + 5*0 + 1*0 + 8*5 = 24 + 0 + 0 40 = 64
3*8 + 5*5 + 1*5 + 8*8 = 24 + 25 + 5 + 64 = 30 + 24 + 64 = 118
# hier stimmt was nicht
# das ist richtig

3*1 + 5*4 + 1*4 + 8*4 = 3 + 20 + 4 + 32 = 59

ROW3
3*1 + 5*5 + 1*4 + 6*9 = 3 + 25 + 4 + 54 = 61 + 25 = 86
3*8 + 5*0 + 1*0 + 6*5 = 24 + 0 + 0 + 30 = 54
3*8 + 5*5 + 1*5 + 6*8 = 24 + 25 + 5 + 48 = 24 + 25 + 5 + 48 = 30 + 24 + 48 = 102
# ist korrekt
3*1 + 5*4 + 1*4 + 6*4 = 3 + 20 + 4 + 24 = 28 + 23 = 51

ROW4
9*1 + 3*5 + 8*4 + 3*9 = 9 + 15 + 32 + 27 = 47 + 27 + 9 = 7 + 7 + 9 = 60 = 83
9*8 + 3*0 + 8*0 + 3*5 = 72 + 0 + 0 + 15 = 87
9*8 + 3*5 + 8*5 + 3*8 = 72 + 15 + 40 + 24 = 70 + 10 + 40 + 20 + 7 + 4 = 140 + 11 = 151
9*1 + 3*4 + 8*4 + 3*4 = 9 + 12 + 32 + 12 = 56 + 9 = 65

ok, jetzt passt die sache.


[ 90  102.  150   57
113  64   118  59
86  54   102  51
```

```
83   87   151   65]

ok, das passt. jetzt weiter.

ZEILE1,  ROW1
9*1 + 3*5 + 3*4 + 6*9 = 9 + 15 + 12 + 54 = 24 + 66 = 90
9*8 + 3*0 + 3*0 + 6*5 = 72 + 30 = 102
9*8 + 3*5 + 3*5 + 6*8 = 72 + 15 + 15 + 48 = 120 + 30 = 150
9*1 + 3*4 + 3*4 + 6*4 = 9 + 12 + 12 + 24 =  48 + 9 = 57

ROW2
3*1 + 5*5 + 1*4 + 9*9 = 3 + 25 + 4 + 81 = 32 + 81 = 3 + 110 = 113
3*8 + 5*0 + 1*0 + 8*5 = 24 + 0 + 0 40 = 64
3*8 + 5*0 + 1*5 + 8*8 = 24 + 0 + 5 + 64 = 93
3*1 + 5*4 + 1*4 + 8*4 = 3 + 20 + 4 + 32 = 59

ROW3
3*1 + 5*5 + 1*4 + 6*9 = 3 + 25 + 4 + 54 = 61 + 25 = 86
3*8 + 5*0 + 1*0 + 6*5 = 24 + 0 + 0 + 30 = 54
3*8 + 5*5 + 1*5 + 6*8 = 24 + 25 + 5 + 48 = 48 + 1 + 48 = 97
3*1 + 5*4 + 1*4 + 6*4 = 3 + 20 + 4 + 24 = 28 + 23 = 51

ROW4
9*1 + 3*5 + 8*4 + 3*9 = 9 + 15 + 32 + 27 = 47 + 27 + 9 = 7 + 7 + 9 = 60 = 83
9*8 + 3*0 + 8*0 + 3*5 = 72 + 0 + 0 + 15 = 87
9*8 + 3*5 + 8*5 + 3*8 = 72 + 15 + 40 + 24 = 70 + 10 + 40 + 20 + 7 + 4 = 140 + 11 = 151
9*1 + 3*4 + 8*4 + 3*4 = 9 + 12 + 32 + 12 = 56 + 9 = 65

jetzt zaehlen wir die operationen.

ZEILE1,  ROW1
9*1 + 3*5 + 3*4 + 6*9 = 9 + 15 + 12 + 54 = 24 + 66 = 90
9*8 + 3*0 + 3*0 + 6*5 = 72 + 30 = 102
9*8 + 3*5 + 3*5 + 6*8 = 72 + 15 + 15 + 48 = 120 + 30 = 150
9*1 + 3*4 + 3*4 + 6*4 = 9 + 12 + 12 + 24 =  48 + 9 = 57

ROW2
3*1 + 5*5 + 1*4 + 9*9 = 3 + 25 + 4 + 81 = 32 + 81 = 3 + 110 = 113
3*8 + 5*0 + 1*0 + 8*5 = 24 + 0 + 0 40 = 64
3*8 + 5*0 + 1*5 + 8*8 = 24 + 0 + 5 + 64 = 93
3*1 + 5*4 + 1*4 + 8*4 = 3 + 20 + 4 + 32 = 59

ROW3
3*1 + 5*5 + 1*4 + 6*9 = 3 + 25 + 4 + 54 = 61 + 25 = 86
3*8 + 5*0 + 1*0 + 6*5 = 24 + 0 + 0 + 30 = 54
3*8 + 5*5 + 1*5 + 6*8 = 24 + 25 + 5 + 48 = 48 + 1 + 48 = 97
3*1 + 5*4 + 1*4 + 6*4 = 3 + 20 + 4 + 24 = 28 + 23 = 51

ROW4
9*1 + 3*5 + 8*4 + 3*9 = 9 + 15 + 32 + 27 = 47 + 27 + 9 = 7 + 7 + 9 = 60 = 83
```

```
9*8 + 3*0 + 8*0 + 3*5 = 72 + 0 + 0 + 15 = 87
9*8 + 3*5 + 8*5 + 3*8 = 72 + 15 + 40 + 24 = 70 + 10 + 40 + 20 + 7 + 4 = 140 + 11 = 151
9*1 + 3*4 + 8*4 + 3*4 = 9 + 12 + 32 + 12 = 56 + 9 = 65

* + * + * + *, |+| = 3, |*| = 4
* + * + * + *, |+| = 3, |*| = 4
* + * + * + *, |+| = 3, |*| = 4
* + * + * + *, |+| = 3, |*| = 4
12 + 16 = 28
# das doppelte an operationen sind:
# 32 waere betracht der elemente, gehen wir zurueck zur addition
# haben wir 16
# weil wir 16 dinger zusammen addieren.
# muessen wir gucken.

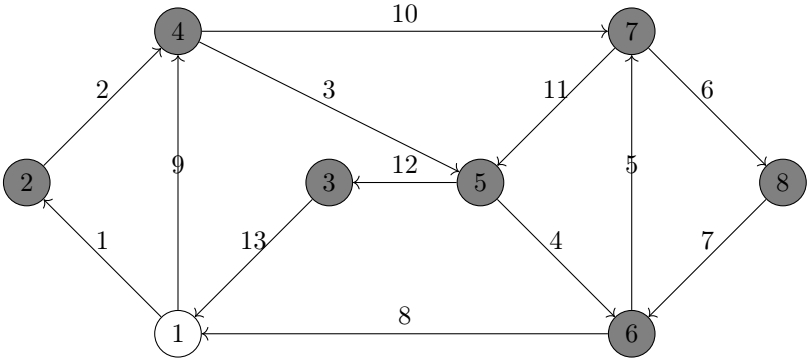
# das jetzt in ordentlicher mathematischer form.
```

so ein paar ergebnisse sind falsch.

$$m = 4, n = 4, s = 4$$

$$\begin{pmatrix} 9 & 3 & 3 & 6 \\ 3 & 5 & 1 & 8 \\ 3 & 5 & 1 & 6 \\ 9 & 3 & 8 & 3 \end{pmatrix} + \begin{pmatrix} 1 & 8 & 8 & 1 \\ 5 & 0 & 5 & 4 \\ 4 & 0 & 5 & 4 \\ 9 & 5 & 8 & 4 \end{pmatrix} = \begin{pmatrix} 10 & 11 & 11 & 7 \\ 8 & 5 & 6 & 12 \\ 7 & 5 & 6 & 10 \\ 18 & 8 & 16 & 7 \end{pmatrix}$$

so, jetzt haben wir das so weit, und probieren es weiter zu tun. nämlich da, wo wir gestern waren mit tiefen und breitensuche




```
jetzt machen wir das ohne namen. das heisst, wir ersparen und
component, und ersparen uns alle namen der mengen. und so weiter.
sondern machen folgendes:

wir merken uns die knoten und notieren uns irgendwas.

1
2,3,4

jetzt merken wir uns:
dass 1 abgearbeitet wurden, dass seine nachbarn
```

2, 3, 4 sind und nehmen die 2

1,2
wir sind bei: 2
2: 1,4

1 und 4 haben wir schon, also kommt:

1,2,3,4
und 1,2
und kommt: 3

3: 1,5
1 haben wir schon
3 hat: 5
1,2,3,4,5
bearbeitet wurden: 1, 2, 3

als naechstes kommt:
4 weil es so auf der schlange steht:
oder rekursiv tiefensuche jetzt 3:

4: 5, 7
1, 2, 3, 4, 5, 7

und bearbeitet sind
1, 2, 3, 4

und weiter: als naechstes kommt von der schlange
5: 5 hat: 7, 8
1, 2, 3, 4, 5, 7, 8
und 1, 2, 3, 4, 5

als naechstes kommt 7:
7 hat:

4, 8 sind bereits
1,2, 3,4,5,7
als naechstes kommt 7
7 hat: 6,8
1,2,3,4,5,7
als naechstes:
6,
und: 1,2,3,4,5,7,6,8
1,2,3,4,5,7,6
6 hat: 1,5,7,8
als naecheste kommt: 8
1,2,3,4,5,7,6,7,8

so, gut: und jetzt ist die breitensuche vollstaendig, wir ueberlegen es uns noch mal neu

probehalber und machen eine tiefen und breitensuche im naechsten schritt notieren wir name

so, jetzt noch mal auf eine ordentliche art und weise.

```
bei null ist auf der schlange: 1

jetzt haben wir nummer 1:
jetzt ist auf der schlange
1,2,3,4
als besucht markiert ist:
1,2,3,4
als: komponenten markiert/zusammenhaengender graph sind:
1
und die schlange ist wiederum:
2,3,4 wg. 1 runter genommen
als naechstes kommt die 2
die 2 hat, als nachbarn 4, 1
diese sind als benutzt markiert worden.
so, die 2 kommt runter:
3,4 ist auf der schlange,
komponente ist: 1,2
naechstes:
3, 4
die nachbarn von 3 sind: 1,5
1 ist drauf und 5 wird genommen.
und jetzt wird als benutzt markiert
1,2,3,4,5
ok: und jetzt weiter.
jetzt auf der schlange sind
4,5
und: als komponente sind
1,2,3,4
jetzt merkt man besser als kanten drauf tun. also:
die komponente kann man nachher herstellen, egal weil
die kommen von der schlange vorne runter und dann muss
man sich keine gedanken machen.

jetzt kommt 4 runter:
nachbarn von 4 sind: 1, 2, 5, 7
dann kommt auf die schlange:

4 runter, 5, drauf 7
5,7
so jetzt kommt:
1,2,3,4,5
jetzt kommt 7
1,2,3,4,5,7
die nachbarn von 7 sind:
8, 5, ist schon
1,2,3,4,5,7,8 sind markiert als benutzt.
1,2,3,5,7,8
```

auf der schlange ist:
7,8 sieben runter:
jetzt kommt die 6
1,2,3,4,5,7,8,6

so jetzt ist die komponente so und jetzt suchen wir der
reihenfolge nach, die kanten die die verbinden

(1,2),(2,4),(1,3),(4,5),(5,7),(5,6),(7,8)

jetzt merken wir uns:
dass 1 abgearbeitet wurden, dass seine nachbarn
2, 3, 4 sind und nehmen die 2

1,2
wir sind bei: 2
2: 1,4

1 und 4 haben wir schon, also kommt:

1,2,3,4
und 1,2
und kommt: 3

3: 1,5
1 haben wir schon
3 hat: 5
1,2,3,4,5
bearbeitet wurden: 1, 2, 3

als naechstes kommt:
4 weil es so auf der schlange steht:
oder rekursiv tiefensuche jetzt 3:

4: 5, 7
1, 2, 3, 4, 5, 7

und bearbeitet sind
1, 2, 3, 4

und weiter: als naechstes kommt von der schlange
5: 5 hat: 7, 8
1, 2, 3, 4, 5, 7, 8
und 1, 2, 3, 4, 5

als naechstes kommt 7:
7 hat:

```
4, 8 sind bereits
1,2, 3,4,5,7
als naechstes kommt 7
7 hat: 6,8
1,2,3,4,5,7
als naechstes:
6,
und: 1,2,3,4,5,7,6,8
1,2,3,4,5,7,6
6 hat: 1,5,7,8
als naecheste kommt: 8
1,2,3,4,5,7,6,7,8

so, gut: und jetzt ist die breitensuche vollstaendig, wir ueberlegen es uns noch mal neu
probehalber und machen eine tiefen und breitensuche im naechsten schritt notieren wir name
```

jetzt haben wir irgendwie die erste komponente mit breitensuche. wir können probieren total richtig oder falsch, eine tabelle mit LaTeXdraus zu machen. einigermassen richtig. der unterschied, rekursion und schlange, die rekursion merkt sich das auch, nimmt den nächsten knoten, jetzt tabelle

$n$te Schritt	Anfang(Schlange)	Schlange	Markiert	Komponente	Nachbarn (Anfang(Schlange))
0	x	(1)	x	(1)	x
$(0,1)$	1	$(1,2,3,4)$	(1)	(1)	$(2,3,4)$
$(0,2)$	1	$(2,3,4)$	(1)	(1)	$(2,3,4)$
1	1	$(2,3,4)$	$(1,2)$	$(1,2)$	$(2,3,4)$
$(1,)$	2	$(2,3,4)$	$(1,2)$	$(1,2)$	$(1,4)$
$(1,)$	2	$(3,4)$	$(1,2)$	$(1,2)$	$(1,4)$
2	3	$(3,4)$	$(1,2,3)$	$(1,4)$	
3	4	4	$(1,2,3,4)$	$(2,5)$	
3	4	4	$(1,2,3,4)$	$(2,5)$	

so, jetzt haben wir das hier fertig. dann haben wir N und NP vollstaendigkeit bereits untersucht und jetzt machen wir hier weiter. das erste, was wir jetzt machen. ist. dass wir den automaten zum suchen von zeichenfolgen weitermachen. so: das heisst also: binäre zeichenfolge, text-zeichenfolge. und das geht so jetzt. erst mal nehmen wir den automaten von neulich, den wir hoffentlich noch im kopf haben. wir nehmen statt

```
ABCD
ABACD
```

nehmen wir die binären zeichen zwei bit lang, baudot code, heisst: 1 bit, 2 bit, ... 8 bit zum beispiel aber

```
rs232
v.24
```

heisst, zwischen 5 und 9 bit. jeder zeichensatz enthaelt, diese könnene ein bit sein, oder 2 oder so. also der zeichenvorrat in einem zeichensatz. deswegen ist zwischen binärer zeichenfolge und text zeichenfolge zu unterscheiden. und wie gesagt, baudat

```
digits/sign
```

unterscheidung und baud-rate

zeichen pro sekunde

das ist so zu sagen die baudrate, die geschwindigkeit mit der zeichen eines zeichenvorrats

trotzdem gibt es noch die anzahl der bit so zu sagen, pro zeichen

das entspricht allerdings der Kardinalitaet, also der Maechtigkeit des Zeichenvorrats, sel
keine eindeutige ansage. weil ein chinesisches zeichen, besteht aus vielen strichen, ebens

0410422000
1f2c64f0f04

irgendwie so. allerdings muessen wir wissen. zum beispiel bei 256 brauchen wir generell:
8 bit. bcd, das macht die sieben anzeige, sind eben trotzdem kein 7 seg code
ebenso wie 1 aus 10. also: beispiele. und jetzt:

jetzt gehen wir diese folge durch

ABCD
ABACD

und jetzt ist:

A: 0b00
B: 0b01
C: 0b10
D: 0b11

ok, dann ist ABACD

0b00,0b01,0b00,0b10,0b11

und das heisst, gepackt und ungepackt, als nibble oder byte oder wort

0b0001.0010.11

zum beispiel oder so
0b00.01.00.10.11

oder ebenso gepackt und ungepackt:

0b0000.0001.0000.0010.0011
ok: das ist das. das geht noch weiter. gepackt, kommt raus:
0b00.001.00.10.11
aber was ist mit dem letzten weil es sind trotzdem vier bit, das heist
0b00001.0010.11 xx
0b00001.0010.1100

ok, zum beispiel oder besser vorne, start zeichen, und stop zeichen. das ist hier schwieri

```
zum Beispiel bei Rechte
0x000 -----
0b001 r---x
0b010 --r--
...
0b111 rwx

ok, drei bit akzeptabler code, heisst:
0b000.0001.000.010.011.100 oder
0b000.0001.000.010.011.111

so, jetzt ist die sache so weit, ist trotzdem besser, dann vier bit zu nehmen: wie BCD AIK

0b0000.00001.0000.0010.0011.1000 oder (?) das allerdings in BCD Vorhanden ist lieber
0b0000.00001.0000.0010.0011.1010 weil das ist am ende 10D und das gibt es so nicht in BCD
0b0000.00001.0000.0010.0011.11111

jetzt weiter.

ABACD

jetzt muessen wir, den automat kenne ich noch auswendig der geht so:

auf der einen seite: ACD
auf der anderen seite AB*A

das heisst, weil wir hier bei B eine wiederholung haben
und das andere, da beginnt die Spaltung bei CD
so ungefaehr, damit koennen wir den malen und machen dazu
noch das selbe mit 0b00, 0b01, 0b10, 0b11 also tabelle.
```

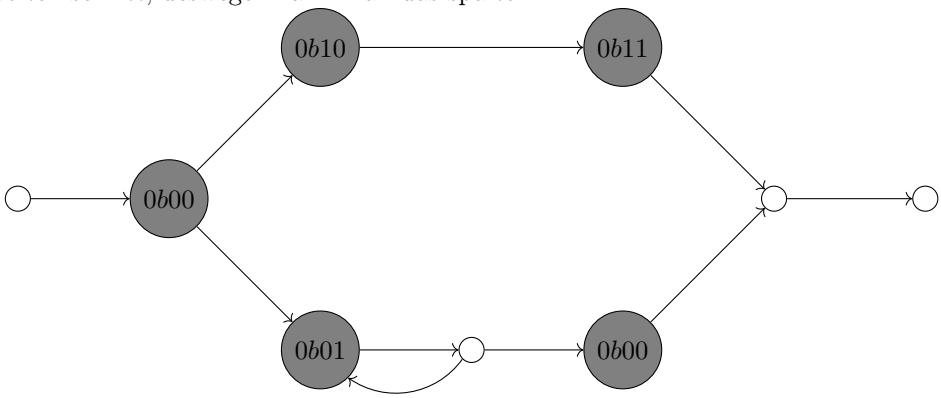






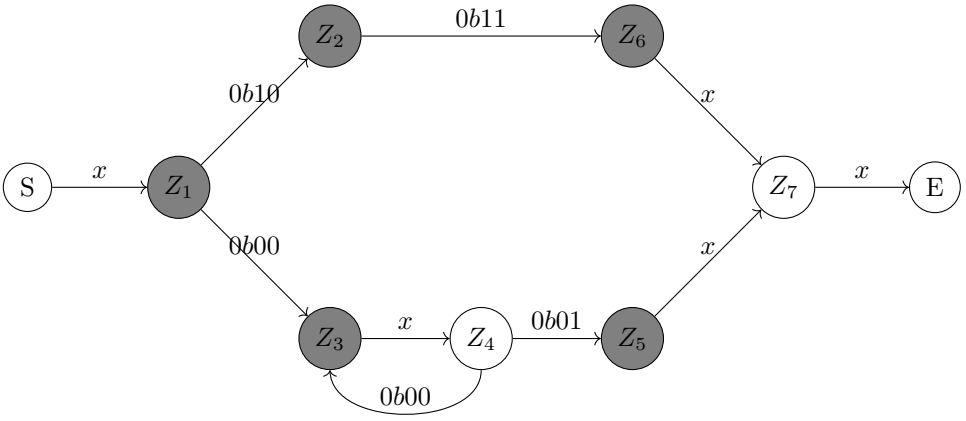
und hier möchte ich hinweisen, gibt es eben zwei mittel: nämlich folgende. das eine ist. so zu sagen, der nicht deterministische automat, das sind natürlich beide. nur: der eine fängt gnadenlos mit a immer an also $0b00$ und da muss überlegen sind da in dem fehler drin. das ist das eine. der andere macht bei so zu sagen nach beider unterscheiden, beim einen a und beim anderen auch. trotzdem erst den. der unterschied. sagen wir. liegt an anderer stelle, weil eigentlich müsste bei WDH von a das Zeichen a mindestens ein mal kommen. trotzdem kann das kein mal kommen und dann müsste so zu sagen,

die WDH wo anders. im zweiten schritt, deswegen kann man das spalten



dann kommt noch mal ein a

wir könnten das ganze getrost in einem mealy automaten umwandeln. nämlich wenn die eingabe passt, kommt wir in den nächsten zustand



jetzt probieren wir dieses muster aus. anhand, stecken es in eine tabelle. wenn nämlich mealy. nehmen die eingabe als stream - lineare folge von zeichen mit zwei binären digitalen bits und prüfen in welchem zustand wir sind.

```
0b00.00.00.00.00.01.01.01
0b00.00.00.01.01.00.00.00.01.11
0b10.11.00.00.00
0b00.00.11.11.11
0b00.00.00.00.01
0b00.01.00.01.00.01.00.01.00.01
```

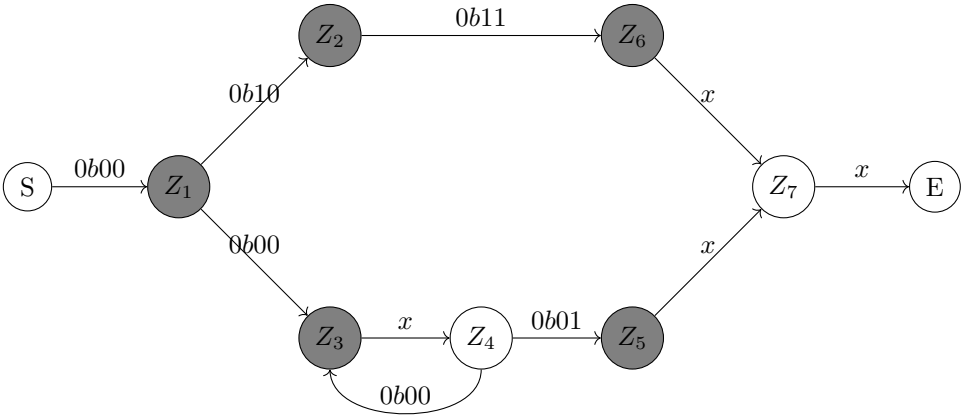
ok, mit tabelle

x	Z	Z^+
x	S	Z_1
$0b00$	Z_1	Z_3
x	Z_3	Z_4
$0b00$	Z_4	Z_3
x	Z_3	Z_4
$0b00$	Z_4	Z_3
x	Z_3	Z_4
$0b00$	Z_4	Z_3
x	Z_3	Z_4
$0b00$	Z_3	Z_3
x	Z_3	Z_4
$0b01$	Z_4	Z_5
x	Z_5	Z_7
x	Z_7	E

das passte. jetzt mal, jetzt kommt das nächste.
wir lassen es dabei und schreiben die grammatik auf, nachdem wir das getan haben, erweitern wir sie.

```
(A*B)|(CD)
# das war die grammatik und ich denke, sie war nicht die von Robert sedgewick, sondern die
A(A*B)|(CD)
was vermutlich das gleiche ist, wie - ich habe das buch hier liegen, aber:
ich gucke jetzt nicht nach.
(A*B)|(ACD)
```

dann können wir: mit Mealy Z_1 rein zu kommen, statt x am anfang, mit: Z_1



gucken wir noch mal nach wie es wirklich ist. als erstes machen wir folgendes:
wir nehmen ein feld/array

```
int state;
int ch []
int state1 [];
int state2 [];

state1 [];

# ok, hier lassen wir es hier, scheitert es mit mealy und moore. ich rechne jetzt weiter!
```

lassen wir grammatik, genug getan! jetzt kommt atmega8! und nasm! und i586
eine schriftliche binäre division machen wir auch noch. mit drei bit, als teiler vielleicht

```
0b101
dec(0b101) = 5dec
```

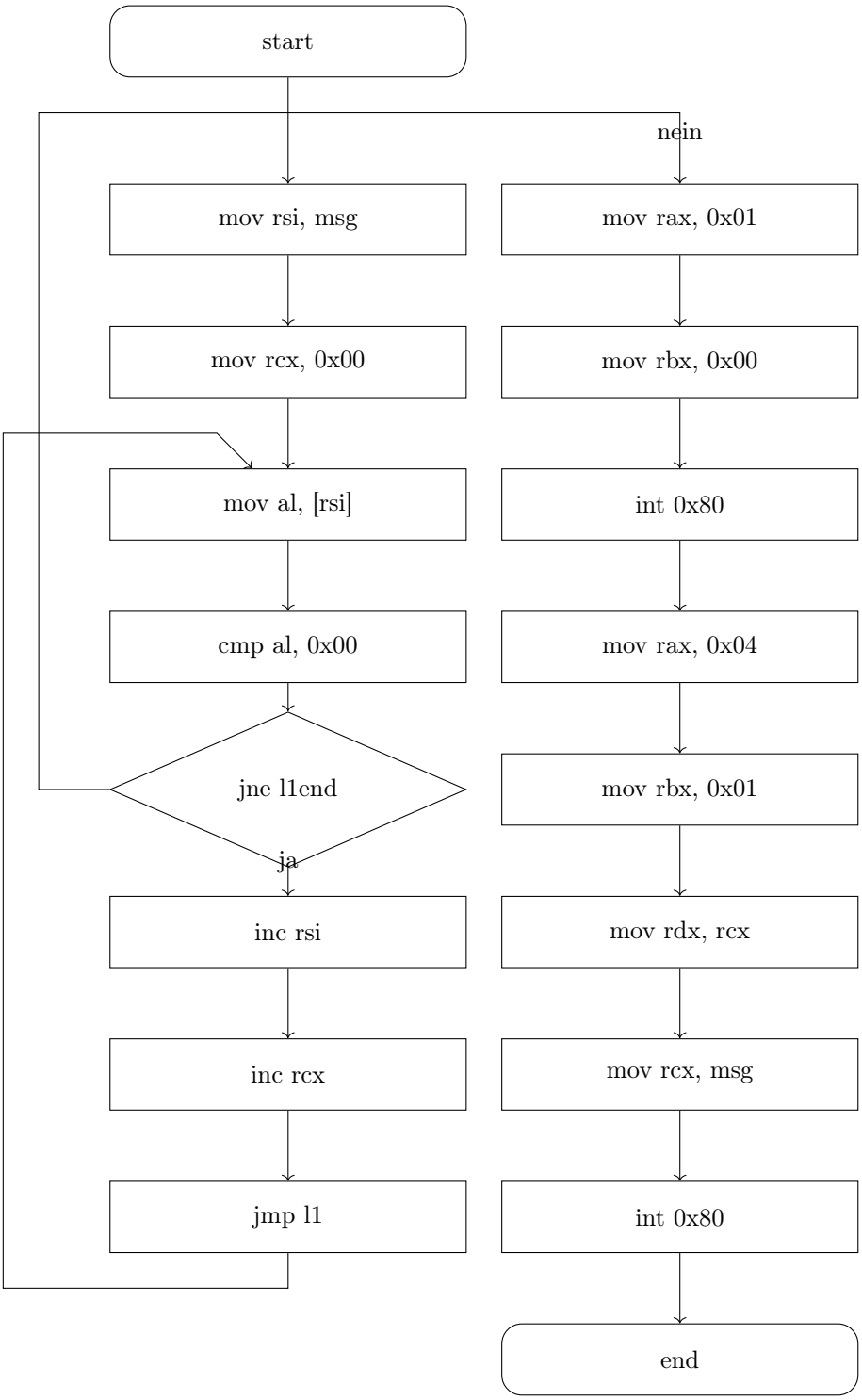
vorher ein nasm/i586/amd64 assembler programm

```
.global _start
section .data
 msg: db "david@www.vajda-breadboard.de", 10, 13, 0
section .text
_start:
 mov  rsi, msg
 mov rcx,  0x00
l1: mov al, [rsi]
 cmp al, 0x00
 je l1end
 inc rsi
 inc rcx
 jmp l1
l1end:

 mov rax, 0x01
 mov rbx, 0x00
 int 0x80

 mov rax, 0x04
 mov rbx, 0x01
 mov rdx, rcx
 mov rcx,  msg
 int  0x80
```

die alternativ idee kann ich mir nicht vorstellen, gerade zu tun, nämlich, dass man: die pruefung am anfang ohne zweites label ausfuehrt, naemlich aus dem grund dass das inkrementieren sonst trotzdem statt findet. jetzt ein asm diagramm dazu.



jetzt teilen wir die zahl binär durch $0b101$. im divisor, nämlich den dividenden, das y-Wort oben.

```
0b0111.1000.1000.0010/0b0101 = 0b1100000001
```

```
0b0101
----------
0b0010.1
0b00101
------------
           0b00
         -0b101 (ohne wrap-arround - also saturation)
         ---------
           0b000
          -0b101
          ---------
          000
         -101
         ---------
         000
        -101
        ---------
            001
           -101
          -------
              10
             -101
            ------
            100
            -101
            ------
            1000
           -0101
           -------
              0011
```

jetzt bin ich durcheinander gekommen, also geht das hier nicht anders. vielleicht in bloecken.

```
0b0111
0b1000
0b1000
0b0010

so jetzt teilen wir das wie bei modulo bei iban in paeckchen - 0b0101
und dann haben wir den uebertrag, dann ist das uebersichtlicher. dann nehmen wir das mit.
machen immer paeckchen weise, immer nibble.

0b0111/0b0101 = 1 und zwar: 0b0111 - 0b0101 = 0b0010

dann kann man das als rest begreifen:

(0b0010 << 4) | 0b1000

0b00101000 / 0b0101 = 0... # dann geht das so weiter.
0b01010.00 / 0b0101 = 0.. # dann geht es weiter.
```

```
0b1010.00 / 0b0101 = 1 .. # dann geht es weiter
0b0000.00
drei weg vorne
0b0.00 / 0b0101 = 0 .. # dann geht es weiter
0b00 / 0b0101 = 0 .. # dann geht es weiter
0b0 / 0b0101 = 0 .. # dann geht es weiter

1001 000 # und jetzt geht es weiter, ohne rest

0b1000 / 0b0101 = 0 (100 - 101)
0b1000 / 0b0101 = 1 Rest 11 (entspricht 1000 - 0101)

110010 / 0b0101 = 1
101 = 1 Rest 1
001 .010 / 0b0101 = 1 # ist 1010 wg. anfang weg 1010
0

10010000110 das koennte raus kommen

0b100.1000.0110
0b0100.1000.0110 Rest 1

und jetzt da als probe ausfuehren:

10 = 2
100 = 4
128 +
nehmen wir den normalen rechner.
```

gerade fehlt konzentration. tue den text in die fussnote, den letzten.

```
(C) David Vajda
Tue Oct 27 10:24:06 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    1
 2 0 0 1 0    0
 3 0 0 1 1    0
 4 0 1 0 0    0
 5 0 1 0 1    0
 6 0 1 1 0    0
 7 0 1 1 1    0
 8 1 0 0 0    1
 9 1 0 0 1    1
10 1 0 1 0    0
11 1 0 1 1    0
12 1 1 0 0    1
13 1 1 0 1    0
14 1 1 1 0    0
```

```
15 1 1 1 1    1
```

so. jetzt weiter:
minimierung:

```
\begin{verbatim}
(C) David Vajda
Tue Oct 27 10:24:06 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    1
 8 1 0 0 0    1
 9 1 0 0 1    1
12 1 1 0 0    1
15 1 1 1 1    1

Gruppe 0:
 0 0 0 0 0    1
Gruppe 1:
 1 0 0 0 1    1
 8 1 0 0 0    1
Gruppe 2:
 9 1 0 0 1    1
12 1 1 0 0    1
Gruppe 4:
15 1 1 1 1    1

(0,1)  0 0 0 x
(0,8)  x 0 0 0
(1,9)  x 0 0 1
(8,9)  1 0 0 x
(8,12) 1 x 0 0
15  1 1 1 1


(0,1)  0 0 0 x
(8,9)  1 0 0 x

(0,1,8,9)  x 0 0 x

(0,8)  x 0 0 0
(1,9)  x 0 0 1

(0,8,1,9)  x 0 0 x

(8,12)   1 x 0 0

15   1 1 1 1
```

```
(0,8,1,9)  x 0 0 x
(8,12)   1 x 0 0
15   1 1 1 1

jetzt so zu sagen: primimplikanten

   0 1 8 9 12 15
(0,8,1,9)  1 1 1 1
(8,12)     1  1
15        1
```

	0	1	8	9	12	15
$(0,8,1,9)$	1	1	1	1		
$(8,12)$			1		1	
15						1

formel:

```
(0,8,1,9)  x 0 0 x
(8,12)  1 x 0 0
15   1 1 1 1

 y = not x2 and not x1 or
   x3 and not x1 and not x0 or
   x3 and x2 and x1 and x0;
```

mit galasm

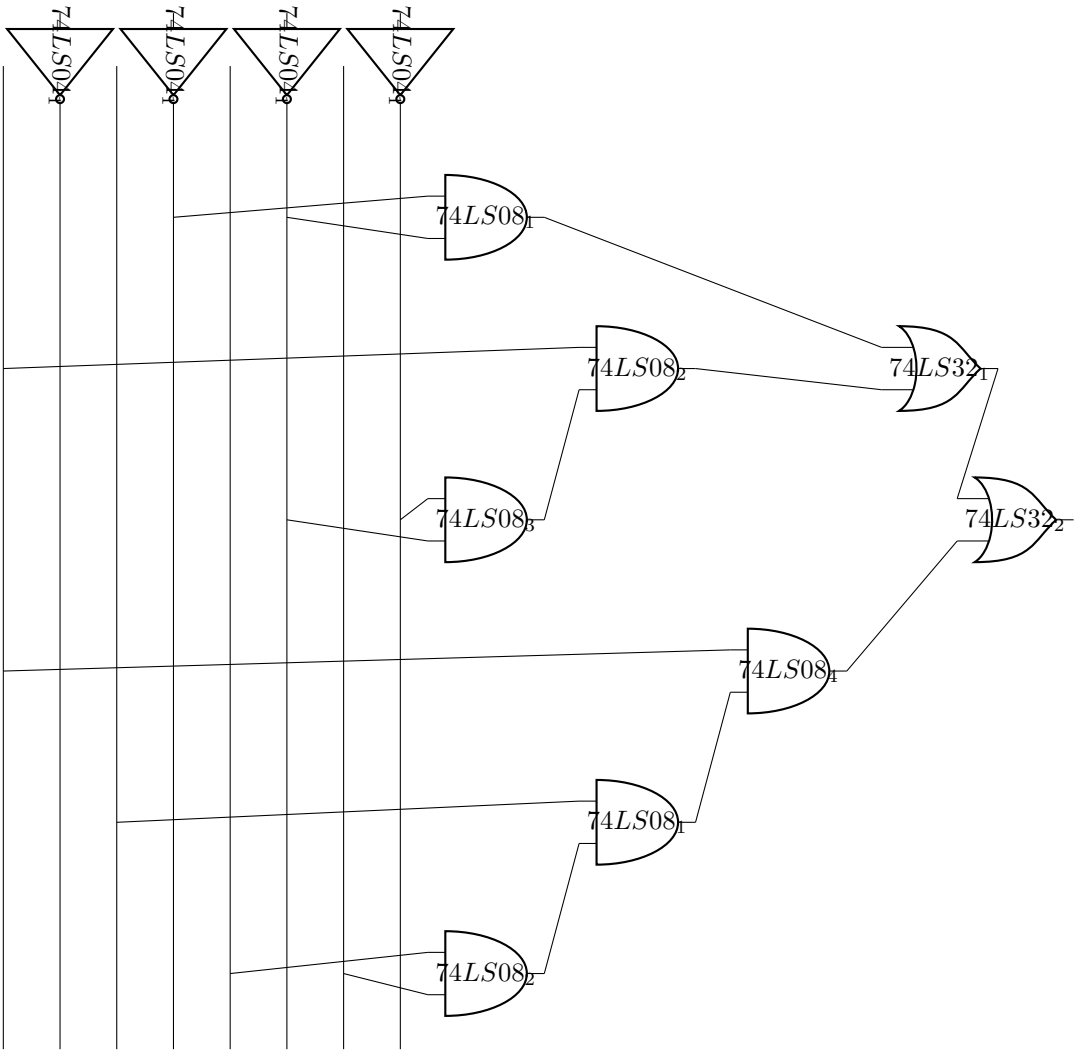
```
GAL16V8

MODULE GAL16V810272026DNF

CLK X3 X2 X1 X0 NC NC NC NC GND
/OE Y NC NC NC   NC NC NC NC VCC

Y = /x2 * /x1 +
  x3 * /x1 * /x0 +
  x3 * x2 * x1 * x0;

DESCRIPTION: GAL16V8 Code wie immer.
```




```
GAL16V8

MODULE GAL16V810272026DNF

CLK X3 X2 X1 X0 NC NC NC NC GND
/OE Y NC NC NC   NC NC NC NC VCC

Y = /x2 * /x1 +
  x3 * /x1 * /x0 +
  x3 * x2 * x1 * x0;

DESCRIPTION: GAL16V8 Code wie immer.
```

7 nächstes

typischerweise: assembler/nasm/i586

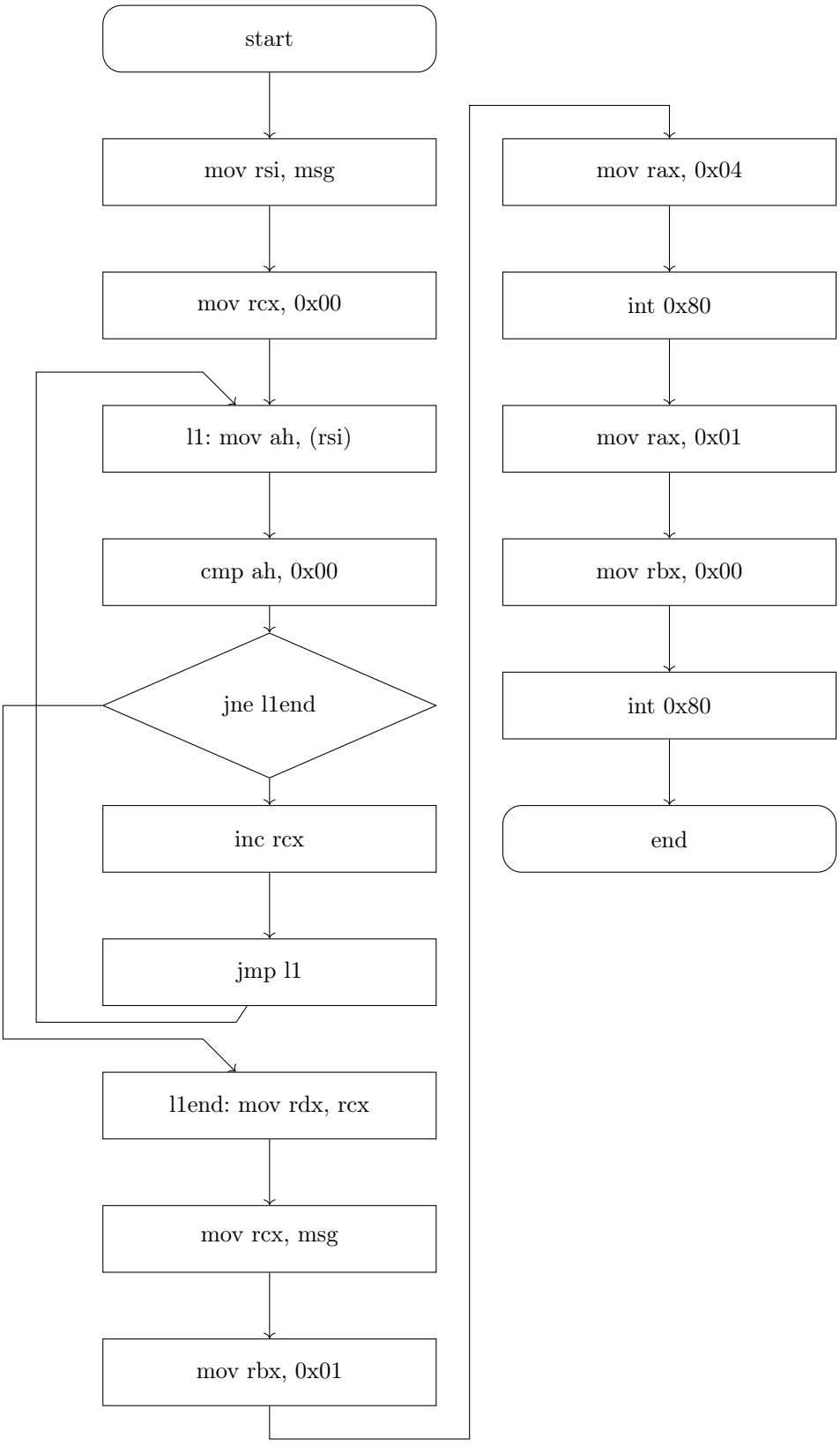
```
;; (c) david vajda
;; 10/27/2026
;; nasm/i586/msg output, write 0x04

global _start
section .data
 msg:  db "david@https://www.vajda-breadboard.de", 10, 13, 0
section. .text
_start:
 mov rsi, msg
 mov rcx, 0x00
l1: mov ah, [rsi]
 cmp ah, 0x00
 jne l1end
 inc rcx
 jmp l1
l1end:

 mov rdx, rcx
 mov rcx, msg
 mov rbx, 0x01
 mov rax, 0x04
 int 0x80

 mov rax, 0x01
 mov rbx, 0x00
 int 0x80
```

jetzt dazu, ein PAP:


start
mov rsi, msg
mov rcx, 0x00
l1: mov ah, (rsi)
cmp ah, 0x00
jne l1end
inc rcx
jmp l1
l1end: mov rdx, rcx
mov rcx, msg
mov rbx, 0x01
mov rax, 0x04
int 0x80
mov rax, 0x01
mov rbx, 0x00
int 0x80
end

8 im nächsten moment lagrange - interpolation

wie immer:

$$X = (1, 2, 3)$$

$$l_i(x) = \sum_{\substack{j=0 \\ j \neq i}}^{3} \frac{x - x_j}{x_i - x_j} \tag{240}$$

$$l_0(x) = \frac{x-2}{1-2} \cdot \frac{x-3}{1-3} \tag{241}$$

$$l_1(x) = \frac{x-1}{2-1} \cdot \frac{x-3}{2-3} \tag{242}$$

$$l_2(x) = \frac{x-1}{3-1} \cdot \frac{x-2}{3-2} \tag{243}$$

$$\tag{244}$$

um es etwas ansehnlicher zu gestalten, ist:

```
(x-2)*(x-3)/(1*(-2)) = (x^2 -3x -2x +6)/(-2) = -(x^2 -5x +6) = (-x^2 +5x -6)/2
```

[36]Raucherpause! - Kaffeepause!
falsch: Überlegungspause + Kaffee!
folgendes: lagrange jetzt nicht fertig! ist ok - vorher was dazwischen schieben, heute nachmittag ab 15:30 Uhr MEZ - (CET) Tuebingen, Germany auswendig lernen

(soll heissen: Stechuhr: Kaffeepause ist zwar! musste aber auf Idee kommen, durch Idee beflügelt - jetzt Kaffee erlauben)! Ernährung + antwort oekonomisch nicht unbedingt 100 pro sinnvoll - jetzt schon! ok!

← vorsicht! irrenfuehrende annähme! nicht 100pro notwendig! heisst! nicht Öko formel (keine ahnung - wie geht, keine ahnung wie heisst: Finanzwirtschaft: professor dr. Rainer bauLe beim scheitern – nicht bei Banane - was hier sinnvoll jetzt nicht klar, an der stelle Stechuhr sinnvoll! sicher! blos ist das keine Finanzwirtschaft - . . . ok - weiter

1. Zufallszahlen! das heisst, uebliche reihe machen, fuer zufallszahlen!
2. dann chi im Quadrat test - das mache ich auch
3. chi im Quadrat test, teste ich dann aber an (realen?? „nicht pseudo zufallszahlen“ob hier auch

go ahead! borscht Sprache abgekupfert - oder real: Fliegerei! jetzt!

ok: go ahead! go on!

now! coffee!

ok: back - go on!

in footnotemark? yes!

footnotemark! Footnotetext - auch in footnotemark!
[37]break cooffee ready! ok back! garten nachher reinigen von ein wenig 5 flocken - Haferflocken in Tasse in garten! heute Abend! jetzt weiter!

alternative! statt footnotemark! seiten (Randnotierung) in dem falle - keine ahnung name - ... steht in buch - keine Erklärung - und: cmd

```
\footnotmark
```

einfach ersetzbar

```
\footnotetext{...}
```

ok: das hier naechste footnotemark! ist wirklich eine! go ahead!
anmerkungen: [38]so, nachher ordentliche Fassung: zwei matrizen. jetzt erst rechnen und dsw. matrizenaddition von A und B. das hier auch ordentliche Fassung! warum addition!?! weil es grund gibt! eine multiplikation und eine addition! die schritte sind nachweislich. dann N und NP Laufzeit, wir zaehlen einfach die schritte mit und muessen untersuchen N und NP Vollstaendig!
ok, jetzt weiter... die zufallszahlen gehen so:

9 10/27/2026 methode der linearen kongruenz

1. methode der linearen kongruenz
2. methode der additiven kongruenz

$$a_{i-1} = (a_i \cdot b + 1) mod m \tag{245}$$
$$\tag{246}$$

heisst, programmiertechnisch

```
a[i-1] = (a[i] * b + 1) % m;
```

ok, erst die regeln für m und b

1. ok, erst die regeln für m und b
2. der kopf der tabelle und selber machen!

- m ist 2er oder 10er potenz, und am besten computerwort
- b
  - b ist eine stelle weniger als m
  - b endet auf $x21$
  - x ist gerade zahl
  - x hat kein besonderes muster, drinne

m nehmen wir

```
0b1000 0000 0000 0000
also

2^(15)

8192 * 4 =
sollte man mal auswendig lernen.
32768 entspricht 32k

weiter:

b ist:
2786
2821
daraus wird:
3821

ok: probieren
```

tabelle:

- a_i
- a_{i-1}
- b
- m
- vielleicht $a_{i-1} \cdot b$ und vielleicht $\cdots + 1 \ldots ???$

ok

hier sieht man a_i muesste eigentlich ans ende!

```
\begin{tabular}{|c|c|c|c|c|}
\hline
$a_i$ & $a_{i-1}$ & $b$ & $m$\\


\hline
\end{tabular}
```

[39]frage vorweg? muss m eine primzahl sein? weiss ich nicht! - frage vorweg ohne footnotemark! frage vorweg? muss m eine primzahl sein? weiss ich nicht! - frage

a_{i-1}	b	m	$(a_{i-1} = (a_i \cdot b + 1) \mod m$	a_i
aha ok what is seed??				
1023	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	9492
				10989
				2324

vorweg ohne footnotemark!

ergibt:

```
davidvajda@localhost ~ % echo $(((1023*3821+1)%32768))
9492
davidvajda@localhost ~ %
```

ok, machen gleich ganzes bash programm:

```
davidvajda@localhost ~ % echo $(((1023*3821+1)%32768))
9492
davidvajda@localhost ~ % echo $(((1023*9492+1)%32768))
10989
davidvajda@localhost ~ % echo $(((1023*10989+1)%32768))
2324
davidvajda@localhost ~ %
```

ok hier war klassischer fehler! fehler werden folgen! [40]deswegen uebung: fehler muessen dokumentiert werden, und beschrieben werden! also hier fehler!

danach eine idee - diese zufallszahlen sind sehr gross, im zweiten schritt $mod 16$.

```
#!/bin/bash
# (c) david vajda - no copyright
# belongs 2 all people on earth!
# ohne weitere diskussionen!
# 10/27/2026
# zufallzahlen - lineare kongruenz

i=0
while [ $i -lt 12 ]
do
        a=$(((1023*$a+1)%32768))
        i=$(($i+1));
        echo -n "$a, "
done
```

ok kommt raus

```
1023, 30722, 4095, 27650, 7167, 24578, 10239, 21506, 13311, 18434, 16383, 15362,
```

bei folgendem code

```
#!/bin/bash
# (c) david vajda - no copyright
# belongs 2 all people on earth!
```

```
# ohne weitere diskussionen!
# 10/27/2026
# zufallzahlen - lineare kongruenz

i=0
while [ $i -lt 12 ]
do
        a=$(((1023*$a+1)%32768))
        i=$((($i+1));
        echo -n "$a, "
done
```

ok, nun mit *modulo*16, stop vorher tabelle! ist mathematische funktion! methode besser und abbildung ! auch nicht, folge, aber am besten methode! funktion reelle nach reelle normalerweise aber abbildung. funktion ist abbildung. ok weiter!

a_{i-1}	b	m	$(a_{i-1} = (a_i \cdot b + 1) \mod m$	a_i

aha ok what is seed??

1023	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	9492 10989 2324

[41]deswegen uebung: fehler muessen dokumentiert werden, und beschrieben werden! also hier fehler!

```
#!/bin/bash
# (c) david vajda - no copyright
# belongs 2 all people on earth!
# ohne weitere diskussionen!
# 10/27/2026
# zufallzahlen - lineare kongruenz

i=0
a=1023
while [ $i -lt 12 ]
do
        a=$(((3821*$a+1)%32768))
        i=$((($i+1));
        echo -n "$a, "
done
```

und besser angaben ueber variablennanmen! so:

```
#!/bin/bash
# (c) david vajda - no copyright
# belongs 2 all people on earth!
# ohne weitere diskussionen!
# 10/27/2026
# zufallzahlen - lineare kongruenz

i=0
a=1023
m=32768
```

```
b=3821
while [ $i -lt 12 ]
do
        a=$(($a*$b+1)%32768))
        i=$(($i+1));
        echo -n "$a, "
done
```

das ist jetzt das ergebnis der methode (bzw. folge) vorsicht reihe und folge

```
9492, 27525, 20514, 2939, 23264, 24929, 29902, 26295, 6508, 28925, 28730, 4531,
```

$RA = (9492, 27525, 20514, 2939, 23264, 24929, 29902, 26295, 6508, 28925, 28730, 4531)$

1. Eine Folge, ist eine Abbildung, was einer Funktion gleicht, allerdings hat eine Folge im gegensatz zu einer Funktionen keinen Reellen Definitionsbereich!

2. Eine Folge kann als Elemente (a_i) so bezeichnet man eine folge im Allgemeinen (a_n) und nicht nicht $f(x)$ sein Element aus einen oder mehreren vorgänger Elementen berechnen

3. eine Reihe wiederum ist die Summe $\sum$ bzw. $\sum_{i=0}^{n}$ von Elementen einer Folge. Sie gleicht einer for schleife

   ```
   for (i = 0;  i <= n;  i++)
    ....
   ```

   bzw. wir koennen das genauer dafinieren! $\sum_{i=0}^{6}$ ist wahrscheinlich die einfachere weise, das zu schreiben:

   ```
   for (i = 0;  i <= 6;  i++)
    ....
   ```

   wenn wir davon ausgehen die zu addierenden elemente sind x_i, für $i = 1 \ldots 6$, $\sum_{i=0}^{6} x_i$

   ```
   S;
   for (i = 0;  i <= 6;  i++)
    S = S + x [i];
   ```

   bei einer folge, wiederum gilt, wo x besser a heisst: $\sum_{i=0}^{6}(a_i)$

   ```
   S;
   for (i = 0;  i <= 6;  i++)
    S = S + x [i];
   ```

und (a_i) kann zum beispiel $i + 4$ sein. oder $a_i = a_{i-1} + 3$. dabei muesen wir zunächst die elmente der Folge ausrechnen, sie als liste notieren und dann als summe zusammenrechnen

so jetzt weiter. so jetzt haben sie vielleicht für menschen, denen das schwer fällt ein paar infos, ich muss leider trotzdem weiter machen!

a_{i-1}	b	m	$(a_{i-1} = (a_i \cdot b + 1) \mod m$	a_i
1023	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	9492
9492	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	27525
27525	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	20514
20514	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	2939
2939	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	23264
23264	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	24929
24929	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	29902
29902	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	26295
26295	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	6508
6508	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	28925
28925	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	28730
28730	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	4531
4531	32768	3821	$(a_{i-1} = (1023 \cdot 3821 + 1) \mod 32768$	...

ok.

tabelle tut: jetzt mit wuerfel und chi quadrat test. ich probiere mal noch mal formel hinzu schreiben

$$\sum_{i=0}^{N} \frac{f_i - \frac{N}{r}}{\frac{N}{r}}$$

blos, was ist r ich weiss die formel noch auswendig, blos was ist r. ok pause, fuer heute, kaffee! uebersetzen, zum facebook, zur homepage, kaffee auswendig lernen! oder doch noch wuerfeln, gucken was r aber vorher kaffee! und nicht nur! sondern: auf den computer fuer ...!

[42]mit taschenrechner[43]dabei kommt eine gleitkommazahl vermutlich ieee-754 ähnliches raus, den ganzzahl wert vor dem komma - differenz - als subtrahend machen - abziehen und den nach komma wert mit dem divisor teilen heisst rest [44]gehört eigentlich die anmekrung in die tabelle wo umgerechnet wird, ist mir aber aufgefallen - das bleibt dann in der tabelle footnotemark! erstens: zweitens es geht besser als mit taschenrechner und nachkomma - mit bash - diese kann ganzzahl! und modulo! schritt fuer schritt! wie wir! koennen aber bash skript schreiben! machen auch! gute uebung! jetzt!

so, nächster teil: jetzt ist r die obere grenze/das maximum/die obere schranke und die entspricht unserem m. und das N entspricht der anzahl der Zufallszahlen nämlich m.

so f_i sind die jeweiligen, zufallszahlen selber. müssen wir diese reihe fertig machen?! jetzt?! nein, auswendig lernen!

10 nächster tag

also, fuer die summe, nämlich - das ist auch eine - den chi quadrat test. die frage: was ergibt sich aus?

$$\sum_{i=0}^{N} \frac{f_i - \frac{N}{r}}{\frac{N}{r}}$$

- Ich vermute: $N = 12$, ist die anzahl der Zufallszahlen, die wir hier haben.
- r ist die obere Grenze das Maximum, von Zufallszahlen.
- f_i sind die einzelnen zufallszahlen

$$\frac{N}{r} = \frac{12}{32768}$$

ok, so rechne ich

$$\frac{N}{r}$$

aus, und das kann ich mit dem computer machen. also lautet die summe.

$$\begin{aligned}
\chi^2 &= \frac{9492 - \frac{12}{32768}}{\frac{12}{32768}} && (247)\\
&+ \frac{27525 - \frac{12}{32768}}{\frac{12}{32768}} && (248)\\
&+ \frac{20514 - \frac{12}{32768}}{\frac{12}{32768}} && (249)\\
&+ \frac{2939 - \frac{12}{32768}}{\frac{12}{32768}} && (250)\\
&+ \frac{23264 - \frac{12}{32768}}{\frac{12}{32768}} && (251)\\
&+ \frac{24929 - \frac{12}{32768}}{\frac{12}{32768}} && (252)\\
&+ \frac{29902 - \frac{12}{32768}}{\frac{12}{32768}} && (253)\\
&+ \frac{26295 - \frac{12}{32768}}{\frac{12}{32768}} && (254)\\
&+ \frac{6508 - \frac{12}{32768}}{\frac{12}{32768}} && (255)\\
&+ \frac{28925 - \frac{12}{32768}}{\frac{12}{32768}} && (256)\\
&+ \frac{28730 - \frac{12}{32768}}{\frac{12}{32768}} && (257)\\
&+ \frac{4531 - \frac{12}{32768}}{\frac{12}{32768}} && (258)\\
& && (259)
\end{aligned}$$

vielleicht kann man das trotzdem mit einem python3 programm vereinfachen.

```
## (c) david vajda
## 10/28/2026
## py 3 - chi quadrat test

RA = (9492, 27525, 20514, 2939, 23264, 24929, 29902, 26295, 6508, 28925, 28730, 4531)
fi = RA
N = 12
r = 32768
Nperr = 12/r
ChiSqr

i = 0
ChiSqr = 0
while i < N:
 ChiSqr = ChiSqr + (RA - Nperr)/Nperr
 i = i + 1

print (ChiSqr)
```

11 chi quadrat zahlen getestet

12 10/29/2026

netzwerktechnische uebungen.

12.1 minix

```
# david vajda
# 10/29/2026
# apache2 config-file

<VirtualHost *.*>
    ServerName      defaultserver.mil.nongov
    ServerAdmin     milchef001@defaultserver.mil.nongov
    DocumentRoot    /var/www/html/
</VirtualHost>
# david vajda
# 10/29/2026
# smtp server procotol

# telnet 127.0.0.1 25
# 25/21 ist smtp/ftp

HELO 127.0.0.1
MAIL FROM xyz@abc.xy
RCPT TO abz@xyz.ab
```

```
DATA

<CR><LF>
.
gpg -k
gpg -K

gpg --generate-key
gpg --generate-secret-key

# gpg
# so weiter weiss ich auch nicht
# david vajda
# 10/29/2026
# main.cf - postfix

smtpd_client_restrictions
smtpd_sender_restrictions
smtpd_recipient_restrictions
smtpd_relay_restrictions
smtpd_helo_restrictions
# da fehlen noch 3 oder so

# reject_unknown_hostname
# reject_unknown_client_hostname
# reject_unknown_hello_hostname
# reject_unknown_sender_domain
# reject_unknown_recipient_domain
# reject_non_fqdn_sender
# reject_non_fqdn_recipient
# reject_unknown_reverse_client_hostname


/* david vajda
 * minix.c
 * 10/29/2026
 *
 * vir phys len
 * /
 */

struct mem_map {
    vir_clicks mem_vir;
    phys_clicks mem_phys;
    vir_clicks mem_len;
};

struct proc{
    struct mem_map p_map [NR_SEGS];
    int p_reg [NR_REGS]; //?
```

```
    pc // programm counter?
    *sp; // stack_pointer
    sp_limit //
    struct proc *next_proc;
} [NR_PROCS + NR_TASKS];

/* bei mm heisst das: mproc */

// bei den defines sind:

#define ES_REG
#define DS_REG
#define CS_REG

// pic 8259A

// mask

#define NERROR 34
#define OK 0
#define ERROR -1
#define ENOMEM
#define ENOENT

// usw

#define SIG_HUP
// wie heissen die definitions konkret?
auto eth0
iface eth0 inet dhcp

auto wlp2s0
iface wlp2s0 inet static
    address 192.168.0.28/24
    netmask 255.255.255.0
    gateway 192.168.0.1
    dns-server

auto lo0
iface lo0 inet loopback
```

12.2 apache2/default.conf

12.3 m8 programms + PAP

12.3.1 PORTB/LED OUT

```
;; (c) david vajda
;; 10/29/2026
```

```
;; atmega8/m8/led out

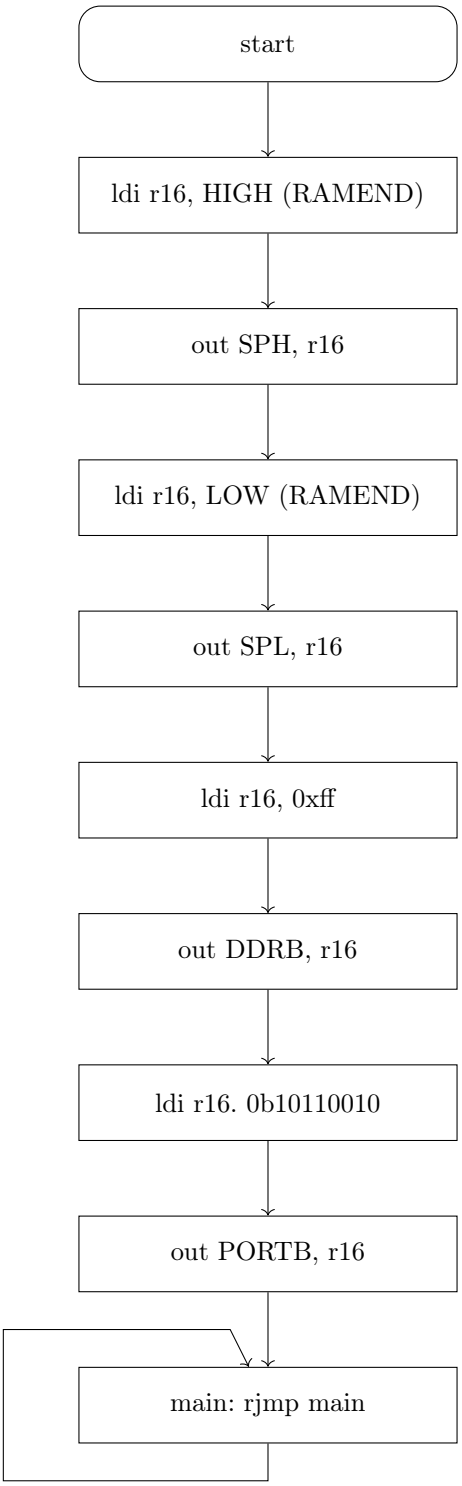
.include "m8def.inc"

ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16

ldi r16, 0xff
out DDRB, r16

ldi r16. 0b10110010
out PORTB, r16

main: rjmp main
```


start
ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16
ldi r16, 0xff
out DDRB, r16
ldi r16. 0b10110010
out PORTB, r16
main: rjmp main

12.3.2 LED Count up/waiting loop

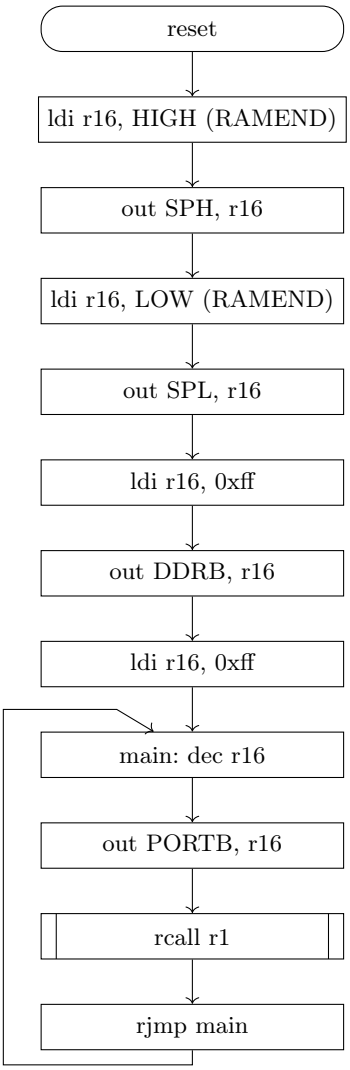
```
;; (c) david vajda
;; 10/29/2026
;; atmega8/m8/led out/up count
.include "m8def.inc"

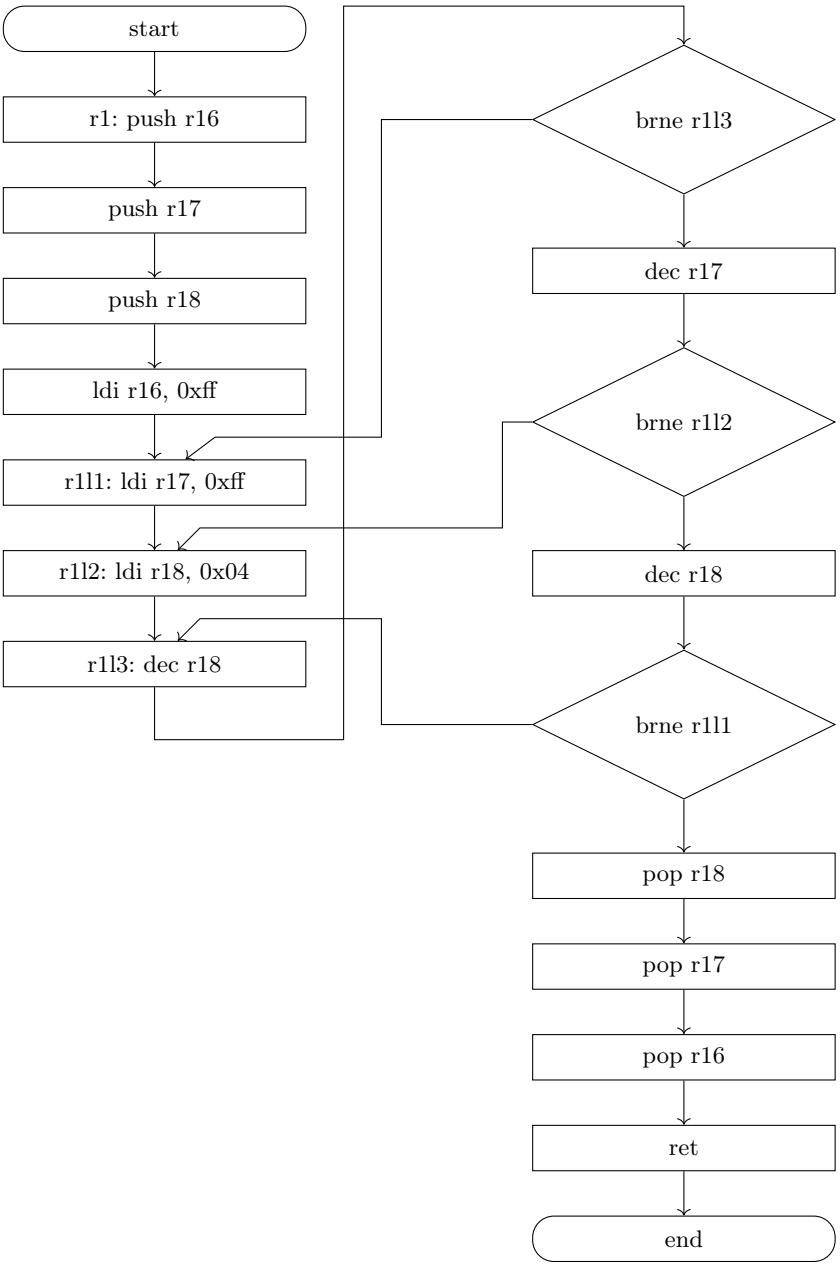
ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16

ldi r16, 0xff
out DDRB, r16

ldi r16, 0xff
main:
dec r16
out PORTB, r16
rcall r1
rjmp main

r1:
push r16
push r17
push r18
ldi r16, 0xff
r1l1:
ldi r17, 0xff
r1l2:
ldi r18, 0x04
r1l3:
dec r18
brne r1l3
dec r17
brne r1l2
dec r18
brne r1l1
pop r18
pop r17
pop r16
retheight=0.6cm
```


reset
ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16
ldi r16, 0xff
out DDRB, r16
ldi r16, 0xff
main: dec r16
out PORTB, r16
rcall r1
rjmp main



12.4 wir probieren auch noch ein struktogramm...

...und zwar zu folgendem: bash skripting. erst das bash skript

```
#!/bin/bash

# (c) david vajda
# 10/29/2026
# bash scripting 4 struktogramme
```

```
i=0
while [ $i -lt 10 ]
do
 j=0
 while [ $j -lt 10 ]
 do
  if [ $i -eq $j ]
  then

  elif [ $i -lt $j ]
  then

  elif [ $(($i%$j)) -eq $2$  ]
  then

  fi
  else

  j=$(($j+1))
 done
 i=$(($i+1))
done
```

struktex also für LaTeXhabe ich eine: version für struktogramme gefunden.

```
\documentclass{article}
\usepackage{struktex}

\begin{document}

\begin{struktogramm}(100,40)
  \assign{Anweisung 1}
  \ifthenelse{1}{1}{Bedingung}{ja}{nein}
    \assign{Anweisung 2}
  \change
    \assign{Anweisung 3}
  \ifend
\end{struktogramm}

\end{document}
```

verrät google ai. die alternative besteht darin: es ohne struktex klassischerweise mit tikzpicture zu machen

```
\documentclass{article}
\usepackage{tikz}
\usetikzlibrary{positioning, calc}

\begin{document}

\begin{figure}[htbp]
```

```
\centering
\begin{tikzpicture}[
    % Definition der Standard-Block-Styles
    block/.style={
        draw,
        rectangle,
        minimum height=0.8cm,
        minimum width=6cm,
        text width=5.6cm,
        align=left,
        inner sep=5pt,
        node distance=-0.5pt % Verhindert doppelte Linien beim Aneinanderreihen
    },
    ifblock/.style={
        draw,
        rectangle,
        minimum height=1.2cm,
        minimum width=6cm,
        node distance=-0.5pt
    }
]

    % 1. Sequenz / Anweisung
    \node[block] (step1) {1. Initialisiere Variable $x = 0$};

    % 2. Schleifen-Kopf (z.B. Solange x < 10)
    \node[block, below=of step1, minimum width=6cm, text width=5.6cm, fill=gray!10] (while

    % 3. Anweisung innerhalb der Schleife (eingerückt angedeutet durch Position)
    \node[block, below=of while, minimum width=5.5cm, text width=5.1cm, shift={(0.25cm,0)}]

    % 4. IF-Bedingung (Verzweigung)
    % Wir zeichnen den Kasten und teilen ihn mit Linien
    \node[ifblock, below=of step2, minimum width=5.5cm, shift={(-0.25cm,0)}] (ifbox) {};

    % Diagonale Linien für das "Treppchen" der Bedingung
    \draw (ifbox.north west) -- ($(ifbox.north)!0.5!(ifbox.south)$) -- (ifbox.north east);

    % Beschriftung der Bedingung
    \node[below=2pt of ifbox.north, font=\small] {Ist $x$ gerade?};
    \node[above left=2pt and 5pt of ifbox.south, font=\footnotesize] {Ja};
    \node[above right=2pt and 5pt of ifbox.south, font=\footnotesize] {Nein};

    % Die zwei Spalten unter der Bedingung
    \node[block, below=of ifbox, minimum width=2.75cm, text width=2.45cm, anchor=north wes
    \node[block, below=of ifbox, minimum width=2.75cm, text width=2.45cm, anchor=north eas

    % 5. Letzter Block (wieder Gesamtbreite)
    % Wir referenzieren den tiefsten Punkt der Verzweigung
    \path (truebranch.south) -- (falsebranch.south) node[scalable, draw, rectangle, minimu
```

```
\end{tikzpicture}
\caption{Nassi-Shneiderman-Struktogramm in TikZ}
\label{fig:struktogramm}
\end{figure}

\end{document}
```

so: jetzt ist folgendes: also: Nassi-Shneiderman-Struktogramm in TikZ. jetzt habe ich das. ich probiere es mit struktex. es hat google ai gesagt. ich probiere es aus. leider hat struktex seinen dienst versagt.

also, folgendes: beide Nassi-Shneiderman-Struktogramm (beide nassi schnudo doktogramme) - ist mir gerade raus gerutscht. damit das nicht passiert...

[45] nassi schnudo doktogramme

12.4.1 externes Interrupt

```
;; (c) david vajda
;; 10/29/2026
;; atmega8/m8/led out/up count/interrupt/extern
.include "m8def.inc"

.org 0x000
.org INT0ADDR:
rjmp INT0HNDLR
.org INT1ADDR:
rjmp INT1HNDLR

ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16

ldi r16, 0xff
out DDRB, r16

ldi r16, 0xff
out PORTB, r16

;; jetzt haben wir externe interrupts
;; zwei register (ports) - wie auch immer
;; bei Timer/Counter
;; TIMSK und TCCRA und TCCRB2
;; TCCRB1
;; Timer/Interrupt Mask Register
;; Timer/Counter Control Register
;; GICR - Global Interrupt Controll Register
;; MCUCR - MCU Controll Register
```

```
ldi r16, (1 << ICS10) | (1 << ICS00)
out GICR, r16
ldi r16, (1 << INT1) | (1 << INT0)
out MCUCR, r16

sei

main: rjmp main


INT1HNDLR:
dec r16
out PORTB, r16
reti

INT0HNDLR:
inc r16
out PORTB, r16
reti
```

12.4.2 v.24 transmit

```
;; (c) david vajda
;; 10/29/2026
;; atmega8/v.24/transmit
.include "m8def.inc"

ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16

ldi r16, HIGH (25)
out UBBRH, r16
ldi r16, LOW (25)
out UBRRL, r16

ldi r16, (1 << UCSZ1) | (1 << UCSZ0) | (1 << USBS) | (1 << UPM1)
out UCSRC, r16
ldi r16, (1 << TXEN)
out UCRSB, r16

;; es gibt 1/2 stopbit, 1 startbit dafuer kein bit
;; es gibt usart character size: 5, 6, 7, 8, 9 datenbit
;; 2400/4800/9600 baudrate: character/second
;; odd/even/no parity - odd und even, ich glaube even ist gerade
;; UCSRB - ist aktivierung, UCSRC, sind viele flags
;; UCSRA ist am Ende, wegen Transmit/Recieve

main2:
```

```
ldi YH, HIGH (2*msg)
ldi YL, LOW (2*msg)
main1:
lpm r16, Y+
rcall v.24transmit
cpi r16, 0x00
brne main1
rjmp main2

msg: db "david@https://www.vajda-breadboard.de/", 10, 13, 0

v.24transmit:
push r16
sbis UCSRA, UDRE
rjmp v.24transmit
out UDR, r16
pop r16
ret
```

13 Ergebnis: was muss gelernt werden?

- stellen in Timer/Counter Interrupt
- gewisse stellen in LCD HD44780
- ganz gewisse Stellen rs232
- eher: ein wenig Externes Interrupt
- Minix, habe ich nicht vollkommen geschrieben, was zu schreiben ist.
- aber: postfix
- noch fehlend ist: kernel log messages im protokoll: syslog
- vergessen zu tun: syslog - aber keine ahnung
- minix, letzten endes, nicht weiter gemacht
- lilo und grub

14 PAP zu den Assembler programmen

15 UML

vorüberlegungen?

- Interaktionsdiagramm muss rein
- Handel - Kaufvertrag kommt zu stande
- Klasse Mathematik -

- die methoden - oder computersystem - fuer algos - oder beide bilden assoziation.

so, was wir in UML jetzt brauchen, sind folgendes bei Diagrammen

1. ein Klassendiagramm
2. ein Interaktionsdiagramm

wenn wir ein klassendiagramm nehmen, nehmen wir folgendes:
sie nehmen methoden (funktionsnamen) und: sie nehmen attribute, variablen. diese ganzen mengen bekommen einen namen, den namen der klasse. vererbung heisst:

1. eine vererbung ist keine untermenge, sondern wenn noch zu der klasse davor, neues dazu kommt
2. assoziationen zwischen klassen, sind paare.

1. Namen der Klasse C
2. Methoden der Klasse $f(x), g(x), h(x)$
3. Attribute der Klasse x, y, z

also:

$$\begin{aligned} C &= \{ && (260)\\ &\quad \{f(x), g(x), h(x)\} && (261)\\ &\quad \{x, y, z\} && (262)\\ &\quad \} && (263)\\ & && (264) \end{aligned}$$

$$C = \{\{f(x), g(x), h(x)\}, \{x, y, z\}\} \quad (265)$$
$$(266)$$

$$C = \left\{ \begin{array}{c} \{f(x), g(x), h(x)\} \\ \{x, y, z\} \end{array} \right\}$$

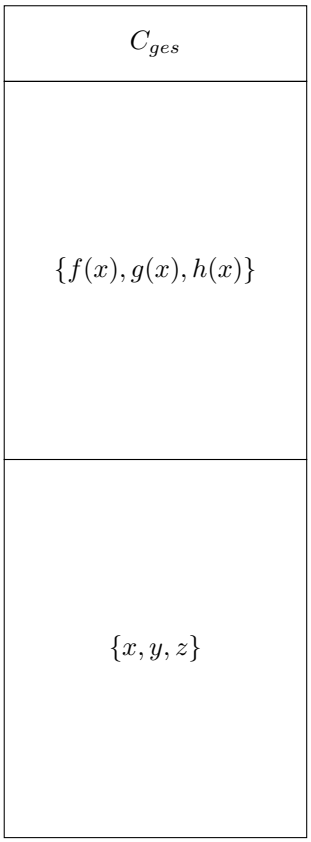
$$\begin{aligned} C_1 &= \{ && (267)\\ &\quad \{f(x), g(x), h(x)\} && (268)\\ &\quad \{x, y, z\} && (269)\\ &\quad \} && (270)\\ & && (271) \end{aligned}$$

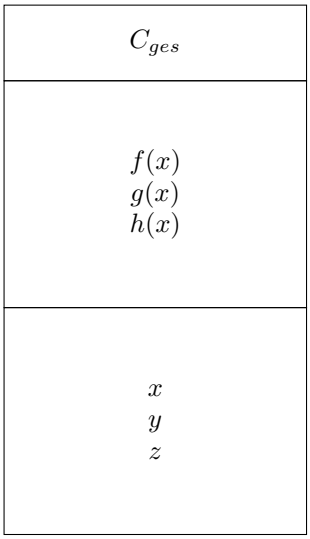
$$\begin{aligned} C_2 &= \{ && (272)\\ &\quad \{f_1(x), g_1(x_1, x_2), h_1(x)\} && (273)\\ &\quad \{x_1, x_2, x_3\} && (274)\\ &\quad \} && (275)\\ & && (276) \end{aligned}$$

nun eine assoziation.

$$
\begin{aligned}
C_1 &= \{ && (277)\\
&\quad \{f(x), g(x), h(x)\} && (278)\\
&\quad \{x, y, z\} && (279)\\
&\quad \} && (280)\\
C_2 &= \{ && (281)\\
&\quad \{f_1(x), f_1(x_1, x_2), f_1(x)\} && (282)\\
&\quad \{x_1, x_2, x_3\} && (283)\\
&\quad \} && (284)\\
G_g &= \{C_1, C_2\} && (285)\\
& && (286)
\end{aligned}
$$

jetzt ein UML-Diagramm



jetzt kommt hier: 

vererbung nicht gleich subklasse. sondern noch was dazu, die toplevelclass ist hauptbestandteil der naechsten.

$$C = \left\{ \begin{array}{c} \{f(x), g(x), h(x)\} \\ \{x, y, z\} \end{array} \right\}$$

$$\begin{aligned} C_1 &= \{ && (287)\\ &\quad \{f(x), g(x), h(x)\} && (288)\\ &\quad \{x, y, z\} && (289)\\ &\quad \} && (290)\\ & && (291) \end{aligned}$$

$$\begin{aligned} C_2 &= \{ && (292)\\ &\quad \{f_1(x), g_1(x_1, x_2), h_1(x)\} && (293)\\ &\quad \{x_1, x_2, x_3\} && (294)\\ &\quad \} && (295)\\ & && (296) \end{aligned}$$

$$\begin{aligned} D &= \{ && (297)\\ &\quad C_1, && (298)\\ &\quad \{k_1(x), k_2(x_1, x_2), k_3(x)\}, && (299)\\ &\quad \{a, b, c\}, && (300)\\ &\quad \} && (301)\\ & && (302) \end{aligned}$$

$$\begin{aligned} A &= \{C_1, C_2\} && (303)\\ & && (304) \end{aligned}$$

stattdessen mit listen, als mengen, geordnet und ungeordnet.

$$C = \left(\begin{array}{c} (f(x), g(x), h(x)) \\ (x, y, z) \end{array} \right)$$

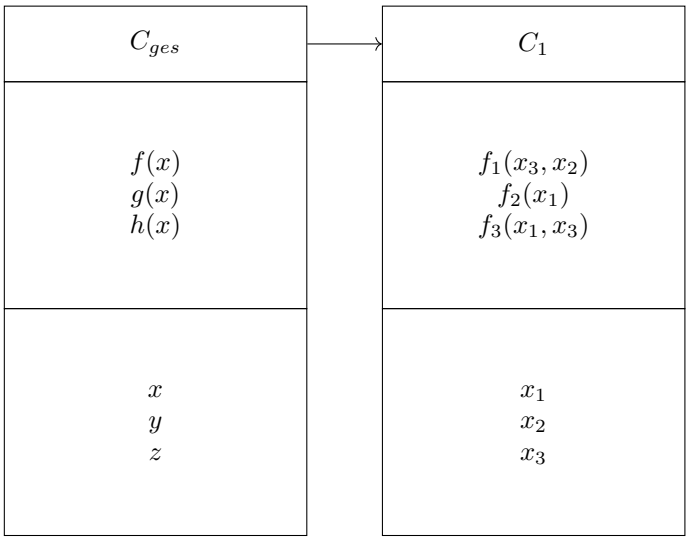
$$\begin{aligned} C_1 &= (&& (305)\\ &\quad (f(x), g(x), h(x)) && (306)\\ &\quad (x, y, z) && (307)\\ &\quad) && (308)\\ & && (309) \end{aligned}$$

$$\begin{aligned} C_2 &= (&& (310)\\ &\quad (f_1(x), g_1(x_1, x_2), h_1(x)) && (311)\\ &\quad (x_1, x_2, x_3) && (312)\\ &\quad) && (313)\\ & && (314) \end{aligned}$$

$$\begin{aligned} D &= (&& (315)\\ &\quad C_1, && (316)\\ &\quad (k_1(x), k_2(x_1, x_2), k_3(x)), && (317)\\ &\quad (a, b, c), && (318)\\ &\quad) && (319)\\ & && (320) \end{aligned}$$

$$\begin{aligned} A &= (C_1, C_2) && (321)\\ & && (322) \end{aligned}$$

so, jetzt haben das, jetzt kommt es ins klassendiagramm. im zweiten schritt - überlegen die lineare kongruenz und additive. als methoden annehmen. dann kommt der mathematiker dazu. ins interaktionsdiagramm. dann kommen noch andere methode - nicht die beweismethode, vielleicht für den mathematiker. Die NP Laufzeit, FFT und Fourier - splines + lagrange - als methoden + interpolation. ins klassendiagramm.
tiefensuche und breitensuche.



Das ist UML, weil die vererbte nicht unter der Superklasse oder Topklasse steht, sondern daneben eher assoziation, von UML. was in mathematik paare sind.

16 quine mc cluskey und DNF und ieee-1984/91 - tikz - jeden tag.

```
(C) David Vajda
Fri Oct 30 13:07:13 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    0
 1 0 0 0 1    0
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    0
 5 0 1 0 1    1
 6 0 1 1 0    0
 7 0 1 1 1    0
 8 1 0 0 0    0
 9 1 0 0 1    1
10 1 0 1 0    1
11 1 0 1 1    1
12 1 1 0 0    0
13 1 1 0 1    0
14 1 1 1 0    1
15 1 1 1 1    0


 2 0 0 1 0    1
 3 0 0 1 1    1
 5 0 1 0 1    1
 9 1 0 0 1    1
```

```
10 1 0 1 0    1
11 1 0 1 1    1
14 1 1 1 0    1

jetzt zu gruppen zusammenfassen.

Gruppe 1:
 2 0 0 1 0    1
Gruppe 2:
 3 0 0 1 1    1
 5 0 1 0 1    1
 9 1 0 0 1    1
10 1 0 1 0    1
Gruppe 3:
11 1 0 1 1    1
14 1 1 1 0    1

n"achster algo schritt, koennen wir PAP draus machen, die dinger zusammenfassen, dann noch


Gruppe 1:
 2 0 0 1 0    1
Gruppe 2:
 3 0 0 1 1    1
 5 0 1 0 1    1
 9 1 0 0 1    1
10 1 0 1 0    1
Gruppe 3:
11 1 0 1 1    1
14 1 1 1 0    1

(2,3)  0 0 1 x
(2,10) x 0 1 0
5  0 1 0 1
(3,11) x 0 1 1
(9,11) 1 0 x 1
(10,11) 1 0 1 x
(10,14) 1 x 1 0

algo schritt, sortieren, pap machen? nicht pap - asm hardware algorithmus oder beides,
gedachter hardware algo?


5  0 1 0 1
(2,10) x 0 1 0
(3,11) x 0 1 1
(10,14) 1 x 1 0
(9,11) 1 0 x 1
(0,3)  0 0 1 x
(10,11) 1 0 1 x
```

```
jetzt naechster schritt, wieder zusammen fassen.

5  0 1 0 1
(2,10) x 0 1 0
(3,11) x 0 1 1

5   0 1 0 1
(2,10,3,11) x 0 1 x
(10,14) 1 x 1 0
(9,11) 1 0 x 1
(0,3)  0 0 1 x
(10,11) 1 0 1 x

(2,3,10,11) x 0 1 x

5   0 1 0 1
(2,10,3,11) x 0 1 x
(10,14) 1 x 1 0
(9,11) 1 0 x 1
(2,3,10,11) x 0 1 x

so, jetzt geht die sache weiter:


5   0 1 0 1
(2,10,3,11) x 0 1 x
(10,14)  1 x 1 0
(9,11)  1 0 x 1

jetzt muessten wir fertig sein.
   2 3 5 9 10 11 14
5     H
(2,10,3,11) H H   H H
(10,14)      H  H
(9,11)     H  H
```

	2	3	5	9	10	11	14
5			H				
$(2,10,3,11)$	H	H			H	H	
$(10,14)$					H		H
$(9,11)$				H		H	

```
5   0 1 0 1
(2,10,3,11) x 0 1 x
(10,14)  1 x 1 0
(9,11)  1 0 x 1

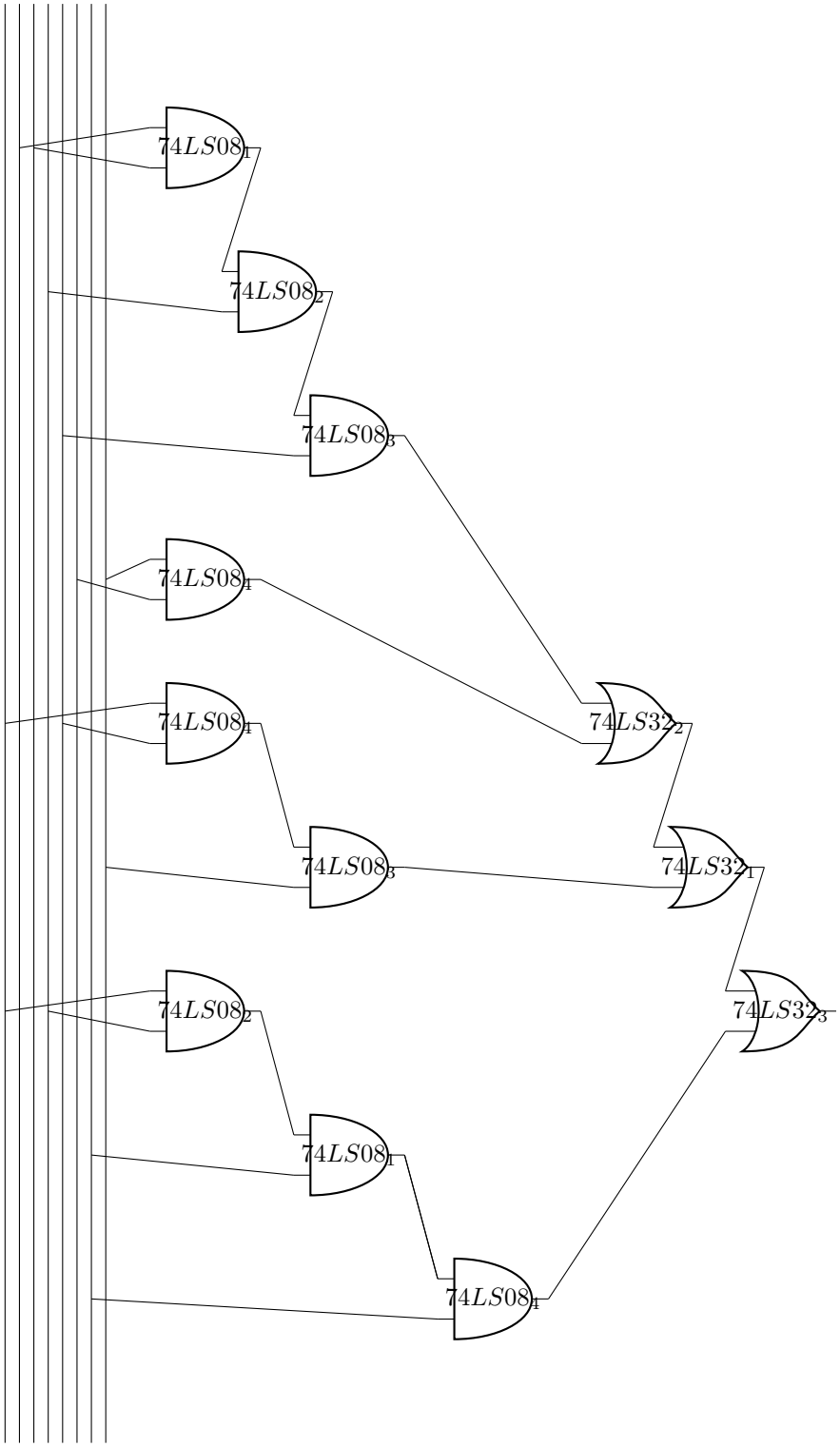
gal16v8
module gal16v8quinednf
```

```
clk x3 x2 x1 x0 nc nc nc nc vcc
/oe y nc nc nc   nc nc nc nc gnd

 y = /x3 * x2 * /x1 * x0 +
   /x2 * x1 +
   x3 * x1 * /x0 +
   x3 * /x2 * x1

description
quine dnk like always - but
very ultra super extra

;; always very
;; always ultra
;; always super
;; always extra
```




```
\begin{tikzpicture}
```

```
\node [and port] (and1) at (0,16) {$74LS08_1$};
\node [and port] (and2) at (1,14) {$74LS08_2$};
\node [and port] (and3) at (2,12) {$74LS08_3$};

\node [and port] (and4) at (0,10) {$74LS08_4$};

\node [and port] (and5) at (0,8) {$74LS08_4$};
\node [and port] (and6) at (2,6) {$74LS08_3$};

\node [and port] (and7) at (0,4) {$74LS08_2$};
\node [and port] (and8) at (2,2) {$74LS08_1$};
\node [and port] (and9) at (4,0) {$74LS08_4$};


\node [or port] (or3) at (6,8) {$74LS32_2$};
\node [or port] (or2) at (7,6) {$74LS32_1$};
\node [or port] (or1) at (8,4) {$74LS32_3$};

\draw (and1.out) -- (and2.in 1);
\draw (and2.out) -- (and3.in 1);

\draw (and7.out) -- (and8.in 1);
\draw (and8.out) -- (and9.in 1);

\draw (and8.out) -- (and9.in 1);

\draw (and5.out) -- (and6.in 1);

\draw (or3.out) -- (or2.in 1);
\draw (or2.out) -- (or1.in 1);

\draw (and3.out) -- (or3.in 1);
\draw (and4.out) -- (or3.in 2);
\draw (and6.out) -- (or2.in 2);

\draw (and9.out) -- (or1.in 2);

%\draw (-2,5) -- (and7.in 1);
%\draw (-2,5) -- (and6.in 2);
%\draw (-3,5) -- (-2,5);

\draw (-2,-2) -- (-2,18);
\draw (-2.2,-2) -- (-2.2,18);
\draw (-2.4,-2) -- (-2.4,18);
\draw (-2.6,-2) -- (-2.6,18);
\draw (-2.8,-2) -- (-2.8,18);
\draw (-3.0,-2) -- (-3.0,18);
\draw (-3.2,-2) -- (-3.2,18);
\draw (-3.4,-2) -- (-3.4,18);
```

```
\draw (-3.2,16) -- (and1.in 1);
\draw (-3.0,16) -- (and1.in 2);
\draw (-2.8,14) -- (and2.in 2);
\draw (-2.6,12) -- (and3.in 2);

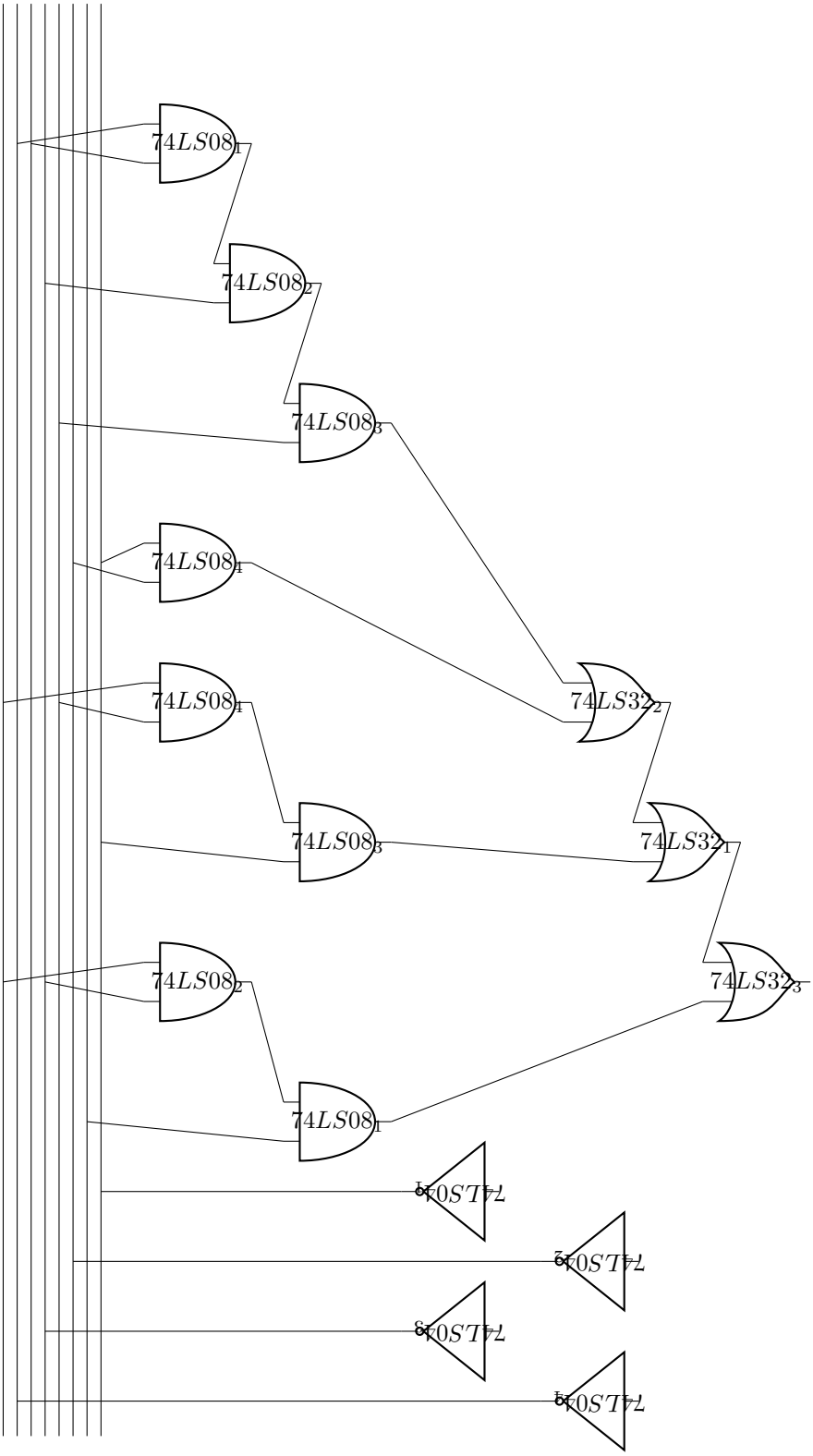
\draw (-2,10) -- (and4.in 1);
\draw (-2.4,10) -- (and4.in 2);

\draw (-3.4,8) -- (and5.in 1);
\draw (-2.6,8) -- (and5.in 2);
\draw (-2,6) -- (and6.in 2);

\draw (-3.4,4) -- (and7.in 1);
\draw (-2.8,4) -- (and7.in 2);
\draw (-2.2,2) -- (and8.in 2);
\draw (-2.2,0) -- (and9.in 2);

\end{tikzpicture}
```

hier ist allerdings im süden ein AND Gatter zu viel.


74LS08 1
74LS08 2
74LS08 3
74LS08 4
74LS08 4
74LS08 3
74LS08 2
74LS08 1
74LS32 2
74LS32 1
74LS32 3
74LS04
74LS04
74LS04
74LS04

17 automat

pattern matching...

also, ich probiere jetzt folgendes: ich mache jetzt einen automaten zum zeichen wiederfinden - pattern matching - reguläre ausdrücke - und der muss

```
(C) David Vajda
Fri Oct 30 13:07:13 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    0
 1 0 0 0 1    0
 2 0 0 1 0    1
 3 0 0 1 1    1
 4 0 1 0 0    0
 5 0 1 0 1    1
 6 0 1 1 0    0
 7 0 1 1 1    0
 8 1 0 0 0    0
 9 1 0 0 1    1
10 1 0 1 0    1
11 1 0 1 1    1
12 1 1 0 0    0
13 1 1 0 1    0
14 1 1 1 0    1
15 1 1 1 1    0

0b00.11.01.00.01.11.00.10

ADBABDAC

(A(B|D)*)C

# jetzt koennen wir probieren den grammatikalisch mit
# expr term fact ab zu leiten.

# wie sind allerdings die ausdruecke bei
# BNF - aber: Regul"are ausdr"ucke pattern matching

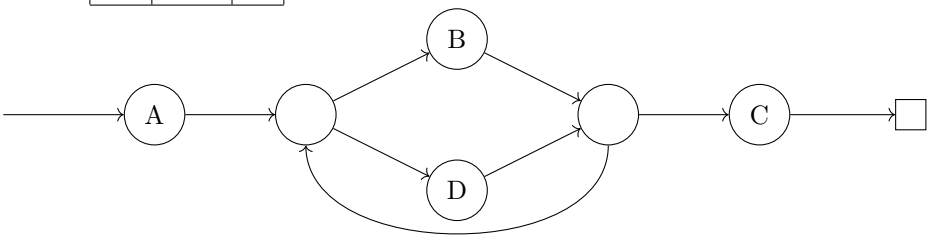
# nicht beseitigung der linksrekursion

<expr> ::= <term> | <term> + <expr>
<term> ::= <fact> | <fact> * <term>
<fact> ::= (<expr>) | v
```

jetzt schmieden des automaten: versuch mit tabelle, wie bei zustandstabelle üblich. oder: klassischer weise mit C. also Feldern. und der normale automat mit kringeln, heisst knoten - Vertices V oder Kanten Edges E.

Mealy:

Z_i	Z_i^{+1}	x
Z_1	Z_2	A




```
/*                  0, 1,  2,  3, 4,  5,  6, 7  */
int next [] = {' ', 'A', ' ',    'B','D',      ' ',     'C',' '};
char state1 [1, 2, 3,       5, 5,         6,      7, 7];
char state2 [1, 2, 4,       5, 5,         2,      7, 7];
```

state	0	1	2	3	4	5	6	7
int next () =		A		B	D		C	
char state1	1	2	3	5	5	6	7	7
char state2	1	2	4	5	5	2	7	7

so, jetzt haben wir den automat richtig.

was machen wir als nächstes für eine aufgabe, das ist die frage?

hier ein paar noten in LaTeXin der C-Dur Tonleiter leider funktioniert diese umgebung nicht:

```
\begin{lilypond}
  \relative c' {
    \key c \major
    c4 d e f g a b c c b a g f e d c
  }
\end{lilypond}
```

deswegen lassen wir es. machen wir die tabelle von oben weiter, zustandstabelle mit mealy.

18 assembler programme

18.1 atmega8

18.1.1 led out

```
;; (c) david vajda
;; 10/31/2026
;; m8/atmega8/portd,led out

.include "m8def.inc"

ldi r16, HIGH (RAMEND)
out SPH, r16
```

```
ldi r16, LOW (RAMEND)
ldi r16, 0xff
out DDRB, r16
ldi r16, 0b01011100
out PORTB, r16
main: rjmp main
```

18.2 nasm

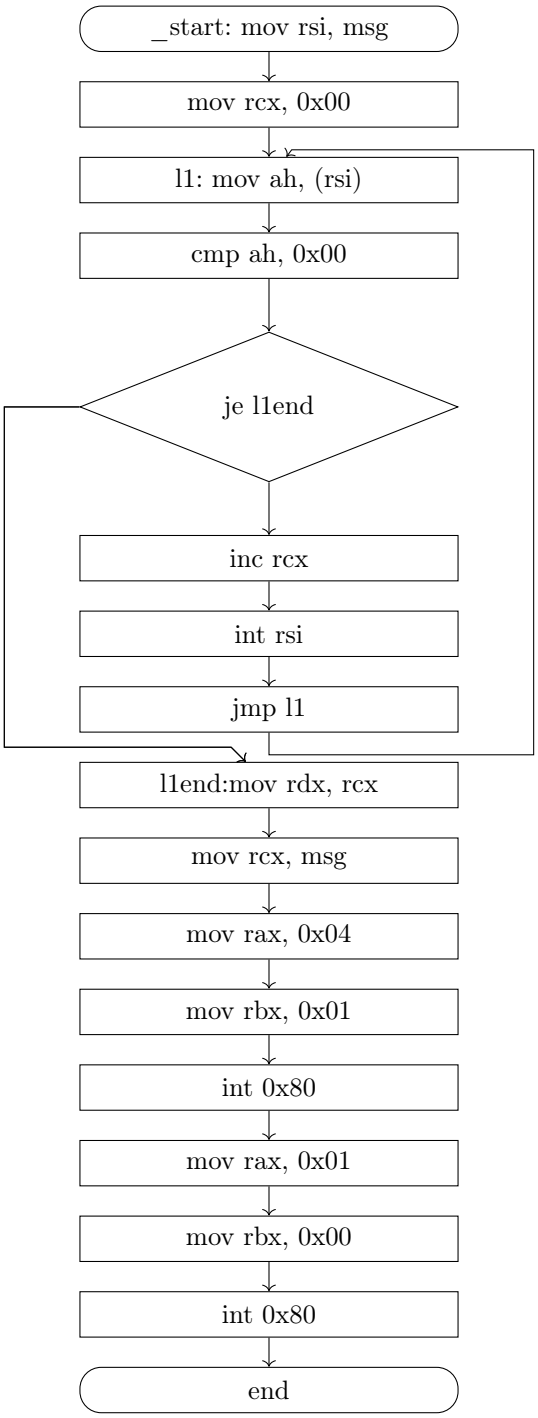
```
;; (c) david vajda
;; 10/31/2026
;; nasm assembler code i586

global _start
section .data
 msg:  db  "david@https://www.vajda-breadboard.de", 10, 13, 0
section .text
_start:
 mov rsi, msg
 mov rcx, 0x00
l1: mov ah, [rsi]
 cmp ah, 0x00
 je l1end
 inc rcx
 int rsi
 jmp l1
l1end:

 mov rdx, rcx
 mov rcx, msg
 mov rax, 0x04
 mov rbx, 0x01
 int 0x80

 mov rax, 0x01
 mov rbx, 0x00
 int 0x80
```

0.6cm,



19 matrizenrechnung

$$m = 4, n = 4, s = 4$$

$$\begin{pmatrix} 9 & 0 & 9 & 9 \\ 9 & 6 & 0 & 4 \\ 8 & 3 & 0 & 6 \\ 7 & 0 & 8 & 5 \end{pmatrix} \begin{pmatrix} 7 & 8 & 3 & 4 \\ 7 & 8 & 9 & 6 \\ 1 & 2 & 7 & 7 \\ 1 & 2 & 4 & 3 \end{pmatrix} = \begin{pmatrix} 81 & 108 & 126 & 126 \\ 109 & 128 & 97 & 84 \\ 83 & 100 & 75 & 68 \\ 62 & 82 & 97 & 99 \end{pmatrix}$$

```
M[1,1] = 9*1 + 3*5 + 3*4 + 6*9
= 7*9 + 9*3  = 27 + 63 = 90
M [1,2] = 9*8 + 3*0 + 3*0 + 6*5
 = 72 + 30 = 102
M [1,3] = 9*8 + 3*5 + 3*5 + 6*8
 = 6*5 + 14*8 = 30+80+24 = 134 falsch!
 9*8 + 3*5 + 3*5 + 6 *8 =
  72 + 30 + 48 = 150
 M [1,4] = 9*1 + 3*4 + 3*4 + 6 *4
  = 12*4 + 9 = 48+9 = 57
  M [2,1] = 3*1 + 5*5 + 1*4 + 8* 9
  = 3 + 25 + 4 + 72 = 79 + 25 = 104
```

relationale Datenbanken:

$$\begin{pmatrix} (n_0, a_{00}, a_{01}, a_{02}, a_{03}) \\ (n_1, a_{10}, a_{11}, a_{12}, a_{13}) \\ (n_2, a_{20}, a_{21}, a_{22}, a_{23}) \\ (n_3, a_{30}, a_{31}, a_{32}, a_{33}) \\ (n_4, a_{40}, a_{41}, a_{42}, a_{43}) \\ (n_5, a_{50}, a_{51}, a_{52}, a_{53}) \\ (n_6, a_{60}, a_{61}, a_{62}, a_{63}) \end{pmatrix}$$

- dabei ist n_i der sogenannte primary key
- $(n_6, a_{60}, a_{61}, a_{62}, a_{63})$ ist eine so genannte relation, weil eine relation ist:
- $x = y$, $x \leq y$, $x \geq y$, was (x, y) unter bestimmten vorraussetzungen entspricht: reflexivität, transitivität, symmetrie/antisymmetrie
- in einem graphen sind (x, y) die paare kanten, wobei eher $e_0 = (v_0, v_1)$. wobei v_0 ein Knoten ist und v_1 ein Knoten und das geordnete Paar, eine Kante (Edge e_0) zwischen den knoten Vertices v_i, v_j bedeutet.
- relationale datenbanken enthalten n tupel, also $(r_n, r_{n-1}, \ldots, r_1, r_0)$.

an der stelle wiederum stellt sich die frage: kann man auch zwischen kanten eine kante herstellen?

$$\begin{aligned} V &= \{v_1, v_2, v_3, v_4\} && (323) \\ E &\subseteq \{V \times V\} && (324) \\ E_k &= \{(v_1, v_2), (v_3, v_1)\} && (325) \\ E_l &= \{(v_2, v_3), (v_1, v_4)\} && (326) \\ EE &= ((v_2, v_3), (v_1, v_4), (v_1, v_2), (v_3, v_1)) && (327) \\ & && (328) \end{aligned}$$

wenn man wiederum zwei tabellen in einer datenbank hat, ...

$$A = \begin{pmatrix} (n_0, a_{00}, a_{01}, a_{02}, a_{03}) \\ (n_1, a_{10}, a_{11}, a_{12}, a_{13}) \\ (n_2, a_{20}, a_{21}, a_{22}, a_{23}) \\ (n_3, a_{30}, a_{31}, a_{32}, a_{33}) \\ (n_4, a_{40}, a_{41}, a_{42}, a_{43}) \\ (n_5, a_{50}, a_{51}, a_{52}, a_{53}) \\ (n_6, a_{60}, a_{61}, a_{62}, a_{63}) \end{pmatrix}$$

$$B = \begin{pmatrix} (n_0, b_{00}, b_{10}, a_{20}, b_{02}, b_{30}) \\ (n_1, b_{01}, b_{11}, a_{21}, b_{12}, b_{31}) \\ (n_2, b_{02}, b_{12}, a_{22}, b_{22}, b_{32}) \\ (n_3, b_{03}, b_{13}, a_{23}, b_{32}, b_{33}) \\ (n_4, b_{04}, b_{14}, a_{24}, b_{42}, b_{34}) \\ (n_5, b_{05}, b_{15}, a_{25}, b_{52}, b_{35}) \\ (n_6, b_{06}, b_{16}, a_{26}, b_{62}, b_{36}) \end{pmatrix}$$

und nun können wir vielleicht paare Bilden

$$E = (A, B) \tag{329}$$
$$\tag{330}$$

oder verschiedene Joins

$$E_0 = A \times B \tag{331}$$
$$E_0 \subseteq A \times B \tag{332}$$
$$\tag{333}$$

neben joins, aber: wäre eben die idee, paare zwischen tabellen in datenbanken her zu stellen. vom ersten her, würde man natürlich an Kreuzprodukte denken, was ja auch nicht falsch wäre. oder untermengen. doch auch das gilt, bei Kanten bei Graphen, ... unabhängig davon liessen sich mit paaren beziehungen zwischen den tabellen einer sehr grossen datenbank herstellen und so in einem graph veranschaulichen. man könnte nun auch damit beginnen, eulertoureren artige dinger ...+ hamulton kreise, bäume. dies ist ja die aufgabe von ERM. Entity Relationship Modell, genauso wie UML und ASM-Diagrammen/PAP wieder eine andere darstellung von graphen. (Klassen)

- arithmetik: arithmos/arithmetikos: die zahl (griechisch): rechnen

  $$+, \cdot, -, \div$$

  . grundrechenoperationen
- arithmetische ausdruecke:

  $$(4 + 5) \cdot 3$$

  (BNF - Bakus Naur Form)
- keine algebraischen
- algebra (luise unger: arabisch: alchawizri) probieren auf arabisch zu schreiben

- Aliph - Lahm - ha - A - waw mit i -
- algebra: luise unger: vermutung david vajda:
- dasselbe, was arithmetik ist, nur mit tischen und stuehlen.
- warum? weil:

COMMENT matrizenrechnung, luise unger:
ok, matrizen sind relationale Datenbank aehnliche konstrukte -
dann kommt: graphen theorie:
graphentheorie tut im zimmer: wenn dame morgens baby versorgt und zum toster rennt damit der weg von toster eierkocher ... nicht mehrfach geschieht, hamilton kreise, bewiesen durch euler.
mathematiker: nummer fuer tisch, stuhl ... aber bei eisenbahn: schwabenlaendle, und vermutete ergaenzung

england: marx: steht: breadfordshire, bukinghamshire, whorchestershire... (verbessern)

und tuebingen stuttgart:
tuebingen kw - kfurt wannweil, reutlingen, mb - metzingen bempflingen - no - nuertingen oberboihingen - pest - plochingen esslingen stuttgart

wobei dreher moeglich: mb und no

ok: blos: hier nummer: eisenbahn verkehr macht klar: die wie vielte station - frage ans gedaechtnis:

ich weiss no und mb - vertausche img raphen

wahr ueberlegung: beim programmierung kommt dann kommentar rein!!! also hier auch auch ueberlegung: kommentar!

arithmetik: grundrechenoperationen ### COMMENT
algebra: arithmetik mit tischen und stuehlen?

arithmetik: grundrechenoperationen: bakus naur: expr term fact - BNF - verknuepfung

$$(4 + 3) \cdot 2$$

und algebra: dasselbe verknuepfen, nur mit tischen stuehlen...

dazu gehoerige rechengesetze.

weitere fragen:

- modell
- theorie
- methode/verfahren
- arithmetik
- algreba
- algorithmus
- maschine
- system
- automat
- schaltwerk

modell und theorie! Dringend 100 pro auswendig lernen, wortlaut

algebra: antwort auf wikipedia offen, keine ernsthafte antwort das Zusammenfügen gebrochener Teile: al-gabr, gleichung

- modell: muster, modulus mass
- modell: analoge darstellung eines originals
- theorie: spekulatives denken, durch interpretation von sachverhalten oder von aussagen, gewonnene erkenntnis

- theorie:
- klassische nationaloekonomie adam smith
- john maynard keynes
- milton friedman - monetarismus
- marxismus: sozialismus vs. kapitalismus
- mikrooekonomie: haushaltstheorie (PHP + M (MariaDB)) - Preistheorie (?) - Produktionskostentheorie - Haushaltstheorie - Marktversagen
- Finanzwirtschaft: finanztheorie, kapitalmarktheorie??

gesellschaftliches modell: sozialismus/kapitalismus freie marktwirtschaft/planwirtschaft
jetzt normal weiter...

methode: Eine Methode (von altgriechisch méthodos – „Weg zu etwas hin“) ist ein planmäßiges, strukturiertes Verfahren, um ein bestimmtes Ziel zu erreichen oder ein Problem zu lösen

methode: weg zu etwas hin

- planmaessig
- strukturiert
- Verfahren

nächstes:

- Verfahren/Methode
- Methode: verfahren: planmaessig, strukturiert

problem: Ein Problem (altgriechisch ... próblema, „Vorsprung, Klippe, Hindernis; das, was zur Lösung vorgelegt wurde

- kosmos - arithmetikos:
- kosmos: ordnung
- mehr muessen wir nicht, folgen wird:
- linear
- konsekutiv
- sequentiell

Universum = kosmos.

Als System (altgriechisch systema „aus mehreren Einzelteilen zusammengesetztes Ganzes“) wird etwas bezeichnet, dessen Struktur aus verschiedenen Komponenten mit unterschiedlichen Eigenschaften besteht, die aufgrund bestimmter geordneter und funktionaler Beziehungen untereinander als gemeinsames Ganzes betrachtet werden (können) und so von Anderem abgrenzbar sind.

Eine Maschine (entlehnt aus französisch machine, von lateinisch machina, dieses von altgriechisch ... mechane, deutsch ‚Werkzeug‘, ‚künstliche Vorrichtung‘ist ein technisches Gebilde mit durch ein Antriebssystem bewegten Teilen. Maschinen werden als technische Arbeitsmittel vor allem für mechanische Einwirkung verwendet.

jetzt normal weiter.

```
(C) David Vajda
Mon Nov  2 10:09:53 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    0
 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
```

```
 4 0 1 0 0    0
 5 0 1 0 1    0
 6 0 1 1 0    1
 7 0 1 1 1    1
 8 1 0 0 0    0
 9 1 0 0 1    0
10 1 0 1 0    0
11 1 0 1 1    0
12 1 1 0 0    1
13 1 1 0 1    1
14 1 1 1 0    1
15 1 1 1 1    0


 1 0 0 0 1    1
 2 0 0 1 0    1
 3 0 0 1 1    1
 6 0 1 1 0    1
 7 0 1 1 1    1
12 1 1 0 0    1
13 1 1 0 1    1
14 1 1 1 0    1

Gruppe 1:
 1 0 0 0 1    1
 2 0 0 1 0    1
Gruppe 2:
 3 0 0 1 1    1
 6 0 1 1 0    1
12 1 1 0 0    1
Gruppe 3:
 7 0 1 1 1    1
13 1 1 0 1    1
14 1 1 1 0    1

Jetzt kommt das verschmelzen.

algorithmus grob:

1.) sortieren + gruppenbildung
2.) zeile fuer zeile verschmelzen
3.) doppeltes ein mal schreiben
4.) wieder: sortieren + gruppenbildung
...

Gruppe 1:
 1 0 0 0 1    1
 2 0 0 1 0    1
Gruppe 2:
 3 0 0 1 1    1
```

```
 6 0 1 1 0    1
12 1 1 0 0    1
Gruppe 3:
 7 0 1 1 1    1
13 1 1 0 1    1
14 1 1 1 0    1

(1,3)  0 0 x 1
(2,3)  0 0 1 x
(2,6)  0 x 1 0
(3,7)  0 x 1 1
(6,7)  0 1 1 x
(6,14) x 1 1 0
(12,13) 1 1 0 x
(12,14) 1 1 x 0

sortieren:
(2,3)  0 0 1 x
(6,7)  0 1 1 x
(12,13) 1 1 0 x
(1,3)  0 0 x 1
(12,14) 1 1 x 0
(2,6)  0 x 1 0
(3,7)  0 x 1 1
(6,14) x 1 1 0

jetzt gruppen und wie mit 3 variablen:

Gruppe 1:
(2,3)  0 0 1 x
Gruppe 2:
(6,7)  0 1 1 x
(12,13) 1 1 0 x

Gruppe 1:
(1,3)  0 0 x 1
Gruppe 2:
(12,14) 1 1 x 0

Gruppe 1:
(2,6)  0 x 1 0
Gruppe 2:
(3,7)  0 x 1 1

Gruppe 2:
(6,14) x 1 1 0



Gruppe 1:
```

```
(2,3)  0 0 1 x
Gruppe 2:
(6,7)  0 1 1 x
(12,13) 1 1 0 x
(2,3,6,7)   0 x 1 x
(12,13)   1 1 0 x

Gruppe 1:
(1,3)    0 0 x 1
Gruppe 2:
(12,14)   1 1 x 0


Gruppe 1:
(2,6)  0 x 1 0
Gruppe 2:
(3,7)  0 x 1 1
(2,6,3,7)   0 x 1 x

Gruppe 2:
(6,14)   x 1 1 0




(2,3,6,7)   0 x 1 x
(12,13)   1 1 0 x
(1,3)    0 0 x 1
(12,14)   1 1 x 0
(2,6,3,7)   0 x 1 x
(6,14)   x 1 1 0


(2,3,6,7)   0 x 1 x
(12,13)   1 1 0 x
(1,3)    0 0 x 1
(12,14)   1 1 x 0
(6,14)   x 1 1 0


(2,3,6,7)   0 x 1 x
(12,13)   1 1 0 x
(1,3)    0 0 x 1
(12,14)   1 1 x 0
(6,14)   x 1 1 0

primimplikanten:
   1 2 3 6 7 12 13 14
(2,3,6,7)   1 1 1 1
```

```
(12,13)       1 1
(1,3)   1  1
(12,14)       1  1
(6,14)     1    1


   1 2 3 6 7 12 13 14
(2,3,6,7)   1 1 1 1    p
(12,13)       1 1  p
(1,3)   1  1      p
(12,14)       1  1 p
```

	1	2	3	6	7	12	13	14	
$(2,3,6,7)$		1	1	1	1				p
$(12,13)$						1	1		p
$(1,3)$	1		1						p
$(12,14)$						1		1	p

von diesen gleichungen und ihren variablen stammt das ab: dann machen wir daraus eine gleichung.

```
(2,3,6,7)   0 x 1 x
(12,13)   1 1 0 x
(1,3)    0 0 x 1
(12,14)   1 1 x 0

 y = /x3 * x1 +
   x3 * x2 * /x1 +
   /x3 * /x2 * x0 +
   x3 * x2 * /x0;
```

so, und jetzt kommt der galasm. fuer den gal16v8

```
gal16v8
module gal16v8quinednf11022026

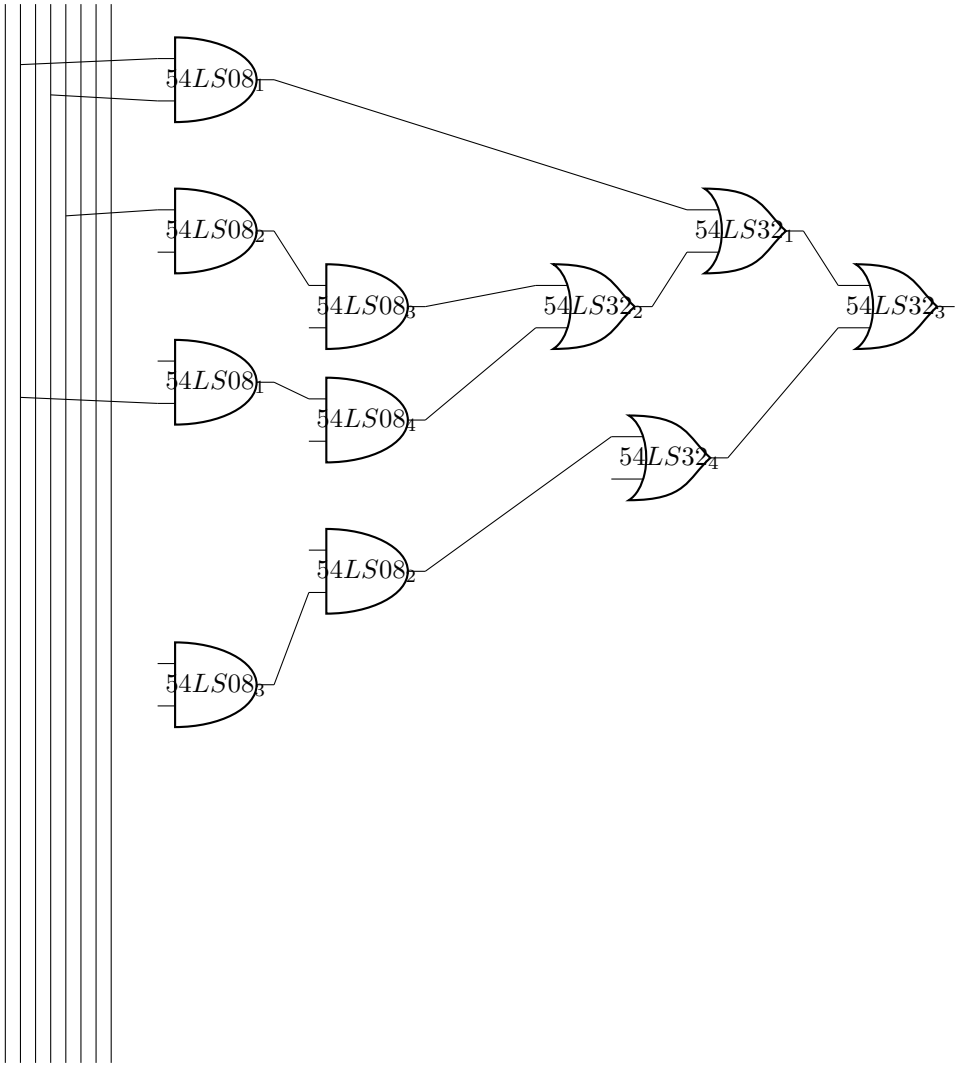
clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

 y = /x3 * x1 +
   x3 * x2 * /x1 +
   /x3 * /x2 * x0 +
   x3 * x2 * /x0;

description: quine16v8
```

usw.

jetzt kommt der schaltplan: ieee-1984/91


54LS08 1
54LS08 2
54LS08 3
54LS08 1
54LS08 4
54LS08 2
54LS08 3
54LS32 1
54LS32 2
54LS32 4
54LS32 3

$$
\begin{aligned}
l_i(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (334)\\
l_0(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (335)\\
&= \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} && (336)\\
l_1(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (337)\\
&= \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} && (338)\\
l_2(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (339)\\
&= \frac{x - x_2}{x_0 - x_2} \cdot \frac{x - x_1}{x_0 - x_1} && (340)\\
& && (341)
\end{aligned}
$$

$$X = (x_0, x_1, x_2)$$

$$X = \{x_0 = 0, x_1 = 1, x_2 = 2\}$$

$$
\begin{aligned}
l_i(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (342)\\
l_0(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (343)\\
&= \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} && (344)\\
&= \frac{x - 1}{0 - 1} \cdot \frac{x - 2}{0 - 2} && (345)\\
l_1(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (346)\\
&= \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} && (347)\\
&= \frac{x - 0}{1 - 0} \cdot \frac{x - 2}{1 - 2} && (348)\\
l_2(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (349)\\
&= \frac{x - x_2}{x_0 - x_2} \cdot \frac{x - x_1}{x_0 - x_1} && (350)\\
&= \frac{x - 2}{0 - 2} \cdot \frac{x - 1}{0 - 1} && (351)\\
& && (352)
\end{aligned}
$$

$$
\begin{aligned}
l_i(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (353)\\
l_0(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (354)\\
&= \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} && (355)\\
&= \frac{x - 1}{0 - 1} \cdot \frac{x - 2}{0 - 2} && (356)\\
&= \frac{x - 1}{-1} \cdot \frac{x - 2}{-2} && (357)\\
&= -\frac{x - 1}{\cdot}(-\frac{x}{2} + 1) && (358)\\
l_1(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (359)\\
&= \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} && (360)\\
&= \frac{x - 0}{1 - 0} \cdot \frac{x - 2}{1 - 2} && (361)\\
&= \frac{x - 0}{1} \cdot \frac{x - 2}{-1} && (362)\\
&= \frac{x}{1} \cdot -(x - 2) && (363)\\
l_2(x) &= \prod_{i=0}^{2} \frac{x - x_j}{x_i - x_j} && (364)\\
&= \frac{x - x_2}{x_0 - x_2} \cdot \frac{x - x_1}{x_0 - x_1} && (365)\\
&= \frac{x - 2}{-2} \cdot \frac{x - 1}{-1} && (366)\\
&= = \frac{x}{2} + 1 \cdot -(x - 1) && (367)\\
& && (368)
\end{aligned}
$$

jetzt kann man die einzeln ausrechnen und addieren.

$$\begin{aligned}
\frac{24}{4} &= \frac{1}{6} && (369)\\
&\quad \sum_{i=0}^{N} \frac{\left(f_i - \frac{N}{r}\right)^2}{\frac{N}{r}} && (370)\\
&= \frac{(1-6)^2}{6} && (371)\\
&+ \frac{(2-6)^2}{6} && (372)\\
&+ \frac{(3-6)^2}{6} && (373)\\
&+ \frac{(4-6)^2}{6} && (374)\\
&+ \frac{(5-6)^2}{6} && (375)\\
&+ \frac{(6-6)^2}{6} && (376)\\
& && (377)
\end{aligned}$$

hier nächster teil:

$$\begin{aligned}
\frac{24}{4} &= \frac{1}{6} && (378)\\
&\quad \sum_{i=0}^{N} \frac{\left(f_i - \frac{N}{r}\right)^2}{\frac{N}{r}} && (379)\\
&= \frac{(1)}{6} && (380)\\
&+ \frac{(-4)^2}{6} && (381)\\
&+ \frac{(-3)^2}{6} && (382)\\
&+ \frac{(-2)^2}{6} && (383)\\
&+ \frac{(-1)^2}{6} && (384)\\
&+ \frac{0^2}{6} && (385)\\
& && (386)
\end{aligned}$$

$$\frac{24}{4} = \frac{1}{6} \tag{387}$$

$$\sum_{i=0}^{N} \frac{\left(f_i - \frac{N}{r}\right)^2}{\frac{N}{r}} \tag{388}$$

$$= \frac{25}{6} \tag{389}$$

$$= 4.1666 \tag{390}$$

$$\frac{(-4)^2}{6} \tag{391}$$

$$= \frac{16}{6} \tag{392}$$

$$= 2.8333 \tag{393}$$

$$\frac{(-3)^2}{6} \tag{394}$$

$$= \frac{9}{6} \tag{395}$$

$$= 1.5 \tag{396}$$

$$\frac{(-2)^2}{6} \tag{397}$$

$$= \frac{4}{6} \tag{398}$$

$$= 0.66667 \tag{399}$$

$$\frac{(-1)^2}{6} \tag{400}$$

$$= 0.16666 \tag{401}$$

$$\frac{(0)^2}{6} \tag{402}$$

$$= 0.0000 \tag{403}$$

$$\tag{404}$$

$$9.17767 = 4.1666 + 2.8333 + 1.5 + 0.66667 + 0.16666 \tag{405}$$

$$\tag{406}$$

ok, Chi Quadrat Test tut nicht.

ok, doch er tut es: folgende begruendung:

ok, chi quadrat test tut nicht, bei wuerfel spielen. zunaechst vielleicht spaeter, und fuehrt zu einer näheren erklaerung mit dem Zufall.

doch! der chi quadrat macht unter umstaenden, dann einen sinn - wenn man:

modulo 6 nimmt

und man nimmt normale Zufallszahlen, wo es erwiesen ist. also zufallszahlen, die garantiert tut, aber mit dem modulo 6. und kann sie mit klassischen Dingern vergleichen.

noch eine moeglichkeit, bei 24 würfeln kam bei mir raus:

9

die argumentation widerspricht sich selber.
ich habe ihn ja schon durchgeführt. und das ergebnis war etwas um 9. etwas hoeher.

weil ich als r, 6 genommen, $r = 6$. und 24 würfe. machte 9.

mir ist aufgefallen, dass ich durchaus hoeher wuerfel. es gibt scheinbar Würfelspieler, die hoehere oder besser Ergebnisse erzielen

mir ist aufgefallen, dass meine Ergebnisse in vielen Zeiten den besseren gleichen. und: dies mal ist das ergebnis 9

ich vermute, dass: sich bei besonders grossen zahlen, von der relativen haeufigkeit zur wahrscheinlichkeit, das ergebnis sich 6 annaehern wuerde, naemlich dem wert r=6

hier das ergebnis.

20 eine rechenübung

ok, wir probieren es mit der hash tabelle. wir nehmen die eben erwuerfelten zahlen und stopfen sie rein.

$$
\begin{aligned}
E_{ges} &= (6,4,3,2,3,6,4,4,2,4,5,6,3,3,6,1,6,3,6,2,6,6,2,2) && (407)\\
E_{ges} &= \{1,2,2,2,2,2,3,3,3,3,3,4,4,4,4,5,6,6,6,6,6,6,6,6\} && (408)\\
H_1 &= |1| = 1 && (409)\\
H_2 &= |2,2,2,2,2| = 5 && (410)\\
H_3 &= |3,3,3,3,3| = 5 && (411)\\
H_4 &= |4,4,4,4| = 4 && (412)\\
H_5 &= |5| = 1 && (413)\\
H_6 &= |6,6,6,6,6,6,6,6| = 8 && (414)\\
& && (415)
\end{aligned}
$$

also, probieren wir es so. und zwar ohne wahrscheinlichkeit. der erwartungswert

(C) David Vajda

```
Tue Nov  3 07:32:33 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    0
 2 0 0 1 0    0
 3 0 0 1 1    0
 4 0 1 0 0    0
 5 0 1 0 1    1
 6 0 1 1 0    0
 7 0 1 1 1    0
 8 1 0 0 0    0
 9 1 0 0 1    0
10 1 0 1 0    0
11 1 0 1 1    0
12 1 1 0 0    0
13 1 1 0 1    0
14 1 1 1 0    0
15 1 1 1 1    0
```

so, nun erst mal ordentliche tabelle:

```
(C) David Vajda
Tue Nov  3 07:32:33 2026
4 Network - TTL - Disjunktive Normalform

\begin{tabular}{|c|c|c|c|c|c|}
\hline
$n$ & $x_3$ & $x_2$ & $x_1$ & $x_0$ & $y$\\
\hline
 $0$ $0$ $0$ $0$ $0$    $1$\\
 1 0 0 0 1    0
 2 0 0 1 0    0
 3 0 0 1 1    0
 4 0 1 0 0    0
 5 0 1 0 1    1
 6 0 1 1 0    0
 7 0 1 1 1    0
 8 1 0 0 0    0
 9 1 0 0 1    0
10 1 0 1 0    0
11 1 0 1 1    0
12 1 1 0 0    0
13 1 1 0 1    0
14 1 1 1 0    0
15 1 1 1 1    0
\end{tabular}

"ubungsaufgabe
\begin{verbatim}
sed
```

die aufgabe klappt mit sed bei mir in diesem falle nur halb so gut

```
cat sed.11032026.tmp.tex | sed 's/\(\s*\)/\$/g'
```

heraus kommt:

```
 cat sed.11032026.tmp.tex | sed 's/\(\s*\)/\$/g'
$$$n$$$ $&$ $$$x$_$3$$$ $&$ $$$x$_$2$$$ $&$ $$$x$_$1$$$ $&$ $$$x$_$0$$$ $&$ $$$y$$$\$\$
$\$h$l$i$n$e$
$ $$$0$$$ $$$0$$$ $$$0$$$ $$$0$$$ $$$0$$$ $ $ $ $$$1$$$\$\$
$ $1$ $0$ $0$ $0$ $1$ $ $ $ $0$
$ $2$ $0$ $0$ $1$ $0$ $ $ $ $0$
$ $3$ $0$ $0$ $1$ $1$ $ $ $ $0$
$ $4$ $0$ $1$ $0$ $0$ $ $ $ $0$
$ $5$ $0$ $1$ $0$ $1$ $ $ $ $1$
$ $6$ $0$ $1$ $1$ $0$ $ $ $ $0$
$ $7$ $0$ $1$ $1$ $1$ $ $ $ $0$
$ $8$ $1$ $0$ $0$ $0$ $ $ $ $0$
$ $9$ $1$ $0$ $0$ $1$ $ $ $ $0$
$1$0$ $1$ $0$ $1$ $0$ $ $ $ $0$
$1$1$ $1$ $0$ $1$ $1$ $ $ $ $0$
$1$2$ $1$ $1$ $0$ $0$ $ $ $ $0$
$1$3$ $1$ $1$ $0$ $1$ $ $ $ $0$
$1$4$ $1$ $1$ $1$ $0$ $ $ $ $0$
$1$5$ $1$ $1$ $1$ $1$ $ $ $ $0$%                                                    davidvajda
```

das, was sich nicht lösen lässt, daran, das möchte ich jetzt nicht, weil es ist ja wo anders eine aufgabe, also verzettele ich mich sonst. so: es erfüllt den zweck. mit einem weiteren ding: kann ich all das ersetzen, was noch zu ersetzen ist, drei dollars. und: die zwei mit white space in der mitte usw.

gut so geht es weiter:

```
davidvajda@localhost ~ % cat sed.11032026.tmp.tex | sed 's/\(\s*\)/\$/g' | sed 's/\$ \$ \$
$$$n$$$ $&$ $$$x$_$3$$$ $&$ $$$x$_$2$$$ $&$ $$$x$_$1$$$ $&$ $$$x$_$0$$$ $&$ $$$y$$$\$\$
$\$h$l$i$n$e$
$ $$$0$$$ $$$0$$$ $$$0$$$ $$$0$$$ $$$0$$$ & $$$1$$$\$\$
$ $1$ $0$ $0$ $0$ $1$ & $0$
$ $2$ $0$ $0$ $1$ $0$ & $0$
$ $3$ $0$ $0$ $1$ $1$ & $0$
$ $4$ $0$ $1$ $0$ $0$ & $0$
$ $5$ $0$ $1$ $0$ $1$ & $1$
$ $6$ $0$ $1$ $1$ $0$ & $0$
$ $7$ $0$ $1$ $1$ $1$ & $0$
$ $8$ $1$ $0$ $0$ $0$ & $0$
$ $9$ $1$ $0$ $0$ $1$ & $0$
$1$0$ $1$ $0$ $1$ $0$ & $0$
$1$1$ $1$ $0$ $1$ $1$ & $0$
$1$2$ $1$ $1$ $0$ $0$ & $0$
$1$3$ $1$ $1$ $0$ $1$ & $0$
```

```
$1$4$ $1$ $1$ $1$ $0$ & $0$
$1$5$ $1$ $1$ $1$ $1$ & $0$%
davidvajda@localhost ~ % cat sed.11032026.tmp.tex | sed 's/\(\s*\)/\$/g' | sed 's/\$ \$ \$
```

das müsste reichen und den rest korrigiere ich von alleine

n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

```
 cat sed.11032026.tmp.tex | sed 's/\(\s*\)/\$/g' | sed 's/\$ \$ \$/\$ \&/g' | sed 's/\$ \$
```

n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
5	0	1	0	1	1

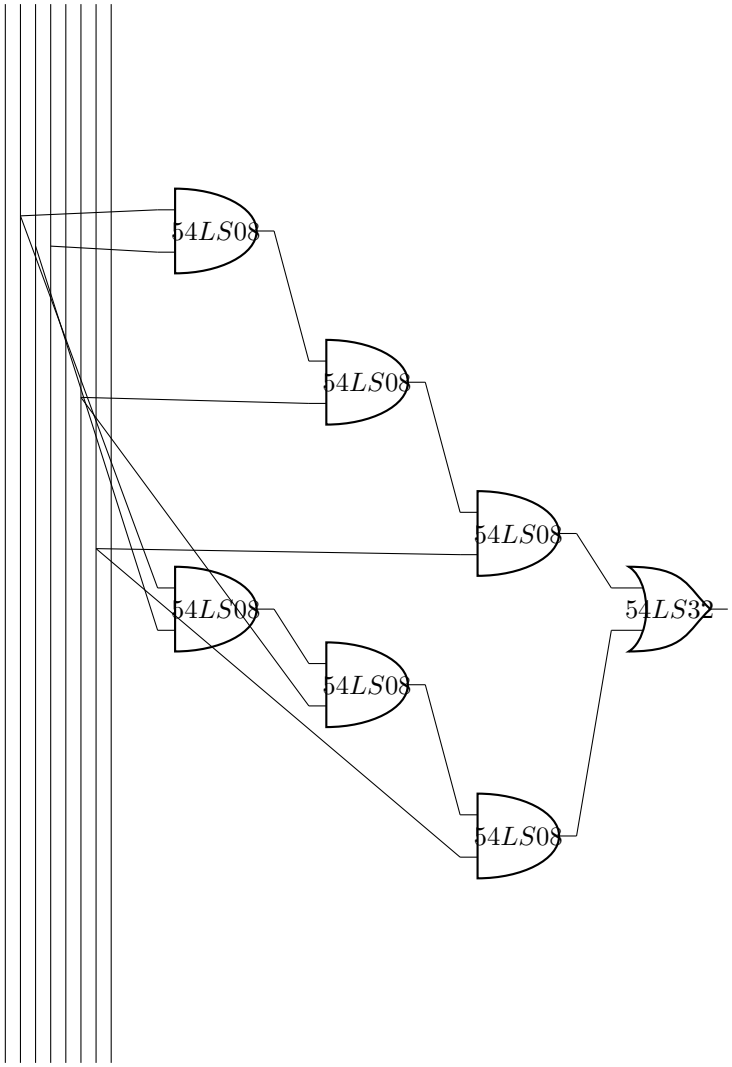
so hier gibt es nichts zu verkürzen.

```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```




```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```

21 fehler korrektur

n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
5	0	1	0	1	1

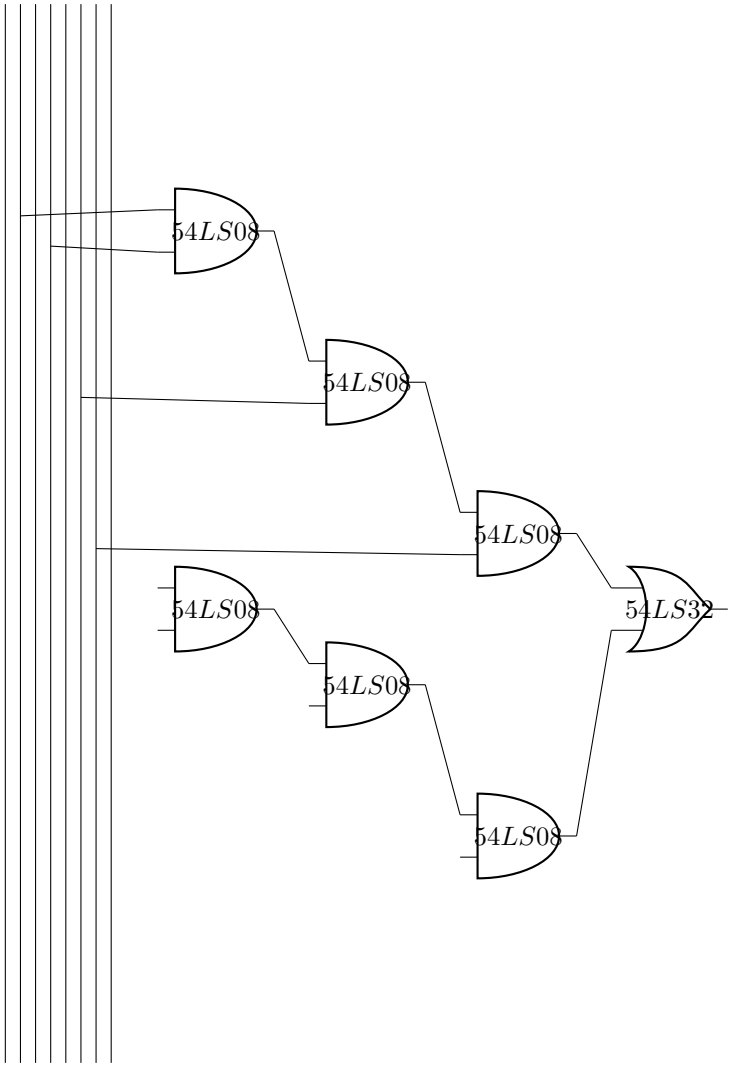
so hier gibt es nichts zu verkürzen.

```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```




```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```

das war falsch das letzte!
im leben kann man fehler leugnen! im leben kann man lügen. im labor nicht!
warum? Weil es sich nicht lohnt!
deswegen gibt es labore

Man kann im leben einfach lügen und es ist schwer das nicht zu machen! Aber im Labor lohnt sich das nicht! weil wenn man nicht alles kaputt gemacht und Fallgatter kommt rein und sieht was man da fuer eine schweinerei gemacht hat. dann: gehoeven im labor die peinlichsten fehler dazu, und man muss erkennen warum?
weil man das y fuer x hielt!
da war ein fehler drin! das ist richtig, den laesst man aber in der aufgabe, korrigiert es und schreibt die richtige version drunter, deswegen mathematische Texte, indem man naemlich das schreibt was man wirklich schreibt, ohne das näher zu erläutern.
stimmt doch nicht es war kein fehler drin!
dann gehoert das zu den Fehlern: der KGB - ist - also normale leute im KGB haben immer Blutdruck gemessen und wissen wie es dem volk, damit sie weiter mathe fuer das volk machen
was das volk nicht wollte, aber der KGB konnte das Volk unterdrücken, und war ein guter KGB, die Stasi konnte das Volk nicht unterdrücken und war eine schlechte Stasi
weil: die leute muessen lernen, dass die Leute nur aus Spass im KGB arbeiten. deswegen muesste man Blutdruck messen, wenn es nicht den Ärzten nahe, jetzt war ich naemlich aufgeregt.
dann habe ich einen fehler prophezeit, der nicht da ist
und doch ist er da. naemlich an anderer stelle.
die frage des Fehlers!

```
(A) -> (B) -> (C) -> (D) -> (E) -> (END)
```

aber wo ist der fehler.
das ergebnis ist falsch! und doch war die sache von anfang an richtig
wo ist der fehler. eingezeichnet auf Abszisse und ordinate!
im Rechenschritt, im algo schritt
und noch besser: im Arbeitsschritt!
und warum? weil hier ein fluechtigkeitsfehler ist, der rein muss, weil ich es korrigiere.
doch es ist einer da, doch er ist an anderer stelle
Parallelverarbeitung! text, da kann fehler rein, der computer muss kleine teile entlarven, hier ist das, was mensch nicht falsch, sondern im falle einer analyse muss computer verzeihen. das fuehrt zu weit, und er muss das als forensischen fehler behandeln, weil selbst hier, steckt was dahinter
so: das andere mathe ist teilweise mit ergebnis verbunden. stimmt aber an anderer stelle nicht: weil: der Arbeitsschritt. der Arbeitsschritt hat nicht parallel viele Produkte
man kann parallel sprechen, viele probleme, anleiern, wenn man das nicht kann man nichts konkret planen. daraus muss man sich so einen wert errechnen, aber jetzt rede ich zu viel.
das ergebnis ist scheisse, und es ist es naturbedingt
weil wenn sie in der Walachei sind, fahren sie mit ihren Auto anders, wie in den niederlanden
Hauptsache es findet noch statt:
ergebnis: fuer die Wissenschaft, osteuropäischen autos glaubt keiner
aber ich mache weiter. das ist nicht relevant. sondern die aufgabe, Schwierigkeiten sind zu begreifen, wie fährt man durch die Walachei.

wie macht man labor in der Walachei, natuerlich nicht anders wie hier, nur nicht stehen bleiben.
gut dann macht Daimler neue autos. wir sind nicht die chefs von Daimler, Daimler hat sich dran gewöhnt. also baut es autos fuer die Walachei. aber jetzt beschreibe nicht das volkloristische, sondern den Gegenstand selber, das objekt - was produziert. objekt Subjekt und nicht Subjekte beschreibt Subjekte, ausser dem Arbeitsprozess.

n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
5	0	1	0	1	1

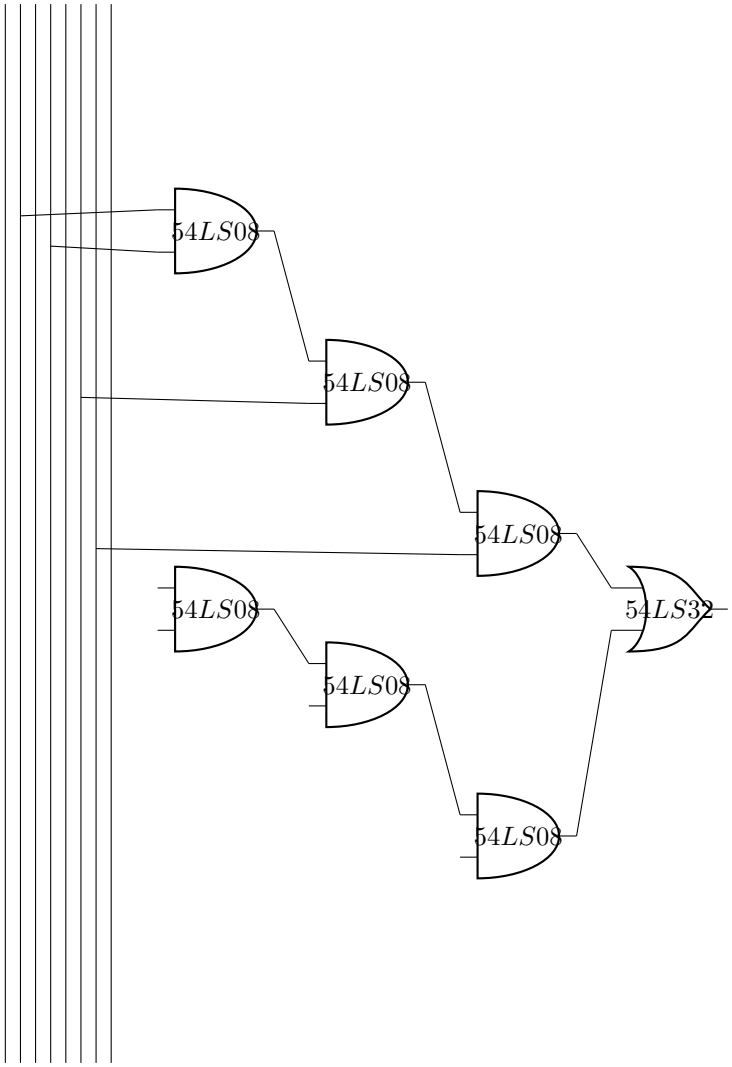
so hier gibt es nichts zu verkürzen.

```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```




```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```

22 fehler korrektur

n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
5	0	1	0	1	1

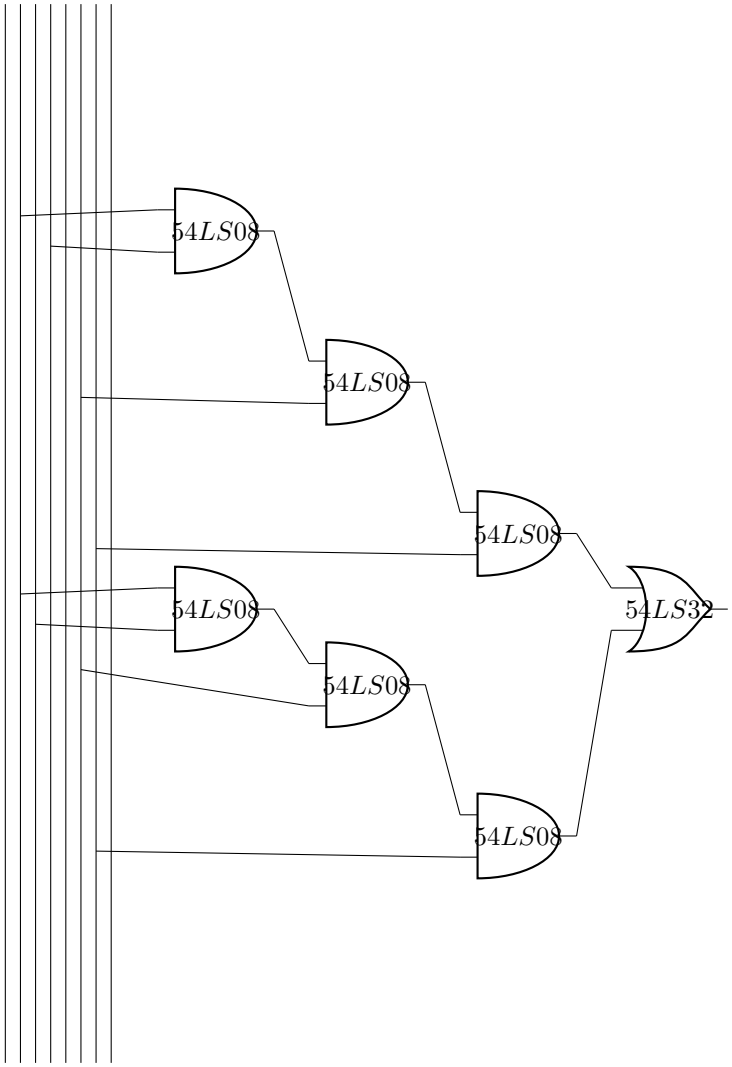
so hier gibt es nichts zu verkürzen.

```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * /x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```




```
GAL16V8
MODULE

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x3 * /x2 * /x1 * /x0 +
  /x3 * x2 * /x1 * x0

DESCRIPTION:
das hier ist die gleichung.
```

23 jetzt geht es weiter

chi quadrat test + wir machen atmega8.

$$
\begin{aligned}
E_{ges} &= (6,4,3,2,3,6,4,4,2,4,5,6,3,3,6,1,6,3,6,2,6,6,2,2) && (416)\\
E_{ges} &= \{1,2,2,2,2,2,3,3,3,3,3,4,4,4,4,5,6,6,6,6,6,6,6,6\} && (417)\\
H_1 &= |1| = 1 && (418)\\
H_2 &= |2,2,2,2,2| = 5 && (419)\\
H_3 &= |3,3,3,3,3| = 5 && (420)\\
H_4 &= |4,4,4,4| = 4 && (421)\\
H_5 &= |5| = 1 && (422)\\
H_6 &= |6,6,6,6,6,6,6,6| = 8 && (423)\\
& && (424)
\end{aligned}
$$

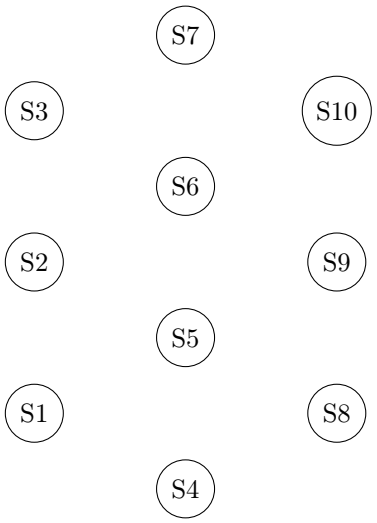
relative Häufigkeiten.

$$
\begin{aligned}
E_{ges} &= (6,4,3,2,3,6,4,4,2,4,5,6,3,3,6,1,6,3,6,2,6,6,2,2) && (425)\\
E_{ges} &= \{1,2,2,2,2,2,3,3,3,3,3,4,4,4,4,5,6,6,6,6,6,6,6,6\} && (426)\\
h(E_1) &= |1| = 1 && (427)\\
h(E_2) &= |2,2,2,2,2| = 5 && (428)\\
h(E_3) &= |3,3,3,3,3| = 5 && (429)\\
h(E_4) &= |4,4,4,4| = 4 && (430)\\
h(E_5) &= |5| = 1 && (431)\\
h(E_6) &= |6,6,6,6,6,6,6,6| = 8 && (432)\\
& && (433)
\end{aligned}
$$

so jetzt hier relative häufigkeit.

$$
\begin{aligned}
E_{ges} &= (6,4,3,2,3,6,4,4,2,4,5,6,3,3,6,1,6,3,6,2,6,6,2,2) && (434)\\
E_{ges} &= \{1,2,2,2,2,2,3,3,3,3,3,4,4,4,4,5,6,6,6,6,6,6,6,6\} && (435)\\
h(E_1) &= \frac{H_1}{N} && (436)\\
&= |1| = 1 && (437)\\
h(E_2) &= |2,2,2,2,2| = 5 && (438)\\
h(E_2) &= \frac{H_2}{N} && (439)\\
h(E_2) &= \frac{5}{24} && (440)\\
h(E_3) &= |3,3,3,3,3| = 5 && (441)\\
h(E_3) &= \frac{H_3}{N} && (442)\\
h(E_4) &= \frac{5}{24} && (443)\\
h(E_4) &= |4,4,4,4| = 4 && (444)\\
h(E_4) &= \frac{H_4}{N} && (445)\\
h(E_4) &= \frac{4}{24} && (446)\\
h(E_5) &= |5| = 1 && (447)\\
h(E_5) &= \frac{H_5}{N} && (448)\\
h(E_5) &= \frac{1}{24} && (449)\\
h(E_6) &= |6,6,6,6,6,6,6,6| = 8 && (450)\\
h(E_6) &= \frac{8}{24} && (451)\\
h(E_6) &= \frac{8}{24} && (452)\\
& && (453)
\end{aligned}
$$

so, jetzt geht es hier weiter. ich glaube, vorher kommt der automat.



erst mal die knoten verbinden, nach welchem algorithmus, such dir nachbarn. was ist die optimalere loesung ohne sie an zu wenden, wir verbinden alle mit allen. danach entfernen wir einzeln die nachbarn bis das bild stimmt, jeder knoten hat eine gerade valenz, das heisst, eine gerade anzahl an nachbarn.
und: damit ist eindeutig ein graph nur mit kreisen gegeben, weil, wenn jeder eine gerade valenz hat, dann ist der graph voller zyklen und nicht zykelfrei sprich eulertour.

dazu habe ich keine antwort die nicht langweilig ist: frage soll ich die langeweile, ja/nein? antwort ja.

alle aufschreiben. zuneachst mit allen knoten und jedem schritt, die tabelle stimmt im äusseren nicht, aber: dadurch wird eine lange liste, vielleicht lernen wir es spaeter.

wir schreiben auf, wie geht das verfahren. 1 schritt

wir machen lauter permutationen und so weiter. das heisst, wir nehmen das erste element raus. damit das einfacher geht. schreiben wir alle Zahlen von 1 ... 10 auf. 10 mal

```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
```

jetzt geht das hier weiter: jetzt kommt jeweils das erste element raus und umgetauscht:

```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
2, 1, 3, 4, 5, 6, 7, 8, 9, 10
3, 1, 2, 4, 5, 6, 7, 8, 9, 10
4, 1, 2, 3, 5, 6, 7, 8, 9, 10
```

```
5, 1, 2, 3, 4, 6, 7, 8, 9, 10
6, 1, 2, 3, 4, 5, 7, 8, 9, 10
7, 1, 2, 3, 4, 5, 6, 8, 9, 10
8, 1, 2, 3, 4, 5, 6, 7, 9, 10
9, 1, 2, 3, 4, 5, 6, 7, 8, 10
10, 1, 2, 3, 4, 5, 6, 7, 8, 9
```

so jetzt stimmt das. jetzt kommen einzelne dinger raus. so nach zufall notiert. von oben bis unten immer durch ja und nein.

```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (1.)
2, 1, 3, 4, 5, 6, 7, 8, 9, 10 (2.)
3, 1, 2, 4, 5, 6, 7, 8, 9, 10 (3.)
4, 1, 2, 3, 5, 6, 7, 8, 9, 10 (4.)
5, 1, 2, 3, 4, 6, 7, 8, 9, 10 (5.)
6, 1, 2, 3, 4, 5, 7, 8, 9, 10 (6.)
7, 1, 2, 3, 4, 5, 6, 8, 9, 10 (7.)
8, 1, 2, 3, 4, 5, 6, 7, 9, 10 (8.)
9, 1, 2, 3, 4, 5, 6, 7, 8, 10 (9.)
10, 1, 2, 3, 4, 5, 6, 7, 8, 9 (10.)
```

das ist zur veranschaulichung, und jetzt kommt die aufzeichung als graph und zustände.

```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (1.)
2, 1, 3, 4, 5, 6, 7, 8, 9, 10 (2.)
3, 1, 2, 4, 5, 6, 7, 8, 9, 10 (3.)
4, 1, 2, 3, 5, 6, 7, 8, 9, 10 (4.)
5, 1, 2, 3, 4, 6, 7, 8, 9, 10 (5.)
6, 1, 2, 3, 4, 5, 7, 8, 9, 10 (6.)
7, 1, 2, 3, 4, 5, 6, 8, 9, 10 (7.)
8, 1, 2, 3, 4, 5, 6, 7, 9, 10 (8.)
9, 1, 2, 3, 4, 5, 6, 7, 8, 10 (9.)
10, 1, 2, 3, 4, 5, 6, 7, 8, 9 (10.)
```

```
(1.) -> (2.) -> (3.) -> (4.) -> (5.) -> (6.) -> (7.) -> (8.) -> (9.) -> (10.)
```

statt zahl punkt, kann zustand sein, das ist aber keine loesung, sondern: trotzdem eine. ich gehe runter, sage immer ja nein. das ist so zu sagen, zeilenweise verkuerzen. dann gehe von rechts nach links, durch und nehme entsprechend spaltenweise zustaende raus. im naechsten schritt wie bei mealy und moore kommt die sache so zu machen dass sie passt. z.B.: kein gerichter Graph,

$$(4, 5) \Rightarrow (5, 4)$$

andere darstellung

```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (S1)
2, 1, 3, 4, 5, 6, 7, 8, 9, 10 (S2)
3, 1, 2, 4, 5, 6, 7, 8, 9, 10 (S3)
4, 1, 2, 3, 5, 6, 7, 8, 9, 10 (S4)
```

```
5, 1, 2, 3, 4, 6, 7, 8, 9, 10 (S5)
6, 1, 2, 3, 4, 5, 7, 8, 9, 10 (S6)
7, 1, 2, 3, 4, 5, 6, 8, 9, 10 (S7)
8, 1, 2, 3, 4, 5, 6, 7, 9, 10 (S8)
9, 1, 2, 3, 4, 5, 6, 7, 8, 10 (S9)
10, 1, 2, 3, 4, 5, 6, 7, 8, 9 (S10)

(S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)

(1, 2, 3, 4, 5, 6, 7, 8, 9, 10), (S1)
(2, 1, 3, 4, 5, 6, 7, 8, 9, 10), (S2)
(3, 1, 2, 4, 5, 6, 7, 8, 9, 10), (S3)
(4, 1, 2, 3, 5, 6, 7, 8, 9, 10), (S4)
(5, 1, 2, 3, 4, 6, 7, 8, 9, 10), (S5)
(6, 1, 2, 3, 4, 5, 7, 8, 9, 10), (S6)
(7, 1, 2, 3, 4, 5, 6, 8, 9, 10), (S7)
(8, 1, 2, 3, 4, 5, 6, 7, 9, 10), (S8)
(9, 1, 2, 3, 4, 5, 6, 7, 8, 10), (S9)
(10, 1, 2, 3, 4, 5, 6, 7, 8, 9), (S10)

(S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)

(S1) ja -> (S2) ja -> (S3) -> (S4) -> (S5) -> ja (S6) -> (S7) -> (S8) -> (S9) -> (S10)
(S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
(S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
(S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
```

jetzt einfach der reihe nach, noch mal. wenn das so geht, nehme ich einfach eine binaere zahl und sage darueber ja und nein. ok, bessere loesung, ich schreibe ueberall dahinter, 2, 4, 6. eines von den dingern. nach den zufallszahlen.
egal.

```
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10), (S1), 2
(2, 1, 3, 4, 5, 6, 7, 8, 9, 10), (S2), 3, 4
(3, 1, 2, 4, 5, 6, 7, 8, 9, 10), (S3), 1, 4
(4, 1, 2, 3, 5, 6, 7, 8, 9, 10), (S4), 3, 4
(5, 1, 2, 3, 4, 6, 7, 8, 9, 10), (S5), 1, 2
(6, 1, 2, 3, 4, 5, 7, 8, 9, 10), (S6), 1, 2
(7, 1, 2, 3, 4, 5, 6, 8, 9, 10), (S7), 3, 3, 6
(8, 1, 2, 3, 4, 5, 6, 7, 9, 10), (S8), 4
(9, 1, 2, 3, 4, 5, 6, 7, 8, 10), (S9), 4
(10, 1, 2, 3, 4, 5, 6, 7, 8, 9), (S10), 5, 3, 6

(2,4,4,4,2,2,6,4,4,6)

das wurde erwuerfelt und jetzt entsprechend streichen.
(S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
(S2) -> (S1) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
(S3) -> (S1) -> (S2) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
(S4) -> (S1) -> (S2) -> (S3) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
(S5) -> (S1) -> (S2) -> (S3) -> (S4) -> (S6) -> (S7) -> (S8) -> (S9) -> (S10)
```

```
(S6) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S7) -> (S8) -> (S9) -> (S10)
(S7) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) ->  (S8) -> (S9) -> (S10)
(S8) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S9) -> (S10)
(S9) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S10)
(S10) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (S7) -> (S8) -> (S9)

(2,4,4,4,2,2,6,4,4,6)
(S1) -> (S2) -> (S3) -> (S4) -> (S5) -> (S6) -> (2 mal wuerfeln), 4, 3
(S7) -> (S8) -> (S9) -> (S10) ;;  (2 mal wuerfeln)
(S2) -> (S1) -> (S3) -> (S4) -> (S5) -> (S6) ->
(S7) -> (S8) -> (S9) -> (S10)
(S3) -> (S1) -> (S2) -> (S4) -> (S5) -> (S6) ->
(S7) -> (S8) -> (S9) -> (S10)
(S4) -> (S1) -> (S2) -> (S3) -> (S5) -> (S6) ->
(S7) -> (S8) -> (S9) -> (S10)
(S5) -> (S1) -> (S2) -> (S3) -> (S4) -> (S6) ->
(S7) -> (S8) -> (S9) -> (S10)
(S6) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) ->
(S7) -> (S8) -> (S9) -> (S10)
(S7) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) ->
(S6) ->  (S8) -> (S9) -> (S10)
(S8) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) ->
(S6) -> (S7) -> (S9) -> (S10)
(S9) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) ->
(S6) -> (S7) -> (S8) -> (S10)
(S10) -> (S1) -> (S2) -> (S3) -> (S4) -> (S5) ->
 (S6) -> (S7) -> (S8) -> (S9)

 ok, das ist falsch, wir muessen mod 10 rechnen. und: wir muessen immer 2 wuerfe machen.
 gelb, blau, ist am besten. blau ist 10^1 und gelb 10^0

 so und jetzt:
 jetzt muessen wir:
 (2 + 4 + 4 + 4 + 2 + 2 + 6 + 4 + 4 + 6) x 2
 wuerfeln
 ((2 + 4) + (4 + 4) + (2 + 2) + (6 + 4) + (4 + 6)) x 2
 =  ((6) + (8) + (4) + (10) + (10)) x 2
 =  ((6 + 8) + (4 + 10) + 10) x 2
 =  ((14) + (14) + 10) x 2
 = 38 x 2 = 76

 76 mal wuerfeln, aber das stimmt wegen gelb und blau. bequemer ist serialized.

2, 4, 1, 3 (4)
1, 1, 3, 1 (4)
5, 2, 3, 1 (4)
4, 6, 4, 1, 4, 5, 2, 6 (8)
6, 2, 1, 4, 5, 1, 5, 5 (8)
2, 4, 4, 3, 1, 6, 4, 5 (8)
1, 1, 1, 3, 5, 5, 6, 1 (8)
```

```
32 + 12 = 44

1, 1, 3, 1 (4)
5, 2, 3, 1 (4)
4, 6, 4, 1, 4, 5, 2, 6 (8)
6, 2, 1, 4, 5, 1, 5, 5 (8)
2, 4, 4, 3, 1, 6, 4, 5 (8)
1, 1, 1, 3, 5, 5, 6, 1 (8)


1 * 10^1 + 1 * 10^0, 3 * 10^1 + 1 * 10^0 (4)
5 * 10^1 + 2 * 10^0, 3 * 10^1 + 1 * 10^0 (4)
4 * 10^1 + 6 * 10^0 , 4 * 10^1 + 1 * 10^0, 4 * 10^1 + 5 * 10^0, 2 * 10^1 + 6 * 10^0 (8)
6 * 10^1 + 2 * 10^0 , 1 * 10^1 + 4 * 10^0, 5 * 10^1 + 1 * 10^0, 5 * 10^1 + 5 * 10^0 (8)
2 * 10^1 + 4 * 10^0 , 4 * 10^1 + 3 * 10^0, 1 * 10^1 + 6 * 10^0, 4 * 10^1 + 5 * 10^0 (8)
1 * 10^1 + 1 * 10^0 , 1 * 10^1 + 3 * 10^0, 5 * 10^1 + 5 * 10^0, 6 * 10^1 + 1 * 10^0 (8)

11 31
52 31
46 41 45 26
62 14 51 55
24 43 16 45
11 13 55 61

11 mod 10 = 1, 31 mod 10 = 1
52 mod 10 = 2, 31 mod 10 = 1
46 mod 10 = 6, 41 mod 10 = 1, 45 mod 10 = 5, 26 mod 10 = 6
62 mod 10 = 2, 14 mod 10 = 4, 51 mod 10 = 1, 55 mod 10 = 5
24 mod 10 = 4, 43 mod 10 = 3, 16 mod 10 = 6, 45 mod 10 = 5
11 mod 10 = 1, 13 mod 10 = 3, 55 mod 10 = 5, 61 mod 10 = 1

1, 1, 2, 1, 6, 1, 5, 6, 2, 4, 1, 5, 4, 3, 6, 5, 1, 3, 5, 1

 (C) David Vajda
Wed Nov  4 14:01:45 2026
4 Network - TTL - Disjunktive Normalform

 0 0 0 0 0    1
 1 0 0 0 1    0
 2 0 0 1 0    1
 3 0 0 1 1    0
 4 0 1 0 0    0
 5 0 1 0 1    0
 6 0 1 1 0    1
 7 0 1 1 1    1
 8 1 0 0 0    1
 9 1 0 0 1    0
10 1 0 1 0    1
11 1 0 1 1    1
```

```
12 1 1 0 0    0
13 1 1 0 1    0
14 1 1 1 0    1
15 1 1 1 1    0
```

so, jetzt normale tabelle:

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	0

quine mc reduzierung.

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
2	0	0	1	0	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
10	1	0	1	0	1
11	1	0	1	1	1
14	1	1	1	0	1

2. algorithmischer schritt, nach . . . gruppieren.

(C) David Vajda
Wed Nov 4 14:01:45 2026
4 Network - TTL - Disjunktive Normalform

i	x_3	x_2	x_1	x_0	y
gruppe 0					
0	0	0	0	0	1
gruppe 1					
2	0	0	1	0	1
8	1	0	0	0	1
gruppe 2					
6	0	1	1	0	1
10	1	0	1	0	1
gruppe 3					
7	0	1	1	1	1
11	1	0	1	1	1
14	1	1	1	0	1

2. algorithmischer schritt, verschmelzen.

(C) David Vajda
Wed Nov 4 14:01:45 2026
4 Network - TTL - Disjunktive Normalform

i	x_3	x_2	x_1	x_0	y
gruppe (0,1)					
$(0, 2)$	0	0	x	0	1
$(0, 8)$	x	0	0	0	1
gruppe (1,2)					
$(2, 6)$	0	x	1	0	1
$(2, 10)$	x	0	1	0	1
$(8, 10)$	x	0	1	0	1
gruppe (2,3)					
$(6, 7)$	0	1	1	x	1
$(6, 14)$	x	1	1	0	1
$(10, 11)$	1	0	1	x	1
$(10, 15)$	1	x	1	0	1

nächster algorithmischer schritt, doppelte vergleichen, gibt noch keine - auch noch PAP - jetzt weiter. weiter verschmelzen, sortieren. + gruppierung neu benennen

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
(x_2, x_1, x_0)					
$(0, 8)$	x	0	0	0	1
$(2, 10)$	x	0	1	0	1
$(8, 10)$	x	0	1	0	1
$(6, 14)$	x	1	1	0	1
(x_3, x_1, x_0)					
$(2, 6)$	0	x	1	0	1
$(10, 15)$	1	x	1	0	1
(x_3, x_2, x_0)					
$(0, 2)$	0	0	x	0	1
(x_3, x_2, x_1)					
$(6, 7)$	0	1	1	x	1
$(10, 11)$	1	0	1	x	1

nächster algo schritt, wieder verschmelzen. erst gruppenbildung!

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
(x_2, x_1, x_0)					
$(0, 8)$	x	0	0	0	1
$(2, 10)$	x	0	1	0	1
$(8, 10)$	x	0	1	0	1
$(6, 14)$	x	1	1	0	1
(x_3, x_1, x_0)					
$(2, 6)$	0	x	1	0	1
$(10, 15)$	1	x	1	0	1
(x_3, x_2, x_0)					
$(0, 2)$	0	0	x	0	1
(x_3, x_2, x_1)					
$(6, 7)$	0	1	1	x	1
$(10, 11)$	1	0	1	x	1

da war ein fehler! $(8, 10)$.

so richtig, gut korrigiert, nicht alles von vorne, autokorrektur beim computer

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
(x_2, x_1, x_0)					
$(0, 8)$	x	0	0	0	1
$(2, 10)$	x	0	1	0	1
$(8, 10)$	1	0	x	0	1
$(6, 14)$	x	1	1	0	1
(x_3, x_1, x_0)					
$(2, 6)$	0	x	1	0	1
$(10, 15)$	1	x	1	0	1
(x_3, x_2, x_0)					
$(0, 2)$	0	0	x	0	1
(x_3, x_2, x_1)					
$(6, 7)$	0	1	1	x	1
$(10, 11)$	1	0	1	x	1

demnach umsortieren, falsche stelle!

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
(x_2, x_1, x_0)					
Gruppe 0					
$(0, 8)$	x	0	0	0	1
Gruppe 1					
$(2, 10)$	x	0	1	0	1
Gruppe 2					
$(6, 14)$	x	1	1	0	1
(x_3, x_1, x_0)					
Gruppe 1					
$(2, 6)$	0	x	1	0	1
Gruppe 2					
$(10, 15)$	1	x	1	0	1
(x_3, x_2, x_0)					
Gruppe 0					
$(0, 2)$	0	0	x	0	1
Gruppe 1					
$(8, 10)$	1	0	x	0	1
(x_3, x_2, x_1)					
Gruppe 2					
$(6, 7)$	0	1	1	x	1
$(10, 11)$	1	0	1	x	1

jetzt nach korrektur schritt, kommt wieder gruppen bilden, verschmelzen naechster schritt, danach PAP/eher ASM Diagramm.
jetzt im nächsten schritt verschmelzen

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
~~(x_3, x_1, x_0)~~					
~~Gruppe 0~~					
~~$(0, 8)$~~	~~x~~	~~0~~	~~0~~	~~0~~	~~1~~
~~Gruppe 1~~					
~~$(2, 10)$~~	~~x~~	~~0~~	~~1~~	~~0~~	~~1~~
~~Gruppe 2~~					
~~$(6, 14)$~~	~~x~~	~~1~~	~~1~~	~~0~~	~~1~~
$(0, 2, 8, 10)$	x	0	x	0	1
$(2, 6, 10, 14)$	x	x	1	0	1
(x_3, x_1, x_0)					
~~Gruppe 1~~					
~~$(2, 6)$~~	~~0~~	~~x~~	~~1~~	~~0~~	~~1~~
~~Gruppe 2~~					
~~$(10, 15)$~~	~~1~~	~~x~~	~~1~~	~~0~~	~~1~~
$(2, 6, 10, 14)$	x	x	1	0	1
(x_3, x_2, x_0)					
~~Gruppe 0~~					
~~$(0, 2)$~~	0	~~0~~	~~x~~	~~0~~	~~1~~
~~Gruppe 1~~					
~~$(8, 10)$~~	~~1~~	~~0~~	~~x~~	~~0~~	~~1~~
$(0, 2, 8, 10)$	x	0	x	0	1
(x_3, x_2, x_1)					
Gruppe 2					
$(6, 7)$	0	1	1	x	1
$(10, 11)$	1	0	1	x	1

so, bei der 15 ist was schief gelaufen, das müssen wir oben korrigieren und angucken. tatsächlich und das ist ein schnell korrigierbarer fehler. weil: die 15 ist mit einer null versehen, wahrheitswert: falsch und entspricht schaltalgebra: 0.

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
$(0, 2, 8, 10)$	x	0	x	0	1
$(2, 6, 10, 14)$	x	x	1	0	1
$(6, 7)$	0	1	1	x	1
$(10, 11)$	1	0	1	x	1

damit hat die sache funktio-

niert als nächststes kommt: primimplikantentafel.

(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform								
$(i_0, i_1, \ldots, i_n)$	0	2	6	7	8	10	11	14
$(0, 2, 8, 10)$	1	1			1	1		
$(2, 6, 10, 14)$		1	1			1		1
$(6, 7)$			1	1				
$(10, 11)$						1	1	

wie heisst die verknüpfung, bei der wenn ein einziges 1 ist, dann ist die schaltung ok. vielleicht XOR. oder Addieren geht.

$$1XOR0 = 1 \tag{454}$$
$$1XOR0 = 1 \tag{455}$$
$$\tag{456}$$

bessere schreibweise:

$$1 \oplus 0 = 1 \tag{457}$$
$$1 \oplus 0 = 1 \tag{458}$$
$$\tag{459}$$

das problem ist, wenn wir 2 XOR Gatter haben, dann genügt die sache nicht. wir müssen quasi fakultaet machen. weil. wenn 2 mal 1, dann erwischt, dann ist das 0. es gibt alternative. ich mache

$$\{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$$

dann muessen die ergebnisse wieder XOR gemacht werden. das heisst, Zeile für Zeile.

von 4 bis 1 in dem falle runter gehen.

$$((1,2),(1,3),(1,4),(2,3),(2,4),(3,4)) \tag{460}$$
$$= (0,0,0,1) \tag{461}$$
$$\tag{462}$$

jetzt kann man die alle wieder machen und man kann überlegen. ob: das reichen müsste, nämlich wenn das erste gewählte.
eine 0 ist. so und jetzt weiter. stimmt so oder so: primimplikanten. und jetzt weiter: wir machen erst: galasm mit gal16v8.
das ist das erste und dann kommt tikzpicture. und jetzt los. danach kommt das Timing Diagramm, was in LaTeXgibt. und wenn das passiert ist. dann kommt der vergleich mit UML. und wenn das passiert ist, der automat mit allem drum und dran. und dann assembler und die PAP und später wahrscheinlichkeitsrechnung.

```
gal16v8
module gal16v8DNFQuine

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc
```

nach dem ding kommt die gleichung:

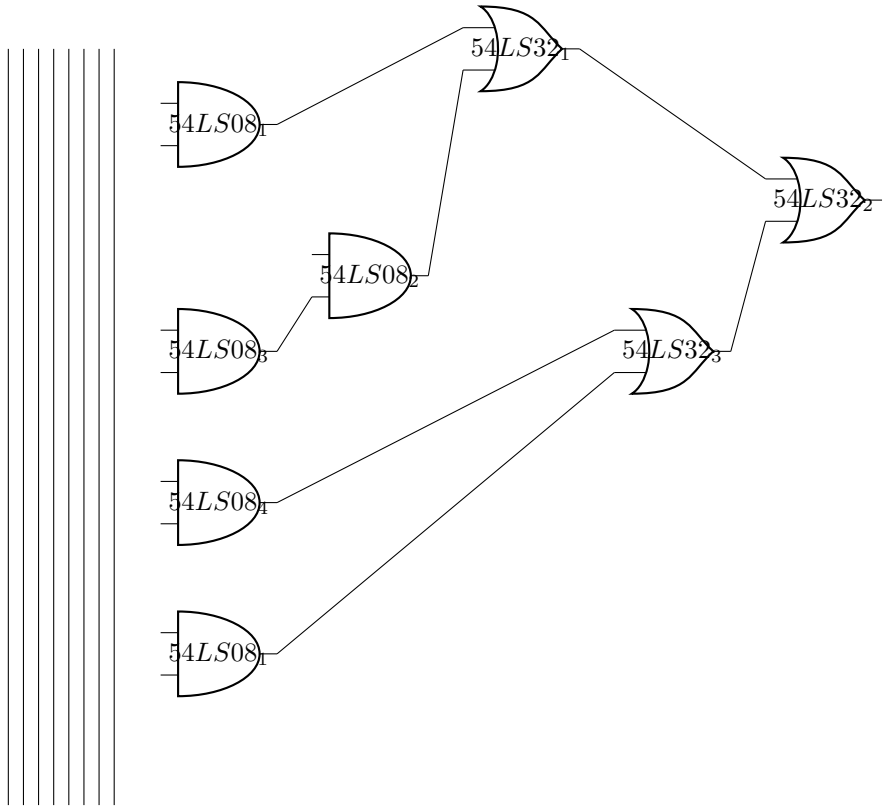
(C) David Vajda Wed Nov 4 14:01:45 2026 4 Network - TTL - Disjunktive Normalform								
$(i_0, i_1, \ldots, i_n)$	0	2	6	7	8	10	11	14
$(0, 2, 8, 10)$	1	1			1	1		
$(2, 6, 10, 14)$		1	1			1		1
$(6, 7)$			1	1				
$(10, 11)$						1	1	

```
gal16v8
module gal16v8DNFQuine

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x2 * /x0 +
  x1 * /x0 +
  /x3 * x2 * x1 +
  x3 * /x2 *x1

description: quine dnf 3 4 network 11/04/2026
```




```
gal16v8
module gal16v8DNFQuine
```

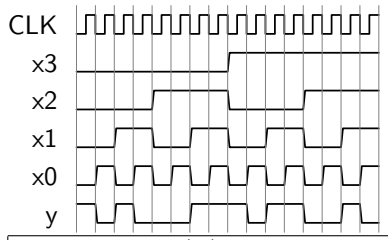
```
clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc  nc nc nc nc vcc

y = /x2 * /x0 +
  x1 * /x0 +
  /x3 * x2 * x1 +
  x3 * /x2 *x1

description: quine dnf 3 4 network 11/04/2026
```

so ist: . . . laut google ai definiert

```
\begin{tikztimingtable}
  % Signalname & Signalverlauf \\
  CLK  & 10{c} \\
  DATA & L 2{H} L 2{H} L \\
  RST  & 3{H} 7{L} \\
  \begin{extracode}
    % Hilfslinien und Gitter einfügen
    \vertlines[help lines]{1,2,3,4,5,6,7,8,9,10}
  \end{extracode}
\end{tikztimingtable}
```



(C) David Vajda
Wed Nov 4 14:01:45 2026
4 Network - TTL - Disjunktive Normalform

i	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	0

perfekt, offensichtlich. und jetzt geht es weiter. jetzt wissen, das ist das timing diagramm. in UML ist das ein:

- Strukturdiagramm: Klassendiagramm, Komponentendiagramm, Kompositionsdiagramm
- Verhaltensdiagramm: Zustandsdiagramm, Use-Case-Diagramm
- Interaktionsdiagramm: Timing-Diagramm, Sequenz-Diagramm, Kommunikationsdiagramm.

Frage was ist ein Sequenzdiagramm und Kommunikationsdiagramm? Sequenz-Diagramm und Kommunikationsdiagramm ist vielleicht, mit atmega8. aber das passt jetzt nicht. jetzt kommt der automat. der hat ein zustandsdiagramm. erstens: und dann geht es weiter.

so weit bin ich gekommen, weiter ging es nicht.

```
## (c) david vajda
## 11/04/2026
## graph mit euler tour algo

V = [1,2,3,4,5,6,7,8,9,10,11,12]
i = 0
E = []
print (len(V))
while i < len (V):
    E.append (V)
    i = i + 1

print (E)

# jetzt zufaellig aussuchen, wie viele
# nachbarn pro knoten existieren sollens

# oder die alternative eine adjazenmatrix
# aber hier vielleicht besser.

import random
random.seed ()

nE = []
npossible = [2,4,6]
i = 0
while i < len (V):
    nE.append (npossible[random.randint(0,2)])
    i = i + 1

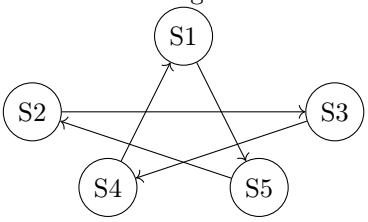
print (nE)

# wir suchen jetzt fuer jeden einen zweiten
# zufaellig aus.
```

```
# und dazu kommt natuerlich anders herum
# das heisst, erst mal ein mal.
E2 = []
i = 0
Etmp = [1,2,3,4,5,6,7,8,9,10,11,12]
m = len (V)
while i < len (V):
    m = m - 1
    n = random.randint (0,m)
    E [i].remove (n)
    print (n)
    print (E[i])
    E2.append (n)
    i = i + 1

# test
print (E2)
```

so weit bin ich gekommen. da es jetzt nicht weiter geht, nehme ich ein pentagram zustandsdiagramm.



mit $sin(x)$ und $cos(x)$ berechnete punkte.

$$(sin(x), cos(x))$$

und dazu. danach mit runden kanten, aber eine alternative aussen rum.

$$\begin{aligned}
P &= (\left(sin\left(\frac{2\cdot\pi\cdot r\cdot 0}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 0}{5}\right)\right), && (463)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 1}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 1}{5}\right)\right), && (464)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 2}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 2}{5}\right)\right), && (465)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 3}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 3}{5}\right)\right), && (466)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 4}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 4}{5}\right)\right)) && (467)\\
& && (468)
\end{aligned}$$

$$
\begin{aligned}
P &= (\left(sin\left(\frac{2\cdot\pi\cdot r\cdot 0}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 0}{5}\right)\right), && (469)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 1}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 1}{5}\right)\right), && (470)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 2}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 2}{5}\right)\right), && (471)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 3}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 3}{5}\right)\right), && (472)\\
&= \left(sin\left(\frac{2\cdot\pi\cdot r\cdot 4}{5}\right), cos\left(\frac{2\cdot\pi\cdot r\cdot 4}{5}\right)\right)) && (473)\\
& && (474)
\end{aligned}
$$

ich stelle fest, dass der durchmesser vielleicht zu gross ist. das ist bei

$$2\cdot\pi = 6,283185307179586$$

dann rechnen wir mit $r = 0.5$

$$
\begin{aligned}
P &= (\left(sin\left(\frac{\pi\cdot 0}{10}\right), cos\left(\frac{\pi\cdot 0}{10}\right)\right), && (475)\\
&= \left(sin\left(\frac{\pi\cdot 1}{10}\right), cos\left(\frac{\pi\cdot 1}{10}\right)\right), && (476)\\
&= \left(sin\left(\frac{\pi\cdot 2}{10}\right), cos\left(\frac{\pi\cdot 2}{10}\right)\right), && (477)\\
&= \left(sin\left(\frac{\pi\cdot 3}{10}\right), cos\left(\frac{\pi\cdot 3}{10}\right)\right), && (478)\\
&= \left(sin\left(\frac{\pi\cdot 4}{10}\right), cos\left(\frac{\pi\cdot 4}{10}\right)\right)) && (479)\\
& && (480)
\end{aligned}
$$

das wiederum ist:

$$
\begin{aligned}
P &= (\left(sin\left(\frac{\pi\cdot 0}{10}\right), cos\left(\frac{\pi\cdot 0}{10}\right)\right), && (481)\\
&= \left(sin\left(\frac{\pi\cdot 1}{10}\right), cos\left(\frac{\pi\cdot 1}{10}\right)\right), && (482)\\
&= \left(sin\left(\frac{\pi\cdot 2}{10}\right), cos\left(\frac{\pi\cdot 2}{10}\right)\right), && (483)\\
&= \left(sin\left(\frac{\pi\cdot 3}{10}\right), cos\left(\frac{\pi\cdot 3}{10}\right)\right), && (484)\\
&= \left(sin\left(\frac{\pi\cdot 4}{10}\right), cos\left(\frac{\pi\cdot 4}{10}\right)\right)) && (485)\\
& && (486)
\end{aligned}
$$

$$\frac{\pi}{10} = 0,314159265358979$$

$$\begin{aligned} X &= (0.314159265358979, && (487)\\ &, 0.628318530717959, && (488)\\ &, 0.942477796076938, && (489)\\ &, 1.256637061435917 && (490)\\ &, 1.570796326794897) && (491)\\ & && (492) \end{aligned}$$

$$\begin{aligned} P &= (((sin(0.314159265358979), cos(0.314159265358979)) && (493)\\ &, ((sin(0.628318530717959), cos(0.628318530717959)), && (494)\\ &, ((sin(0.942477796076938), cos(0.942477796076938)), && (495)\\ &, ((sin(1.256637061435917), cos(1.256637061435917)), && (496)\\ &, ((sin(1.570796326794897), cos(1.570796326794897))) && (497)\\ & && (498) \end{aligned}$$

so, hier sind die werte

$$\begin{aligned} P &= (((sin(0.314159265358979), cos(0.314159265358979)) && (499)\\ &, ((sin(0.628318530717959), cos(0.628318530717959)), && (500)\\ &, ((sin(0.942477796076938), cos(0.942477796076938)), && (501)\\ &, ((sin(1.256637061435917), cos(1.256637061435917)), && (502)\\ &, ((sin(1.570796326794897), cos(1.570796326794897))) && (503)\\ & && (504) \end{aligned}$$

24 Klasse Algorithmen

24.1 Class All - economics - algorithms - operating systems + mathematiker cions

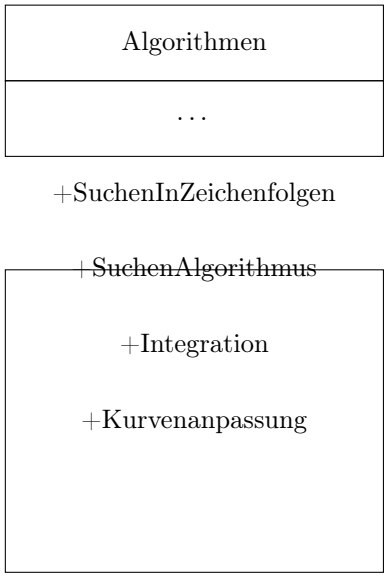
$$\begin{aligned} Random &= ((&& (505)\\ &, linearеKongruenz() && (506)\\ &, additiveKongruenz() && (507)\\ &, TestChiQuadrat()) && (508)\\ &, (N) && (509)\\ &, r && (510)\\ &, m && (511)\\ &, b)) && (512)\\ & && (513) \end{aligned}$$

$$
\begin{aligned}
SuchenInZeichenFolgen &= ((&& (514)\\
&\quad , && KnuthMorrisPratt(a,p), && (515)\\
&\quad , && BoyerMoore(a,p), && (516)\\
&\quad , && BoyerMoore(a,p)), && (517)\\
&\quad , && a[], && (518)\\
&\quad , && p[] && (519)\\
&\quad , && n, && (520)\\
&\quad , && m)) && (521)\\
& && && (522)
\end{aligned}
$$

$$
\begin{aligned}
SuchenAlgorithmus &= ((&& (523)\\
&\quad , && InsertionsSort(), && (524)\\
&\quad , && SelectionSort(), && (525)\\
&\quad , && BubbleSort(), && (526)\\
&\quad , && ShellSort(), && (527)\\
&\quad , && QuickSort(), && (528)\\
&\quad , && MergeSort(), && (529)\\
&\quad , && HeapSort())) && (530)\\
& && && (531)
\end{aligned}
$$

$$
\begin{aligned}
Integration &= ((&& (532)\\
&\quad , && Symbolisch(), && (533)\\
&\quad , && EinfachesQuadraturverfahren(), && (534)\\
&\quad , && AdaptiveQuadratur(), && (535)\\
&\quad , && ZusammengestzteVerfahren()), && (536)\\
&\quad , && (FunktionsVorschrift())) && (537)\\
& && && (538)
\end{aligned}
$$

$$
\begin{aligned}
Kurvenanpassung &= ((&& (539)\\
&\quad , && InterpolationLagrange(), && (540)\\
&\quad , && Splines(), && (541)\\
&\quad , && KleinsteQuadrate())) && (542)\\
& && && (543)
\end{aligned}
$$



besser kriege ich es nicht hin:

```
\begin{tikzpicture}
\node [rectangle,draw=black,minimum width=5cm, minimum height=1cm] at (2.5,4.5) {Algorithm
\node [rectangle,draw=black,minimum width=5cm, minimum height=1cm] at (2.5,3.5) {\dots};
\node [rectangle,draw=black,minimum width=5cm, minimum height=4cm] at (2.5,-0.5) {};
\node [rectangle,minimum width=5cm, minimum height=1cm] at (2.5,2.5) {+SuchenInZeichenfolg
\node [rectangle,minimum width=5cm, minimum height=1cm] at (2.5,1.5) {+SuchenAlgorithmus};
\node [rectangle,minimum width=5cm, minimum height=1cm] at (2.5,0.5) {+Integration};
\node [rectangle,minimum width=5cm, minimum height=1cm] at (2.5,-0.5) {+Kurvenanpassung};
\end{tikzpicture}
```

es gibt vielleicht UML mit LaTeX?! und es gibt: SysML und SOA. mal gucken, was da passiert. nächster punkt. das Pentagram von Gestern kommt. und jetzt die punkte richtig. und dann als automat. und dann kommt darauf tiefensuche, breitensuche, eulertour. und gestern muss ein graph mit eulertouren generiert werden.
dann PAP.

25 graph und eulertour, tiefensuche, breitensuche + automat + schaltwerk

$$
\begin{aligned}
P &= (((sin(0.314159265358979), cos(0.314159265358979)) && (544)\\
&, ((sin(0.628318530717959), cos(0.628318530717959)), && (545)\\
&, ((sin(0.942477796076938), cos(0.942477796076938)), && (546)\\
&, ((sin(1.256637061435917), cos(1.256637061435917)), && (547)\\
&, ((sin(1.570796326794897), cos(1.570796326794897))) && (548)\\
& && (549)
\end{aligned}
$$

$$
\begin{aligned}
P &= ((0,1) && (550)\\
&, (0.02193069589621, 0.999759493367034), && (551)\\
&, (0.043850842836762, && (552)\\
&, ((sin(0.628318530717959), cos(0.628318530717959)), && (553)\\
&, ((sin(0.942477796076938), cos(0.942477796076938)), && (554)\\
&, ((sin(1.256637061435917), cos(1.256637061435917)), && (555)\\
&, ((sin(1.570796326794897), cos(1.570796326794897))) && (556)\\
& && (557)
\end{aligned}
$$

```
\begin{tikzpicture}
\node [circle,draw=black] (S0) at (0.0,[0.0015926529164868282, -0.9999987317275395]) {S0
\node [circle,draw=black] (S1) at (0.0015926529164868282,[0.0015926529164868282, -0.9999
\node [circle,draw=black] (S2) at (-0.0031853017931379904,[0.0015926529164868282, -0.999
\node [circle,draw=black] (S3) at (0.0047779425901285115,[0.0015926529164868282, -0.9999
\node [circle,draw=black] (S4) at (-0.006370571267652135,[0.0015926529164868282, -0.9999
\end{tikzpicture}
```

das ergebnis ist nicht optimal.

```
## (c) david vajda
## 11/05/2026
## python3 - pentagramm - graph
## ... koordinaten - python

import math

r = 0.5
pi = 3.14
sector = pi*2*r

PUNKTE = []
i = 0
while i < 5:
    PUNKTE.append ([math.sin(sector*i),math.cos(sector*i)])
    i = i + 1

print (PUNKTE)

# LaTeX Version

# PUNKTE = []
# i = 0
# while i < 5:
#    PUNKT = [sin(sector*i),cos(sector*i)]
#    PUNKTE.append (PUNKT)
#    print ()
#    i = i + 1
```

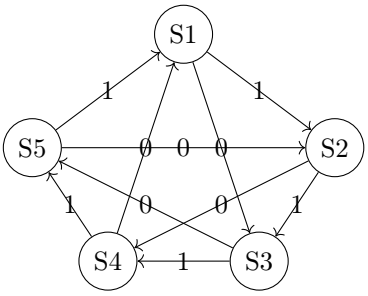
```
# diese version war schlecht, einfach noch mal ausgeben

i = 0
while i < 5 :
    print (f"\\node [circle,draw=black] (S{i}) (at) ({PUNKTE[i][0]},{PUNKTE[1]}) S{i};")
    i = i + 1
```

so jetzt geht das hier weiter.

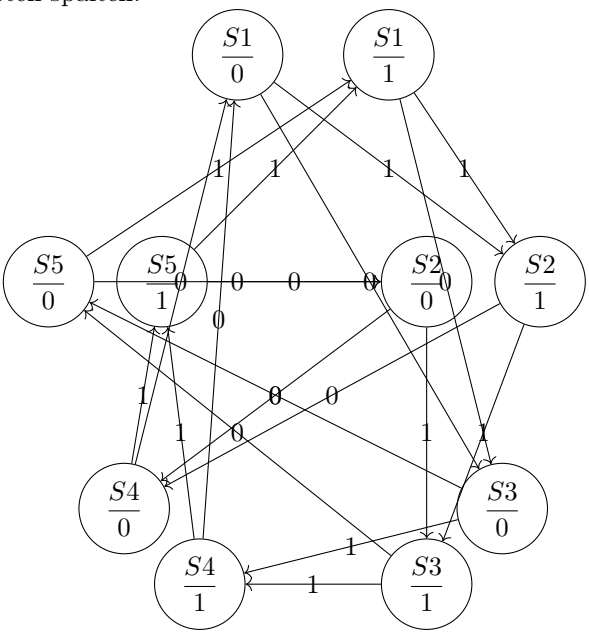
die punkte berechnen hat hier nicht funktioniert.

also, machen wir das pentagram von vorher.

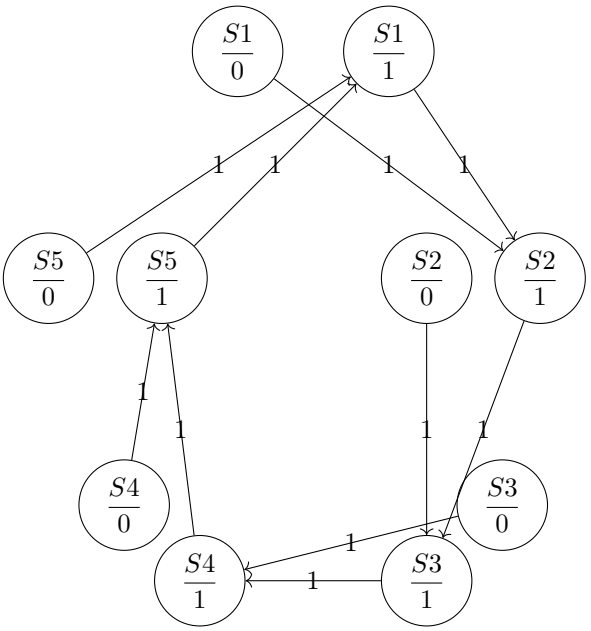


jetzt haben wir hier den graphen. so besser kriege ich das mit der 0 nicht hin. vielleicht zeigt sich ja hier was mathematisches.

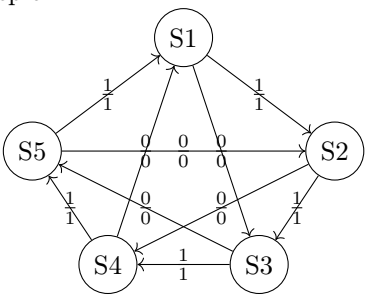
daraus machen wir einen automat. das ist bisher moore. wir müssen alle knoten spalten.



inzwischen kam eine interessante version:



hier steht die version, wenn die ausgabe 1 ist, wenn die eingabe 1 ist, als ein beispiel



so, jetzt machen wir das, wir haben das transformiert. und jetzt kommt, die zustandstabelle. dann unär kodiert. und wenn das geschehen ist, dann kommt: die umsetzung mit dem schaltnetz und schaltwerk, übergang und ausgang, getrennt.

Z	x	Z^+	y
$S1$	0	$S3$	0
$S1$	1	$S2$	1
$S2$	0	$S4$	0
$S2$	1	$S3$	1
$S3$	0	$S5$	0
$S3$	1	$S4$	1
$S4$	0	$S1$	0
$S4$	1	$S5$	1
$S5$	0	$S2$	0
$S5$	1	$S1$	1

so, da können wir ein vernünftiges diagramm draus machen.

```
s5 = s4 * x + s3 * /x
s4 = s2 * /x + s3 * x
s3 = s1 * /x + s2 * x
```

```
s2 = s2 * x + s5 * /x
s1 = s4 * /x + s5 * x
```

$$
\begin{aligned}
S_5 &= S_4 \wedge x \vee S_3 \wedge \neg x && (558)\\
S_4 &= S_2 \wedge \neg x \vee S_3 \wedge x && (559)\\
S_3 &= S_1 \wedge \neg x \vee S_2 \wedge x && (560)\\
S_2 &= S_2 \wedge x \vee S_5 \wedge \neg x && (561)\\
S_1 &= S_4 \wedge \neg x \vee S_5 \wedge x && (562)\\
y &= x && (563)\\
& && (564)
\end{aligned}
$$

Formel für Flip Flops

```
q = r nor q'
q' = s nor q
```

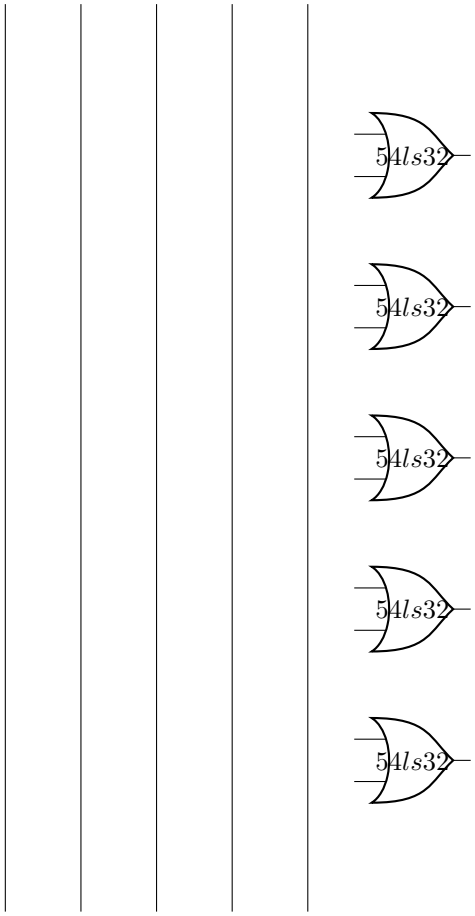
vielleicht kann man jetzt eine formel für das FF formulieren.

$$
\begin{aligned}
S_5 &= S_4 \wedge x \vee S_3 \wedge \neg x && (565)\\
S_4 &= S_2 \wedge \neg x \vee S_3 \wedge x && (566)\\
S_3 &= S_1 \wedge \neg x \vee S_2 \wedge x && (567)\\
S_2 &= S_2 \wedge x \vee S_5 \wedge \neg x && (568)\\
S_1 &= S_4 \wedge \neg x \vee S_5 \wedge x && (569)\\
q &= \neg(q' \vee r) && (570)\\
q' &= \neg(s \vee q) && (571)\\
f_i(r_i, s_i) &= ((\neg(q_i' \vee r_i), (s_i \vee q_i)) && (572)\\
y &= x && (573)\\
& && (574)
\end{aligned}
$$

irgendwie so, mit übergangsschaltnetz.

jetzt ist das, das übergangsschaltnetz

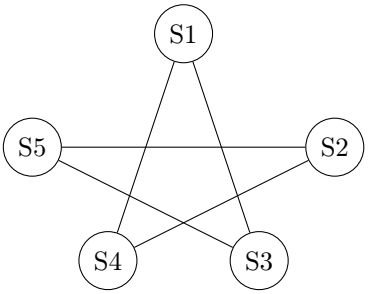
Z	$un(Z)$	x	Z^+	$un(Z^+)$	y
$S1$	$0b00001$	0	$S3$	$0b00100$	0
$S1$	$0b00001$	1	$S2$	$0b00010$	1
$S2$	$0b00010$	0	$S4$	$0b01000$	0
$S2$	$0b00010$	1	$S3$	$0b00100$	1
$S3$	$0b00100$	0	$S5$	$0b10000$	0
$S3$	$0b00100$	1	$S4$	$0b01000$	1
$S4$	$0b01000$	0	$S1$	$0b00001$	0
$S4$	$0b01000$	1	$S5$	$0b10000$	1
$S5$	$0b10000$	0	$S2$	$0b00010$	0
$S5$	$0b10000$	1	$S1$	$0b00001$	1



so oder so ähnlich kann man das machen. so. ohne die striche ein zu zeichnen.

jetzt kann man hier vielleicht eine tiefen und breitensuche machen, war die aufgabe.

Tiefensuche. Breitensuche. zunächst der ungerichtete Graph eines Pentagrams.



$$E = ((S1, S3), (S3, S5), (S5, S2), (S2, S4), (S4, S1))$$

umbennen in Richtige Kanten

$$
\begin{aligned}
V &= \{1,2,3,4,5\} && (575)\\
E &= \{\{1,3\},\{3,5\},\{5,2\},\{2,4\},\{4,1\}\} && (576)\\
G &= \{V,E\} && (577)\\
& && (578)
\end{aligned}
$$

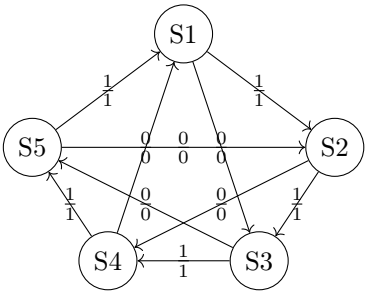
so, ich wollte eine euler tour machen, mit den kanten und so weiter.
aber, irgendwas tut hier bei dem mathe ding nicht.
ich mache jetzt erst eine tiefen und breitensuche. ich sage: gut, dass, wir die 1er für den kreis aussen rum genommen haben und die 0en für das pentagramm. der algorithmus in python3. dass man einen graphen mit 12 knoten macht, der immer eulertouren macht, zum üben kommt nachher. aber: jetzt eine tiefen und breitensuche aus dem kopf, geht so:

- Breitensuche: wir gehen an einen knoten und wir tun die nachbarn auf eine schlange. diese nehmen wir beim kopf wieder ab, die elemente und gehen dann damit weiter. und legen hinten drauf. ... die nachbarn und machen damit weiter,
- tiefensuche geht so: dass wir zu den nachbarn gehen. und dann nehmen wir den ersten und gehen rekursiv weiter.
  an dieser stelle gut, mit 1er aussen, 0er innen oder anders herum. damit: wir entscheiden, 1er entscheidung, 0er entscheidung, abwechselnd, oder zufaellig, oder nach irgendeinem muster. jetzt anfangen.

jetzt habe ich die sache noch mal hinbekommen, wir machen eine eulertour, das war schon eine.

$$
\begin{aligned}
V &= \{1,2,3,4,5\} && (579)\\
E &= ((1,3),(3,5),(5,2),(2,4),(4,1)) && (580)\\
G &= \{V,E\} && (581)\\
& && (582)
\end{aligned}
$$

aber mit dem: tiefen und breitensuche. aber wie wir es geometrisch sehen.



sagen wir das ist ein gerichter graph, danach nehmen wir ungerichtete, jetzt kommt die breitensuche.

```
S1 und dann tun wir uns merken, nachbarn S2, S3
```

```
1.) Schlange: S1
Komponente: S1
2.) Schlange: S2, S3
Komponente: S1, S2
Nachbarn (S2) = S3, S4,
 S3 ist schon
Schlange: S2, S3, S4
3.) Schlange: S3, S4
Komponente: S1, S2, S3
Nachbarn (S3) = S4, S5
Schlange: S4, S5
Komponente: S1, S2, S3, S4
4. Schlange: S4, S5
Nachbarn (S4): S5, S1
Schlange: S1, S2, S3, S4, S5
...

das waere die eine moeglichkeit.
```

jetzt tiefensuche:

```
wir nehmen die 1er
S1, Nachbarn (S1) = {S2, S3}
Komponente: S1, S2
Rekursiv, S2, Nachbarn (S2) = {S3, S4}
S3
da muessten einen baum des besuchens malen.

S1
 |-S2
 |-S3

wir gehen den S2 weg

S1
  |-S2
     |-S3
     |-S4
  |-S3

 naechster schritt

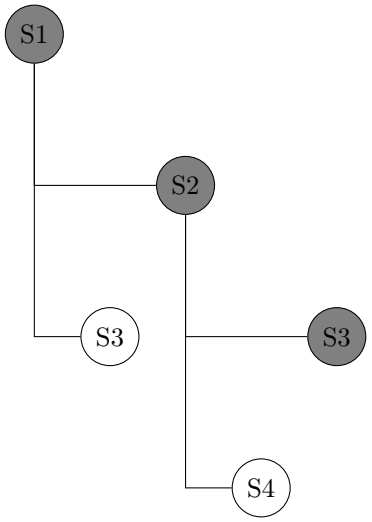
 S1
  |-S2
     |-S3
         |-S4
         |-S5
     |-S4
  |-S3

  S1
```

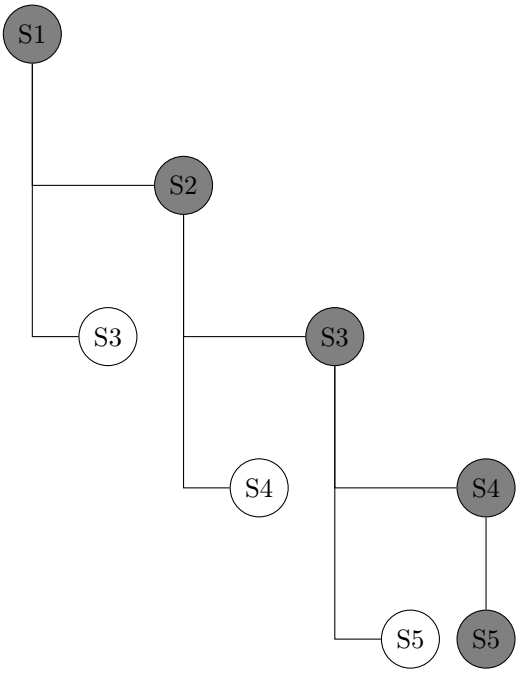
```
  |-S2
     |-S3
         |-S4
            |-S1
            |-S2
         |-S5
     |-S4
  |-S3
```

```
ok, rekursion nach einer. und jetzt nach.
sagen wir 0er

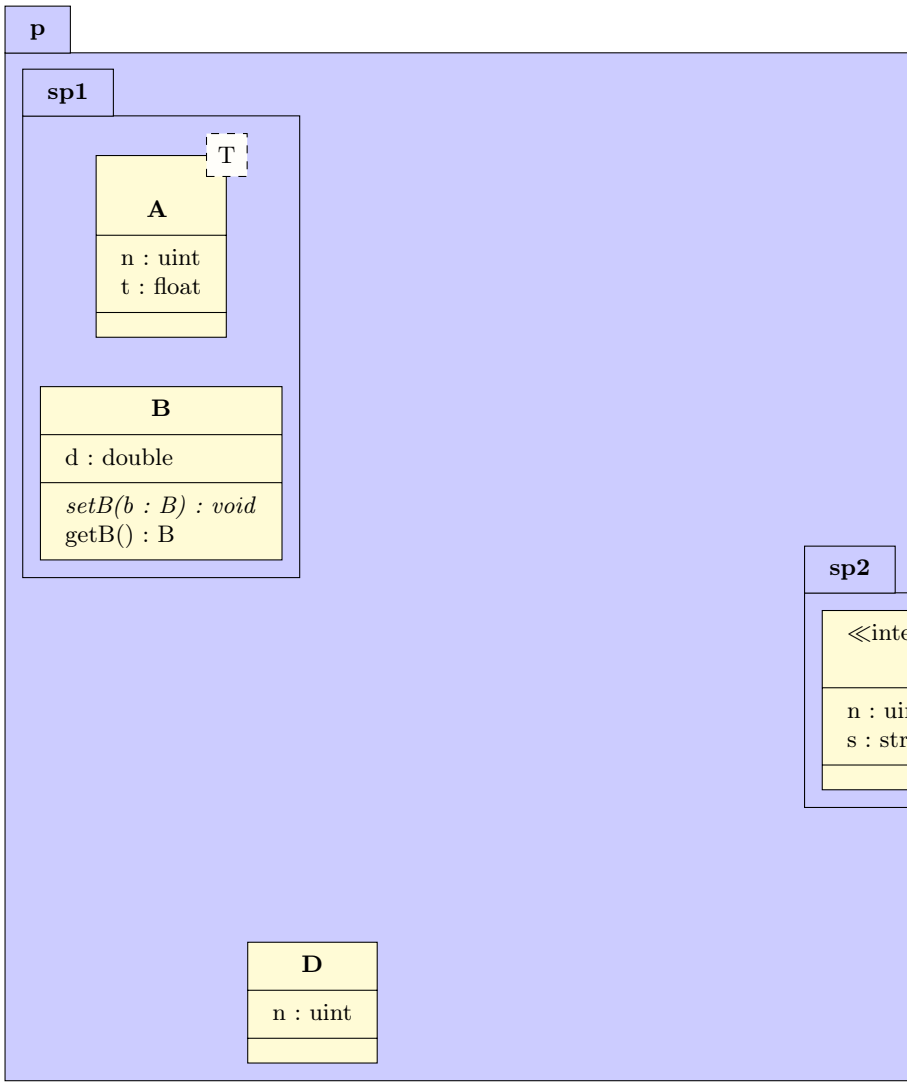
machen wir graphen baum. und das ist komponente, bis wieder am anfang.
S1
dazu kommt noch tabelle bei breitensuche.
```



jetzt hier kommt mit mehr knoten komponente


S1
S2
S3
S3
S4
S4
S5
S5

jetzt probieren wir das aus:



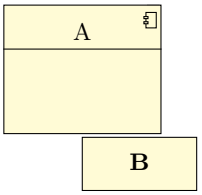

```
\begin{umlpackage}{p}
\begin{umlpackage}{sp1}
\umlclass[template=T]{A}{
  n : uint \\ t : float
  }{}
\umlclass[y=-3]{B}{
  d : double
  }{
  \umlvirt{setB(b : B) : void} \\ getB() : B
}
\end{umlpackage}
\begin{umlpackage}[x=10, y=-6]{sp2}
\umlinterface{C}{
  n : uint \\ s : string
  }{}
```

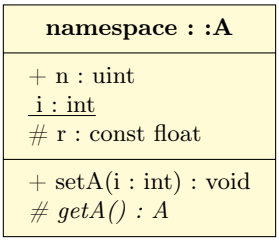
```
\end{umlpackage}
\umlclass[x=2, y=-10]{D}{
  n : uint
  }{}
\end{umlpackage}
```

gut, das ding tut, aber mit

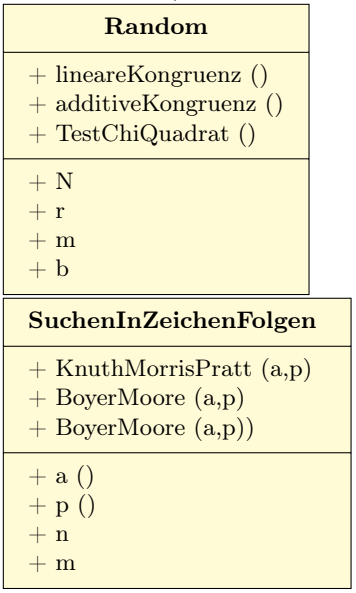
```
\begin{tikzpicture}
```

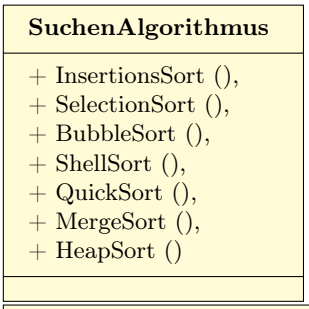
drum rum. ok, ausprobieren.

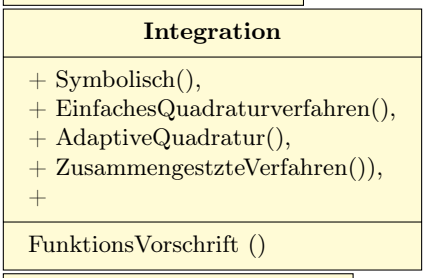


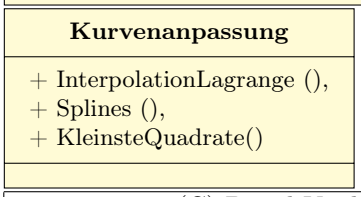


so, jetzt haben wir ja die klasse von gestern: und da kommt jetzt das rein, was wir da hatten, mal mit einer richtigen klasse.









(C) David Vajda
Sat Nov 7 05:04:30 2026
4 Network - TTL - Disjunktive Normalform

i	x_3	x_2	x_1	x_0	y
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	0

(C) David Vajda Sat Nov 7 05:04:30 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
6	0	1	1	0	1
11	1	0	1	1	1
13	1	1	0	1	1

so, naechstes ding:

(C) David Vajda Sat Nov 7 05:04:30 2026 4 Network - TTL - Disjunktive Normalform					
i	x_3	x_2	x_1	x_0	y
Gruppe 2					
6	0	1	1	0	1
Gruppe 3					
11	1	0	1	1	1
13	1	1	0	1	1

$$y = \neg x_3 \wedge x_2 \wedge x_1 \wedge \neg x_0 \vee \quad (583)$$
$$x_3 \wedge \neg x_2 \wedge x_1 \wedge x_0 \vee \quad (584)$$
$$x_3 \wedge x_2 \wedge \neg x_1 \wedge x_0 \vee \quad (585)$$

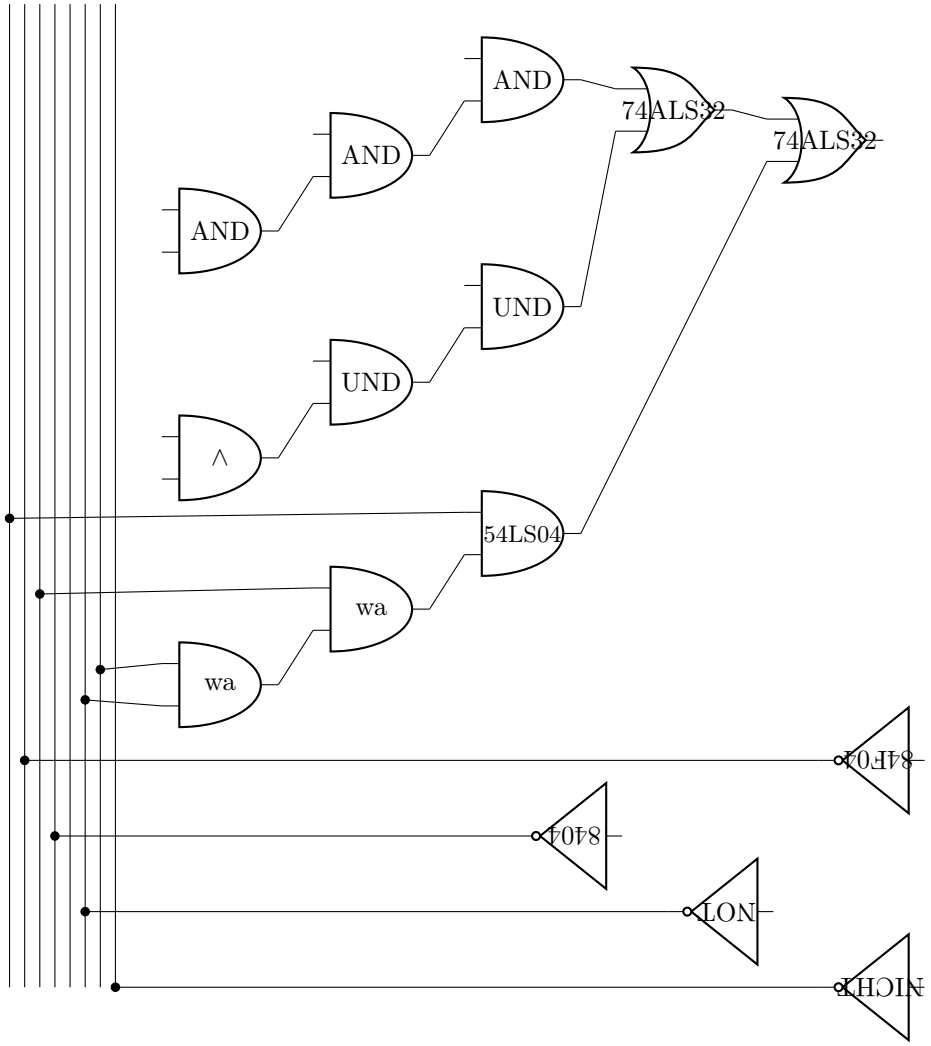
so, ok und nun, geht es weiter.

```
gal16v8
module gal16v8quinednf

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc   nc nc nc nc vcc

y =  /x3 * x2 * x1 */x0 +
 x3 * /x2 * x1 * x0 +
 x3 * x2 * /x1 * x0

description: gal16v8 quine mc cluskey galasm
```




```
gal16v8
module gal16v8quinednf

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc   nc nc nc nc vcc

y =  /x3 * x2 * x1 */x0 +
 x3 * /x2 * x1 * x0 +
 x3 * x2 * /x1 * x0

description: gal16v8 quine mc cluskey galasm
```

ok, ab jetzt nehme ich figure, zum durchnummerieren der bilder.
also:

jetzt kommt der PAP. und vorher erwartungswert - und: ich wuerfle 24 mal.
mache erwartungswert für

```
gal16v8
module gal16v8quinednf

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc    nc nc nc nc vcc

y =      /x3 * x2 * x1 */x0 +
         x3 * /x2 * x1 * x0 +
         x3 * x2 * /x1 * x0

description: gal16v8 quine mc cluskey galasm
```

Abbildung 1: ein galasm quine mc cluskey DNF Code gal16v8

$$P(X_i)$$

ist die Wahrscheinlichkeit. Nationation?

$$X_i = i^2 = \{1, 4, 9, 16, 25, 36\}$$

und jetzt wahrscheinlichkeit:

$$P(X = x_i)_{i=1\ldots6} = \left\{\frac{1}{6}, \ldots, \frac{1}{6}\right\}$$

$$E(X) = \sum_{i=1}^{6} (P(X = x_i)) \cdot \frac{1}{6}$$

$$E(X) = (1 + 4 + 9 + 16 + 25 + 36) \cdot \frac{1}{6} = (30 + 61) \cdot \frac{1}{6} = \frac{91}{6} = 15 + \frac{1}{6}$$

jetzt die varianz und standardabweichung. varianz geht so:

$$s^2 = (x_1 - E(X))^2 \cdot P(X = x_1) + \cdots + (x_n - E(X))^2 \cdot P(X = x_n)$$

$$s^2 = (x_1 - E(X))^2 \cdot P(X = x_1) + \cdots + (x_6 - E(X))^2 \cdot P(X = x_6)$$

$$s^2 = (1 - E(X))^2 \cdot \frac{1}{6} + \cdots + (36 - E(X))^2 \cdot \frac{1}{6}$$

$$\begin{aligned}
s^2 &= ((X = x_1) - E(X))^2 \cdot P(X = x_2) && (586)\\
&+ ((X = x_2) - E(X))^2 \cdot P(X = x_2) && (587)\\
&+ ((X = x_3) - E(X))^2 \cdot P(X = x_3) && (588)\\
&+ ((X = x_4) - E(X))^2 \cdot P(X = x_4) && (589)\\
&+ ((X = x_5) - E(X))^2 \cdot P(X = x_5) && (590)\\
&+ ((X = x_6) - E(X))^2 \cdot P(X = x_6) && (591)\\
& && (592)
\end{aligned}$$

ok.

$$
\begin{aligned}
s^2 &= (1-15.1666)^2 \cdot 0.16666 && (593)\\
&+ (4-15.1666)^2 \cdot 0.16666 && (594)\\
&+ (9-15.1666)^2 \cdot 0.16666 && (595)\\
&+ (16-15.1666)^2 \cdot 0.16666 && (596)\\
&+ (25-15.1666)^2 \cdot 0.16666 && (597)\\
&+ (36-15.1666)^2 \cdot 0.16666 && (598)\\
& && (599)
\end{aligned}
$$

das sind:

$$
\begin{aligned}
s &= E(x) && (600)\\
Var(s) &= s^2 && (601)\\
s^2 &= (1-15.1666)^2 \cdot 0.16666 && (602)\\
&+ (4-15.1666)^2 \cdot 0.16666 && (603)\\
&+ (9-15.1666)^2 \cdot 0.16666 && (604)\\
&+ (16-15.1666)^2 \cdot 0.16666 && (605)\\
&+ (25-15.1666)^2 \cdot 0.16666 && (606)\\
&+ (36-15.1666)^2 \cdot 0.16666 && (607)\\
&= 12,54340159722225 && (608)\\
&+ 7,793393472225 && (609)\\
&+ 2,376730972225 && (610)\\
&+ 6,043410972225 && (611)\\
&+ 4,88901848765625 && (612)\\
&+ 27,126753472225 && (613)\\
&= 60,7727089737785 && (614)\\
& && (615)
\end{aligned}
$$

Standardabweichung von

$$E(X)) = \sqrt{}(Var(E(X)) = 7,795685279292546$$

gut nehmen wir würfel prognosen an, bei denen wir z.B. folgendes tun, wir rechnen nun noch den erwartungswert von auge, entspricht rendite aus. indem wir zum beispiel markowitz probieren wir danach. demnach gibt es zwei reihen. wir könnten wie bisher vollkommen unregelmässig. oder das ergebnis anpassen. erstes: blauer würfel ist, aktie A, gelber Würfel Aktie B. allerdings. wir legen um die zufallszahlen eine kurve, die nach innen oder aussen geht. ähnlich einer gedämpften schwingung.

haben wir das: dann machen wir folgendes:

26 nächster Tag

folgendes: bisher hatte ich auf Mac OS X wenig kontakt zu compilern, bis auf weiterhin auf linux. doch der schnelle griff zu compilern gelang nicht. meine Mac OS X version ist Mac OS X 11.7.11 und erlaubt das installieren vieler apps über AppStore nicht. etwa XCode. mit GUI.

wohl aber konnte ich

`nasm`

installieren, auf die klassische art und weise von der homepage

`\href{https://www.nasm.us}{https://www.nasm.us}`

als auch.

`gcc`

mit

`xcode-select`

und statt

`python3`

läuf bei mir version 2.7. ok. ich hatte ein buch. C++, UML und Design Patterns, das werde ich mir wieder kaufen. ebenso: niklas wirth. oder dejkstra. aber: dann kommt dazu: ulmen, Compilerbau I, ich hatte auch 2. oder das der fernuniversität in Hagen. hier habe ich auch noch Java. Objektorientierte. Wie wäre es mal wieder mit Pascal. Ebenso will ich lernen Basic. Zurück zur Sache.
für bisheriges genügt das auf dem Mac:

- gcc
- g++
- LaTeXzur Visualisierung, vor allem Hardware - mit: tikzpicture - und circuitikz - UML, ieee-1984/91, PAP, ...
- nasm
- python3 o. 2.7

jedenfalls:

`nasm`

habe ich hier gefunden:

`https://forums.macrumors.com/threads/x86_64-assembly-w-nasm-on-intel-based-mac.2410309/`

das bedeutet. genauso wie bei linux/minix/UNIX. gehen die SystemCalls ungefähr so:

1. 1: EXIT

2. 2: FORK
3. 3: READ
4. 4: WRITE
5. 5: OPEN
6. 6: CLOSE
7. ...

sie werden in

```
rax
```

rein getan. man kann jetzt den befehl

```
systemcall
```

machen, was wohl ein macro für unter UNIX/Linux üblich, Linux zwingend

```
int 0x80
```

trotzdem: der unterschied ist noch die übersetzung, das muss man sich merken. mit dem format

```
macho64
```

den befehl zum übersetzen auswendig lernen. so, der unterschied zu linux, keine register

```
rbx
rcx
```

sondern

```
rsi
rdi
```

ok. dann kann man damit los legen, wie immer dazu pap. und es geht natürlich dazu auch

```
MMX, SSE, SSE2,
xxx-512 (RAX??)
```

oder wie letzteres auch immer heist. das hier ist ein

```
Intel Core i7
```

und damit geht das gesamt MMX ich über

```
SSE4
```

oder wie nicht hinaus, das moderne geht mich noch nichts an. dazu kommen, dann die register

```
mm0 .. mm7 (MMX)
xmm0 .. xmm7 (SSE)
```

und das geht genauso mit den befehlen:

```
paddb
paddw
paddd

movq

pcmpeqb
pcmpeqw
pcmpeqd

P - wie immer Packed
CMP
EQ - equal
B, W, D: Byte Word Double Word

pcmpgtb
pcmpgtw

Packed CMP GT - Greater
Byte Word Doubleword

PUNPCK
unpack
PACK
usw.

und dann kommt

paddsb
paddsw
usw.

paddusb
paddusw
```

und so weiter, da bin ich ja am üben. gut das kann ich direkt auf dem mac ausführen. so, dann kommen aber die algorithmen besonders in C. dazu gehören eben

- Suchen in Zeichenfolgen
  - Knuth-Morris-Pratt
  - Boyer-Moore
  - Rabin-Karp
- das parsing
  - erst mal für BNF - Bakus Naur Form (Das ist richtig) - allerdings fehlt mir die beseitigung der linksrekursion. ich hatte bücher kaufe ich neu. das kann man allerdings überall finden und auswendig lernen

  - dann anwenden, jeden Tag in C ein Compiler für die beiseitigung der Links Rekursion in BNF - aber arithmetische ausdrücke
  - und BNF für automaten - pattern matching
  - ohne beseitigung - quasi
- dann geht das weiter, dann kommt:
  - ein automat für das pattern matching, mit oder ohne compiler, aber als C-Code schreiben
  - daran fehlt es mir
- dann, die klassischen sortier und suchalgorithmen
  - Selection Sort
  - Insertion Sort
  - Shell Sort
  - Bubble Sort
  - Quick Sort

der unterschied: ich finde jeden tag sollten diese programme geschrieben werden. woran es mir auch fehlt, sind C-Funktionen

```
gets
getch
getc
printf
```

man beherrscht das auch irgendwie und doch wie in PHP am ende fehlt es an den 100 pro C funktionen selber, meine C Bücher sind nicht da. ich bevorzuge original. natürlich. C++, UML Design Patterns ist perfekt kaufe ich neu. Ulmen Seti auch. C muss ein mal ein solches Original Addison Wesley hinweise kommen aus dem Minix Leitfaden. Aber dann muss man jetzt los legen. die aufgabe lautet: eine liste der codes, die geschrieben werden müssen

- nasm: klassisches programm für die ausgabe von zeichenketten in Assembler (Intel Core i7) aber Mac OS X
- nasm: MMX und SSE: Addieren usw.
- C: Knuth Morris Pratt
- C: Boyer Moore
- C: Rabin Karp
- C: Insertion Sort
- C: Selection Sort
- C: Bubble Sort
- C: Shell Sort
- C: Vielleicht Quick Sort

- C: Compiler - BNF - Arithmetische Ausdrücke mit Beseitigung der Liniksrekursion
- C: und ohne
- C: BNF mit Automat für Pattern Matching
- C: Pattern Matching Automat
- Python3, irgendwelche programme, irgendwie ich sitze nicht so in python3
- besonders dann auch wert legen auf mathematische algorithmen, besonders in C: also: sagen wir: lagrange, aber in C. wie allerdings in Robert Sedgewick

der unterschied ist: erstens mal: es ist schön den algorithmus so vor augen zu haben, den code nicht zu schreiben. hardware entwickelt mit Gattern und Mit LaTeX. besonders. wegen Quine und so weiter. dafür brauche ich: IEEE-1984/91 und so weiter. programme sind da. blos:
es ist auch schön hash tabellen zu haben und trotzdem die implementation in C ist nicht jeden Tag wie bei Zufallszahlen und so wie jeden Tag die eulertour dazu gehört nur kleine programme. so sollte auch mit den algorithmen, aber korrekt in C. ohne lange Worte, wir fangen an. und eine ergänzung, in LaTeX, in zukunft bitte figure benutzen, und an den bildern eine nummerierung. jetzt anfangen! besonders, als erstes die beseitigung der links rekursion, nämlich im internet gucken und die grammatik auswendig lernen, die formale beschreibung, der formalen grammatik. und: anwenden. das findet man. so. und eben KMP, gleich als erstes. das sind kleines codes, das geht ebenso langsam los, wie alles. jeden tag den Code. das führt schnell zu

```
lex
yacc
```

blos: das problem nicht jeden tag eine BNF arithmetisch und die beseitung sitzt nicht und kein

```
lex yacc
```

dazu muss man den algorithmus vor augen haben, ihn auch KMP nicht schreiben, lohnt sich nicht, aber das dann einfach jeden tag, ist insofern wenig code. jetzt gleich anfangen, ein wenig kaffee und ein paar zigaretten.

26.1 Knuth Morris Pratt

so, der algorithmus geht so, wie jeder sage wir: Suchen in Zeichenfolgen.

geht der: nur mit einem vorlauf. sie suchen in der zeichenfolge selber.

also wir suchen im text nach dem muster, wie bisher, und vorher suchen in der zeichfolge nach der zeichenfolge.

jetzt das probieren wir mal aus:

dabei müssen wir unter anderem, aber auch ein paar vergleiche kennen, nämlich

```
while ((p [j] != p [i]) && (j > 0))
```

und das aus dem Text abgeguckt für den Text

```
while ((p [j] != a [i]) && (j > 0))
```

so, jetzt müssen wir: auf feinheiten achten. das erste, wir müssen den normalen algorithmus schreiben. für.
den algorithmus suchen von zeichenfolgen im text. das ist das eine. klassischerweise, hier muss man trotzdem auf unterschiede achten z.B. beispiel heisst das nicht.

```
while ((p [j] != a [i]) && (j > 0))
```

sondern:

```
while ((p [j] != a [i]) && (j >= 0))
```

also groessen gleich null fuer die variable julius. so. wenn das ist. so, dann erst mal probieren, das klassische zu üben. und wir müssen auch noch die grammatik mit der beseitigung der linksrekursion ausführen. jetzt hier, das suchen von zeichenfolge in einem text, nämlich ein muster im text

```
for (i = 0, j = 0;  i < N; i++, j++) {

}
```

der unterschied beim programmieren ist, das muss man eigentlich ausprobieren, das heisst, das jetzt ohne Buch. mit und ohne buch. das bedeutet aber: folgendes: ich muss den C Compiler nehmen, was beim suchen von zeichenfolge nicht offensichtlich ist, das muss genauso debuggt werden. das heisst: wenn ich normalerweise eine Eingabe mache, wenn und dann. dann muss ich jetzt eigentlich. das ebenso tun. das macht man wahrscheinlich weniger - bei: sagen wir: suchen von Zeichenfolgen, sortieralgorithmen und so weiter. jetzt schreibe ich das mal hier.

jetzt nehme ich den Code. ich tue ihn in den text und nehme den compiler probiere es aus. da das hier ein editor - ich nehme das raus und mache das in den normalen editor und übersetze das.

```
for (i = 0, j = 0;  i < N; i++, j++) {
  while ((a [i] != p[j]) && (j < M))
     j++;
}
```

so würde das hier aussehen mit der Verwirrung dass

```
j < N
```

ist. ok, jetzt muss ich weiter machen. so jetzt muss ich hier weiter machen. der unterschied, jetzt hole ich mir eine zigarette. nur, dass das

```
j >= 0
```

ist - dann grösser gleich null. so, der unterschied, von zeichenketten im muster. das kann man so machen. so, also, das lassen wir

```
N, M
```

das macht aber nichts, wegen

```
next
```

so, wenn wir das haben. dann ist der unterschied wieder. und das ist der unterschied. wir können mit den gewöhnlichen algorithmus im muster nach muster suchen. das ist wie wenn wir im text nach dem text suchen, das probieren wir aus. jetzt gleich. blos: wir nehmen den C Compiler zur hilfe. wir müssen das wirklich ausprobieren.

jetzt machen wir den code von gerade eben besser:

```
for (i = 0, j = 0;  i < N; i++) {
  while ((a [i] == p[j]) && (j < M))
     j++;
}
```

in diesem falle müsste das so heissen. der unterschied ist. so: dass, wir müssen diese dinger hier auswendig lernen, und die gehen so:

```
for (i = 0, j = 0;  i < N; i++,...) {
  while ((a [j] != p[j]) && (j >= 0))
     j++;
     ...
}
```

das muss man allerdings klassischer weise auswendig lernen. an der stelle wird der code nicht funktionieren, weil: wir müssen das

```
j = 0
```

setzen, wenn das muster nicht gefunden wurde. das kann man vielleicht aussen machen, mit:

```
for (i = 0, j = 0;  i < N; i++, j = 0) {
  while ((a [i] == p[j]) && (j < M))
     j++;
}
```

und wenn wir das haben. dann mache ich das jetzt so. und probiere es aus, danach gucke ich in Algorithmen in C. im buch, ob das stimmt. so jetzt. so, jetzt habe ich das gestet

```
/* (c) david vajda
Suchen in Zeichenfolgen ...
... ohne KMP
11/08/2026 */
```

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

char a [] = "hallo welt sagen manche leute";
char p [] = "sagen";

int main (void) {
 int i, j;
 int N = strlen (a);
 int M = strlen (p);
 for (i = 0, j = 0;  i < N; i++, j = 0) {
    while ((a [i] == p[j]) && (j < M))
        j++;
 }
 if (j == M)
  printf ("Zeichenfolge gefunden\n");
 else
  printf ("Zeichenfolge nicht gefunden\n");

return 0;
}
```

so und wenn wir das haben. dann ergibt die übersetzung

```
davidvajda@localhost ~ % ./grobzeichenfolgesuche11082026
Zeichenfolge nicht gefunden
davidvajda@localhost ~ %
```

kurz und gut: also, ich verbessere das an der stelle, dass ich

```
M -1
```

mache. so. die zeichenfolge ist immer noch nicht gefunden:

```
/* (c) david vajda
Suchen in Zeichenfolgen ...
... ohne KMP
11/08/2026 */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

char a [] = "hallo welt sagen manche leute";
char p [] = "sagen";

int main (void) {
 int i, j;
 int N = strlen (a);
 int M = strlen (p);
```

```
 for (i = 0, j = 0;  i < N; i++, j = 0) {
    while ((a [i] == p[j]) && (j < M))
        j++;
 }
 if (j == (M-1))
  printf ("Zeichenfolge gefunden\n");
 else
  printf ("Zeichenfolge nicht gefunden\n");

return 0;
}
```

also:

```
davidvajda@localhost ~ % ./grobzeichenfolgesuche11082026
Zeichenfolge nicht gefunden
davidvajda@localhost ~ %
```

vielleicht mit einer

```
for
```

Schleife im inneren. und der unterschied, trotzdem kann man das darauf hin zurückführen. das funktioniert komischerweise auch nicht

```
/* (c) david vajda
Suchen in Zeichenfolgen ...
... ohne KMP
11/08/2026 */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

char a [] = "hallo welt sagen manche leute";
char p [] = "sagen";

int main (void) {
 int i, j;
 int N = strlen (a);
 int M = strlen (p);
/*
 for (i = 0, j = 0;  i < N; i++, j = 0) {
    while ((a [i] == p[j]) && (j < M))
        j++;
 }
*/

 for (i = 0, j = 0;  i < N;  i++) {
  for (j = 0;  (a [i] == p [j]) && (j < M);  j++)
   printf ("%i, %i\n", i, j);
```

```
 }
 printf ("%i, %i\n", j, i);
 if (j > (M-1))
  printf ("Zeichenfolge gefunden\n");
 else
  printf ("Zeichenfolge nicht gefunden\n");

return 0;
}
```

jetzt habe ich eine funktionierende art und weise

```
/* (c) david vajda
Suchen in Zeichenfolgen ...
... ohne KMP
11/08/2026 */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

char a [] = "hallo welt sagen manche leute";
char p [] = "sagen";

int main (void) {
 int i, j;
 int N = strlen (a);
 int M = strlen (p);
/*
 for (i = 0, j = 0;  i < N; i++, j = 0) {
    while ((a [i] == p[j]) && (j < M))
        j++;
 }
*/

 for (i = 0, j = 0;  i < N;  i++) {
  for (j = 0;  (a [i] == p [j]) && (j < M);  j++, i++)
   printf ("%i, %i\n", i, j);
  if (j >= (M-1))
   break;
 }
 printf ("%i, %i\n", j, i);
 if (j >= (M-1))
  printf ("Zeichenfolge gefunden\n");
 else
  printf ("Zeichenfolge nicht gefunden\n");

return 0;
}
```

so, wenn wir das haben. dann tut das jetzt, das problem war. dass

```
i++
```

nicht miterhöht wird. so, das ist das eine. und. dann muss trotzdem ein abbruchkriterium geschaffen werden. jetzt probieren wir die zeichenfolge in der zeichenfolge zu suchen. und wir nehmen ein Minimum an oder - wie auch immer. dazu muss allerdings die zeichenfolge gleich anfangen, ... das hat funktioniert

```
/* (c) david vajda
Suchen in Zeichenfolgen ...
... ohne KMP
11/08/2026 */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

char a [] = "hallohallohallo";
char p [] = "hallo";

int main (void) {
 int i, j;
 int N = strlen (a);
 int M = strlen (p);
/*
 for (i = 0, j = 0;  i < N; i++, j = 0) {
    while ((a [i] == p[j]) && (j < M))
        j++;
 }
*/

 for (i = 0, j = 0;  i < N;  i++) {
  for (j = 0;  (a [i] == p [j]) && (j < M);  j++, i++)
   printf ("%i, %i\n", i, j);
  if (j >= (M-1))
   break;
 }
 printf ("%i, %i\n", j, i);
 if (j >= (M-1))
  printf ("Zeichenfolge gefunden\n");
 else
  printf ("Zeichenfolge nicht gefunden\n");

return 0;
}
```

und jetzt geht die sache weiter. weil wir

```
i = 0
```

haben, müssen das höher ansetzen, dann wird es trotzdem tun. das ergebnis:

```
davidvajda@localhost ~ % ./grobzeichenfolgesuche11082026ZEICHENFOLGEinZEICHENFOLGE
```

```
0, 0
1, 1
2, 2
3, 3
4, 4
5, 5
Zeichenfolge gefunden
davidvajda@localhost ~ %
```

und nötigenfalls: das hat funktioniert.

```
davidvajda@localhost ~ % ./grobzeichenfolgesuche11082026ZEICHENFOLGEinZEICHENFOLGE
5, 0
6, 1
7, 2
8, 3
9, 4
5, 10
Zeichenfolge gefunden
davidvajda@localhost ~ %
```

so und:

```
/* (c) david vajda
Suchen in Zeichenfolgen ...
... ohne KMP
11/08/2026 */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

char a [] = "hallohallohallo";
char p [] = "hallo";

int main (void) {
 int i, j;
 int N = strlen (a);
 int M = strlen (p);
/*
 for (i = 0, j = 0;  i < N; i++, j = 0) {
    while ((a [i] == p[j]) && (j < M))
        j++;
 }
*/

 for (i = 1, j = 0;  i < N;  i++) {
  for (j = 0;  (a [i] == p [j]) && (j < M);  j++, i++)
   printf ("%i, %i\n", i, j);
  if (j >= (M-1))
   break;
```

```
 }
 printf ("%i, %i\n", j, i);
 if (j >= (M-1))
  printf ("Zeichenfolge gefunden\n");
 else
  printf ("Zeichenfolge nicht gefunden\n");

return 0;
}
```

27 so, die sache geht weiter.

jetzt es nicht darum, in der phantasie den code weiter zu entwickeln, sondern es geht darum. erstens mal kann man den code bei robert sedgewick nachlesen, auch von dem groben algorithmus.
zweitens, geht es um alltägliche übungen.

und die mache ich jetzt weiter, da geht es auch im KMP und nicht das regelmässige verständnis, sondern den code üben. was ich auswendig weiss.

27.1 Knuth-Morris-Pratt

erste übung

```
while ((p [i] == p [j]) && (j >= 0))
```

so, müsste die schleife aussehen. ok: ich probiere ein mal den algorithmus in C auf zu schreiben

```
int kmpsearch (char *p, char *a) {
 int i, j, M = strlen (p), N = strlen (a);

 initnext (p);


}
```

ok: merken, erst kommt

```
initnext (p);
```

und davor kommen noch, die länge vom Muster und vom Text feststellen

```
M = strlen (p);
N = strlen (a);
M = strlen (p);
N = strlen (a);

initnext (p);
initnext (p);
```

ok, so weit erst mal: jetzt kommen die schleifen, für den Knuth-Morris-Pratt Algorithmus

```
for (i = 0, j = 0;  j < M && i < N;  i++, j++)...;
```

das ist die schleife vom algorithmus

```
for (i = 0, j = 0;  j < M && i < N; i++, j++) ...;
for (i = 0, j = 0; (j < M) && (i < N);  i++, j++) ...;
for (i = 0, j = 0; (j < M) && (i < N);  i++, j++) ...;

while ((p [i] != p [j]) && (j >= 0))...;
```

ok, diesen quelltext kann man sich dann so merken, ein mal mit

```
p [i] != p [j];  // im muster
a [i] != p [j];  // im text
```

also:

```
for (i = 0, j = 0;  (i < N) && (j < M);  i++, j++) ...;
for (i = 0, j = 0;  (i < N) && (j < M);  i++, j++) ...;

while ((j >= 0) && (a [i] != p [j]))...;
while ((j >= 0) && (p [i] != p [j])) ...;
while ((j >= 0) && (a [i] != p [j])) ...;
```

und jetzt geht der algorithmus weiter.

```
int kmpsearch (char *p, char *a) {
 int i, j, M = strlen (p), N = strlen (a);

 initnext (p);

 for (i = 0, j = 0;  (i < N) && (j < M);  i++, j++)
  while ((j >= 0) && (a [i] != p [j]))
   j = next [j];
 ...
}
```

also, hier geht die sache weiter: jetzt kommt der andere algorithmus, im ersten muss kein

```
j = next [j];
```

stattfinden. das ist das eine. in der äusseren schleife, und was dazu kommt. sonst passt die sache. das ist das eine. und den rest umdrehen.

```
initnext (char *a, char *p) {
 int i, j, M = strlen (p);

 next [0] = -1;

 for (i = 0, j = -1;   i < M;  i++, j++, next [i] = j)
  while ((j >= 0) && (p [i] != p [j])) j = next [j];
}
```

so, müssen wir die sache jetzt anwenden, dass wir:

- in der inneren schleife nur

  ```
  j = next [j];
  ```

  haben. das ist das eine. und das andere draussen, sonst nichts.

also, folgendes

```
int kmpsearch (char *a, char *p) {
 int i, j, M = strlen (p), N = strlen (a);
 initnext (p);

 for (i = 0, j = 0;  (j < M) && (i < N);  i++, j++)
  while ((j >= 0) && (a [i] != p [j]))
   j = next [j];
}
```

in dem wirklichen suchalgorithmus, muss natürlich

```
j = next [j];
```

sein. weil damit wird es initialisiert. das ist das ziel, diese tabelle. verstanden habe ich, sie nachdem, das muster im muster gesucht wurde. ich verstehe auch den mechanismus, bzw. so la la, wie auch immer. trotzdem vom programmieren, erst programmieren, dann den rest. keine erklärung, mechanismus, hier mechanimus, suche des musters am anfang des musters, nicht das. sondern mechanismus mit tabelle.
weiter machen. algorithmus nachher aufschreiben:

```
int kmpsearch (char *a, char *p) {
 int i, j, M = strlen (p), N = strlen (a);

 for (i = 0, j = 0; (j < M) && (i < N);  i++, j++)
  while ((j >= 0) && (a [i] != p [j]))
   j = next [j];
}
```

dahinter folgt noch etwas, ... vergessen

```
int initnext
```

wurde nebenbei vergessen

```
init initnext (char *a, char *p) {
 int i, j, M = strlen (p), N = strlen (a);

 for (i = 0, j = 0;  (j < M) && (i < N); i++, j++, next [i] = j)
  while ((j >= 0) && (p [i] != p [j])
   j = next [j];
}
```

der zweite teil muss bleiben. das ist logisch. im ersten teil wiederum wird - die tabelle

```
next [i] = j;
```

installiert. im nachfolgenden teil und das kann man wegen dem:

```
(p [i] != p [j])
```

erkennen: dass hier der vergleich nicht passt. wenn er nicht passt. kann das nur eine initialisierung für das muster sein. also, es ist das muster, was nicht gefunden wurde. dann passiert das gleiche, was passiert, wenn ich im text nach muster suche. ich suche. und finde es nicht, also initialisiere ich

```
j
```

also, noch mal, den code wollte - ich noch mal schreiben und geht so. wir müssen noch den assembler aufbauen. wir müssen noch die struktogramme/PAP und so weiter machen. also noch mal.

```
int kmpinit (char *a, char *p) {
 int i, j, N = strlen (a), M = strlen (p);

 for (i = 0, j = 0;  (j < M) && (i < N);  i++, j++)
  for ((p [i] != p [j]) && (j >= 0))
   next [j] = j;
}
```

so, jetzt geht der code weiter:

```
int kmpsearch (char *a, char *p) {
 int i, j, N = strlen (a), M = strlen (p);

 kmpinit (p);

 for (i = 0, j = 0; (j < M) && (i < N);  i++, j++, next [j] = j);
  for ((p [i] != p [j]) && ((j >= 0))
   j = next [j];
}
```

ok, hier stimmt was nicht. hier ist was falsch gemacht.

ich habe

```
kmpsearch ()
kmpinit ()
```

und jetzt ist die verwechslung, dass

```
kmpinit ();
```

ok: dann muss dieses zwar in

```
kmpsearch ()
```

trotzdem sind:

```
a, p
```

das erstere muss in die suche, das zweitere tritt zwei mal auf beim initiliasieren. also noch mal.

```
int kmpsearch (char *a, char *p) {
 int i, j, N = strlen (a), M = strlen (p);

 kmpinit (p);

 for (i = 0, j = 0;  (j < M) && (i < N);  i++, j++, next [j] = j) {
  while ((p [i] != p [j]) && (j >= 0))
   j = next [j];
 }
 /* ... */
}
```

also, geht es hier weiter:

```
int kmpinit (char *a, char *p) {
 int i, j, N = strlen (a), M = strlen (p);

 for (i = 0, j = 0;  (j < M) && (i < N);  i++, j++, next [j])
  while ((p [i] != p [j]) && (j >= 0))
   j = next [j];
}
```

ok, das stimmte oben nicht. in den zweiten teil muss es oben in die schleife.

```
int kmpinit (char *a, char *p) {
 int i, j, N = strlen (a), M = strlen (p);

 for (i = 0, j = 0;  (j < M) && (i < N);  i++, j++, next [j] = j)
  while ((p [i] != p [j]) && (j >= 0))
   j = next [j];
}
```

ok, und jetzt kommen die suchen. ok, jetzt: jetzt kommt: wir brauchen eine BNF - eine formale Grammatik.

eine Formale Grammatik für eine Backus-Naur-Form.

```
Expr ::= Expr+Term | Term
Term ::= Term*Factor | Factor
Factor ::= id | (Expr)
```

so, jetzt machen wir unsere sachen weiter. nämlich als nächstes kommt auch noch die beseitigung der linksrekursion.

```
expr  ::= term expr'
expr'  ::= term expr' | e
term  ::= fact term'
term'  ::= *fact term' | e
```

die sache geht so:

```
ExprTerm Expr
Expr+Term Expre
TermFactor Term
Term *Factor Term' | e
Factor  id | (Expr)
```

so, jetzt schreiben wir das ding weiter.

```
expr () {
 term ();
 expR ();
}
expR () {
 term ();
 if (gettoken () == '+')

}
```

$$\begin{aligned}
\text{Expr} &\rightarrow \text{Expr} + \text{Term} \mid \text{Term} \\
\text{Term} &\rightarrow \text{Term} * \text{Factor} \mid \text{Factor} \\
\text{Factor} &\rightarrow \text{id} \mid (\text{Expr})
\end{aligned}$$

$$\begin{aligned}
\text{Expr} &\rightarrow \text{Expr} + \text{Term} \mid \text{Term} \\
\text{Term} &\rightarrow \text{Term} * \text{Factor} \mid \text{Factor} \\
\text{Factor} &\rightarrow \text{id} \mid (\text{Expr})
\end{aligned}$$

nun mit beseitigung der linksrekursion:

$$\begin{aligned}
\text{Expr} &\rightarrow \text{Term Expr'} \\
\text{Expr'} &\rightarrow +\text{Term Expr'} \mid \varepsilon \\
\text{Term} &\rightarrow \text{Factor Term'} \\
\text{Term'} &\rightarrow *\text{Factor Term'} \mid \varepsilon \\
\text{Factor} &\rightarrow \text{id} \mid (\text{Expr})
\end{aligned}$$

```
/* (c) david vajda
11/08/2026
C Bakus Naur Form Compiler */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

typdef char token
char *s = "(5+3)*6";
char *p = s;

int gettoken () {
```

```
        token to = *p;
        p++;
        return to;
}

int Expr () {
        Term ();
        Expr ();
}
int ExpR () {
        if (gettoken () == '+')
                Term ();
        ExpR ();
}
int Term () {
        Factor ();
        TerM ();
}
int TerM () {
        if (gettoken () == '*')
                Factor ();
        TerM ();
}
int Factor () {
        int to = gettoken ();
        else if ((to >= '0') && (to <= '9'))
                return;
        else if (ch == '(')
                Expr ();
                to = gettoken ();
                if (to != ')') {
                        printf ("Error");
                        return -2;
                }
        }
        else {
                printf ("Error");
                return -3;
        }
}
int main (void) {
        Expr ();
return 0;
}
```

also, das hier ist keine lßoung

```
/* (c) david vajda
11/08/2026
C Bakus Naur Form Compiler */
```

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

typedef char token;
char s [] = "(5+3)*6;";
char *p = &s[0];

char gettoken (int i);
int Expr ();
int ExpR ();
int Term ();
int TerM ();
int Factor ();

#define e ';'

char gettoken (int i) {
        token to = *p;

        printf ("%c", *p);
        if (to != e)
                p = p + i;
        return to;
}

int Expr () {
        Term ();
        Expr ();
}
int ExpR () {
        token to = gettoken (1);
        if (to == '+') {
                // gettoken (1);
                Term ();
        }
        else if (to == e)
                Factor ();
        ExpR ();
}
int Term () {
        Factor ();
        TerM ();
}
int TerM () {
        token to = gettoken (1);
        if (to == '*') {
                // gettoken (1);
                Factor ();
        }
```

```
        else if (to == e)
                Factor ();
        TerM ();
}
int Factor () {
        token to = gettoken (1);
        if ((to >= '0') && (to <= '9')) {
                gettoken (0);
                return to -'0';
        }
        else if (to == '(') {
                // gettoken (1);
                Expr ();
                to = gettoken (1);
                if (to != ')') {
                        printf ("Error");
                        return -2;
                }
        }
        else if (to == e)
                printf ("BNF success!\n");
        else {
                printf ("Error");
                return -3;
        }
}
int main (void) {
        Expr ();
return 0;
}
```

```
/*
char gettoken (int i) {
        token to = *p;

        printf ("%c", *p);
        if (to != ';')
                p = p + i;
        return to;
}

int Expr () {
        Term ();
        Expr ();
}
int ExpR () {
        if (gettoken (1) == '+') {
                // gettoken (1);
                Term ();
        }
        ExpR ();
}
int Term () {
        Factor ();
        TerM ();
}
```

```
int TerM () {
        if (gettoken (1) == '*') {
                // gettoken (1);
                Factor ();
        }
        TerM ();
}
int Factor () {
        token to = gettoken (1);
        if ((to >= '0') && (to <= '9')) {
                gettoken (0);
                return to -'0';
        }
        else if (to == '(') {
                // gettoken (1);
                Expr ();
                to = gettoken (1);
                if (to != ')') {
                        printf ("Error");
                        return -2;
                }
        }
        else if (to == ';')
                printf ("BNF success!\n");
        else {
                printf ("Error");
                return -3;
        }
}
int main (void) {
        Expr ();
return 0;
} */
```

besonders erste unauskommentierte form oben nicht und zwar aus folgendem grund. weil, die klammern müssten nicht geschlossen sein.

also, so geht es weiter:

```
/* (c) david vajda
automat nicht deterministisch
11/08/2026 */

int state;
char [] = ' ', ' ', 'A', 'B', ' ', 'C', 'A'
int state1 [] =
int state2 [] =
```

also, was hier nicht geht: ich habe das mit stapel und schlange und dem nicht deterministischen automaten nicht im griff. weil der algorithmus, geht mit einem nicht deterministischen automaten. so.

jetzt geht es weiter.

28 elementare sortierverfahren

also selection sort und insertion sort sind fast das gleiche!

- selection sort
- insertion sort
- bubble sort

blos: insertion sort, tauscht schon inzwischen während dem inneren suchlauf um. während, selection sort, ein element auswählt und es dann am ende, mit dem anfang tauscht. bubble sort ist klar. ok. jetzt mit . also erst mal normal üben

```
// (c) david vajda
// selection sort
// 11/08/2026

int selectionsort (int c, char *a) {
        int i, j;
        int ti;
        int t;
        int N = strlen(a);

        for (i = 0;  i < N;  i++) {
                ti = i;
                for (j = i +1;  j < N;  j++) {
                        if (a [j] < a [ti]) {
                                ti = j;
                        }
                }
                t = a [ti];
                a [ti] = a[j];
                a [j] = t;
        }
return;
}
```

so, ist es auch im buch formuliert. allerdings mit einem

```
int min;
// statt ti
```

also machen wir das noch mal mit

min

folgendes:

```
// (c) david vajda
// selection sort
// 11/08/2026

int selectionsort (int c, char *a) {
        int i, j;
        int min;
        int t;
        int N = strlen(a);

        for (i = 0;  i < N;  i++) {
                min = i;
                for (j = i +1;  j < N;  j++) {
                        if (a [j] < a [min]) {
                                min = j;
                        }
                }
                t = a [min];
                a [min] = a[j];
                a [j] = t;
        }
return;
}
```

ok, und jetzt der insertion sort

```
// (c) david vajda
// insertion sort
// 11/08/2026

int selectionsort (int c, char *a) {
        int i, j;
        int min;
        int t;
        int N = strlen(a);

        for (i = 0;  i < N;  i++) {
                t = a [i];
                for (j = i +1;  j < N;  j++) {
                        if (a [j] < a [min]) {
                                t = a [i];
                                a [i] = a[j];
                                a [j] = t;
                        }
                }
        }
return;
}
```

ok, das müsste insertion sort sein! so, jetzt hier weiter. jetzt kommt der bubble sort.

dann ueben wir shell sort. erst quicksort.

```
// (c) david vajda
// 11/08/2026
// quicksort ... eigenes partition

int quicksort (a [], int l, int r) {
        int i;
        if (r < l) {
                i = partitition (a, l, r);
                quicksort (a, l, i-1);
                quicksort (a, i+1, r);
        }
}

int partition (a [], int l, int r) {

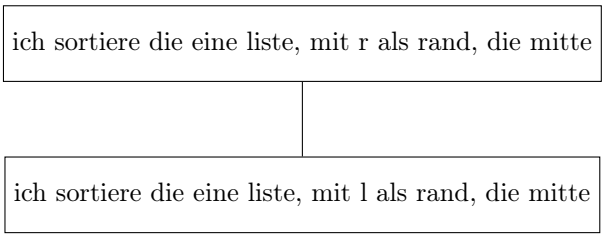
}
```

ok, das bringt mich auf eine idee. dass man das

```
partition (a, l, r);
```

am ende anbringt. das ist das eine. dann kann man es bei drei elementen lassen. nein, das ist was anderes. das eine mischung aus

- shell sort
- quicksort.



bei dem einen, das ist nicht ganz durch, jedes mal alles tauschen, wenn das groesser ist, landet es in der mitte. von der ersten. dann kommen alle aus der ersten partition in die erste die groesser sind. damit verdraengen sie die anderen. und die werden erneut ausgetauscht.

```
int partitition (int a [], int l, int r) {
        int i, j;
        int N = strlen (a);

        v = a [0];
        for (i = 0;  i < (N/2);  i++) {
                if (a [i] > v) {
                        j = i;
                        v = a [i];
                }
        }
```

```
        t = a [i];
        a [i] = a [0];
        a [0] = a [t];

        for (; i < N;  i++) {
                if (a [i] < a [(N/2-i)]) {
                        t = a [i];
                        a [(N/2)-1] = a [i];
                        a [i] = t;
                }
        }
}
```

besser angepasste quicksorts, z.B. bei aktien. wenn der kurs steigend ist, fallend ist, dann zufallszahlen, aber zwei partitionen, dann ist es zum beispiel so angebracht. sonst dauert es lange, wenn man die mitte korrigieren muss, auch nicht unbedingt. weil das mathematisch nicht stimmt. wenn ich das groesste elemente, der erste partition suche und in der zweiten sind die gleichen zufallszhalen, dann sind da so und so viele grosse. und so und so viele kleine und werden entsprechend mit der ersten getauscht.
jetzt lassen wir das so weit. weil das keine arbeit mehr ist. also, jetzt kommt was?

(C) David Vajda
Sun Nov 8 15:57:00 2026
4 Network – TTL – Disjunktive Normalform

0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	0

(C) David Vajda Sun Nov 8 15:57:00 2026 4 Network - TTL - Disjunktive Normalform					
n	x_3	x_2	x_1	x_0	y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	0

so, jetzt kommt aiken, dreiexcess, wie immer, alle codes, als zwischenübung, dies mal ohne zufallszahlen. zunächst die tabellen, dann: das y-Wort.

$$0b0010.0110.1101.1001$$

das ist als

0b0010.0110.1101.1001

ok, so ist die sache ok. jetzt erst tabelle

- Aiken
- Dreieexcess
- 7 Segment
- 1 aus 10
- 2 aus 5

oder in dem falle ist es wirklich besser als lstlisting

Binaer	BCD	Aiken	Dreieexcess
0b0000	0b0000 (0)	pt	0b0000 (0)
0b0001	0b0001 (1)	pt	0b0001 (1)
0b0010	0b0010 (2)	pt	0b0010 (2)
0b0011	0b0011 (3)	0b0011 (0)	0b0011 (3)
0b0100	0b0100 (4)	0b0100 (1)	0b0100 (4)
0b0101	0b0101 (5)	0b0101 (2)	pt
0b0110	0b0110 (6)	0b0110 (3)	pt
0b0111	0b0111 (7)	0b0111 (4)	pt
0b1000	0b1000 (8)	0b1000 (5)	pt
0b1001	0b1001 (9)	0b1001 (6)	pt

```
0b1010   pt                    0b1010 (7)       pt
0b1011   pt                    0b1011 (8)       0b1011 (5)
0b1100   pt                    0b1100 (9)       0b1100 (6)
0b1101   pt                    pt                        0b1101 (7)
0b1110   pt                    pt                        0b1110 (8)
0b1111   pt                    pt                        0b1111 (9)
```

jetzt muss ich hier ordentlich soll ich das eine richtig ordentliche tabelle packen???
muss ich dies mal nicht. oder? ich wuerde sagen lstlisting. also richtige tabelle.
also, LaTeX, Sieben Segment Anzeige, und: 1 aus 10, 2 aus 5.

Binär	BCD	Aiken	Dreieexcess
0b0000	0b0000 (0)	pt	0b0000 (0)
0b0001	0b0001 (1)	pt	0b0001 (1)
0b0010	0b0010 (2)	pt	0b0010 (2)
0b0011	0b0011 (3)	0b0011 (0)	0b0011 (3)
0b0100	0b0100 (4)	0b0100 (1)	0b0100 (4)
0b0101	0b0101 (5)	0b0101 (2)	pt
0b0110	0b0110 (6)	0b0110 (3)	pt
0b0111	0b0111 (7)	0b0111 (4)	pt
0b1000	0b1000 (8)	0b1000 (5)	pt
0b1001	0b1001 (9)	0b1001 (6)	pt
0b1010	pt	0b1010 (7)	pt
0b1011	pt	0b1011 (8)	0b1011 (5)
0b1100	pt	0b1100 (9)	0b1100 (6)
0b1101	pt	pt	0b1101 (7)
0b1110	pt	pt	0b1110 (8)
0b1111	pt	pt	0b1111 (9)

jetzt 1 aus 10, 2 aus 5, 7 segment, 7 segment ist

```
041  042  2000
0f2  6c4  f0104
```

so ist das glaube ich richtig. also

```
00
4f
12
0
```

hier stimmt was nicht, hier muss ich aufhören

```
0b11000
0b10100
0b01100
0b10010
0b01010
0b00110
0b10001
0b01001
0b00101
0b00011
```

jetzt haben wir mal die liste.

```
0b11000 0
0b10100 1
0b01100 2
0b10010 3
0b01010 4
0b00110 5
0b10001 6
0b01001 7
0b00101 8
0b00011 9
```

so, jetzt kommt das nächste thema:

```
0b10000 00000
0b01000 00000
0b00100 00000
0b00010 00000
0b00001 00000
0b00000 10000
0b00000 01000
0b00000 00100
0b00000 00010
0b00000 00001
```

das war 1 aus 10. und jetzt kommt der nächste, das y ding. und da kommt: dann danach ieee-1984/91. tikzpicture/circuitikz. und y jetzt. also das geht so.

```
0b0010.0110.1101.1001
```

hier klassische fehler

```
\begin{tabular}{|c|c|c|c|c|}
\hline
bin? & aiken & dec(aiken) & dreiexcess & dec (dreiexcess)\\
\hline
0b0010 & pt & pt &
0b0110 & 0b0110 & 3
0b1101 &
0b1001 &
\hline
\end{tabular}
```

so, jetzt haben wir das. und der nächste teil, weil das bin oder dump, passt so

bin?	aiken	dec(aiken)	dreiexcess	dec (dreiexcess)
0b0010	pt	pt	0b0010	2
0b0110	b0110	3	pt	pt
0b1101	pt	pt	0b1101	7
0b1001	0b1001	6	pt	pt

nicht, aber dann als aiken. blos, doch passt?

ok, nächste tabelle.

$x' = hex(x)$	$dec(x')$	$aiken(x')$	$dreiexcess(x')$
$0b0010$	$0x2$	$0b0101$	$0b0010$
$0b0110$	$0x6$	$0b1001$	$0b1100$
$0b1101$	$0xD$	pt	pt
$0b1001$	$0x9$	$0b1101$	$0b1111$

jetzt kommt das quine -

also verkürzung - bei DNF - disjunktive Normalform - Gatter IEEE-1984/91

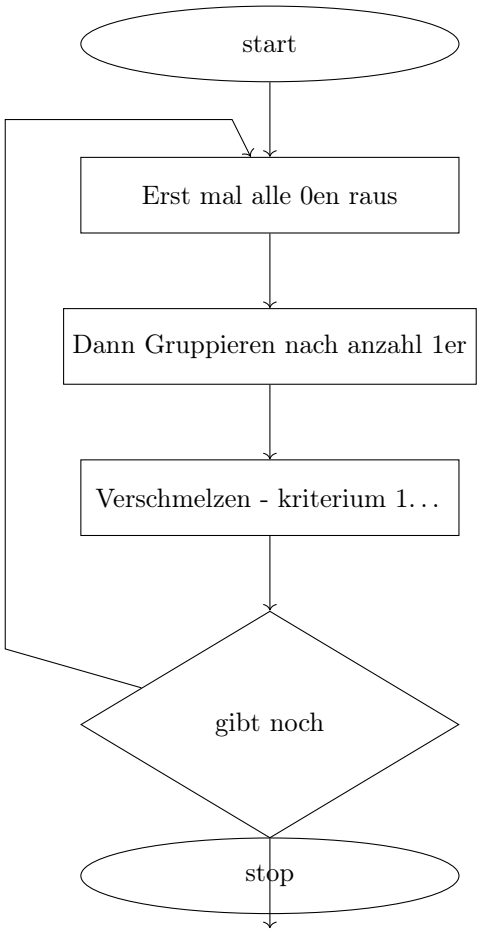
```
(C) David Vajda
Sun Nov  8 15:57:00 2026
4 Network - TTL - Disjunktive Normalform

 0  0  0  0  0     1
 3  0  0  1  1     1
 4  0  1  0  0     1
 6  0  1  1  0     1
 7  0  1  1  1     1
 9  1  0  0  1     1
10  1  0  1  0     1
13  1  1  0  1     1

so, jetzt das naechste:

Gruppe 0:
   0  0  0  0  0     1
Gruppe 1:
   4  0  1  0  0     1
Gruppe 2:
   3  0  0  1  1     1
   6  0  1  1  0     1
   9  1  0  0  1     1
10  1  0  1  0     1
Gruppe 3:
   7  0  1  1  1     1
13  1  1  0  1     1

algorithmischer schritt, die dinger verschmelzen.
```



erst ein algorithmisches diagramm. als PAP.
jetzt haben wir das hinbekommen. jetzt das machen...

```
Gruppe 0:
  0 0 0 0 0      1
Gruppe 1:
  4 0 1 0 0      1
Gruppe 2:
  3 0 0 1 1      1
  6 0 1 1 0      1
  9 1 0 0 1      1
10 1 0 1 0      1
Gruppe 3:
  7 0 1 1 1      1
13 1 1 0 1      1

(0,4)            0 x 0 0
(0,6)            0 1 x 0
(0,3)            0 x 1 1
(6,7)            0 1 1 x
(9,13)           1 x 0 1
```

so, jetzt sind wir so weit.

sind wir da nicht fertig. der nächste teil kommt.

```
Gruppe 0:
  0 0 0 0 0     1
Gruppe 1:
  4 0 1 0 0     1
Gruppe 2:
  3 0 0 1 1     1
  6 0 1 1 0     1
  9 1 0 0 1     1
10 1 0 1 0     1
Gruppe 3:
  7 0 1 1 1     1
13 1 1 0 1     1

(0,4)            0 x 0 0
(4,6)            0 1 x 0
(0,3)            0 x 1 1
(6,7)            0 1 1 x
(9,13)           1 x 0 1

# 10 passt nicht also

10               1 0 1 0
# ok, dann jetzt weiter.

10               1 0 1 0
(6,7)            0 1 1 x
(4,6)            0 1 x 0
(0,4)            0 x 0 0
(9,13)  1 x 0 1
(0,3)            0 x 1 1

# so, jetzt sind wie so weit..

10               1 0 1 0
Gruppe 2:
(6,7)            0 1 1 x
Gruppe 1:
(4,6)            0 1 x 0
Gruppe 0:
(0,4)            0 x 0 0
Gruppe 2:
(9,13)  1 x 0 1
(0,3)            0 x 1 1

# ok, dann ist das so. da ist hier nichts, mehr, also so:

10               1 0 1 0
```

```
(6,7)            0 1 1 x
(4,6)            0 1 x 0
(0,4)            0 x 0 0
(9,13)  1 x 0 1
(0,3)            0 x 1 1
```

jetzt kommt primimplikantentafel.

```
10               1 0 1 0
(6,7)            0 1 1 x
(4,6)            0 1 x 0
(0,4)            0 x 0 0
(9,13)  1 x 0 1
(0,3)            0 x 1 1
```

	0	3	4	6	7	9	10	13
10							1	
(6,7)				1	1			
(4,6)			1	1				
(0,4)	1		1					
(9,13)					1		1	
(0,3)	1	1						

und jetzt zusammen.

	0	3	4	6	7	9	10	13
10							1	
(6,7)				1	1			
(4,6)			1	1				
(9,13)					1		1	
(0,3)	1	1						

die sind alle einzeln. und jetzt weiter... kann man formel machen.

```
10               1 0 1 0
(6,7)            0 1 1 x
(4,6)            0 1 x 0
(9,13)  1 x 0 1
(0,3)            0 x 1 1
```

$$
\begin{aligned}
y &= x_3 \wedge \neg x_2 \wedge x_1 \wedge \neg x_0 && (616)\\
&\vee \neg x_3 \wedge x_2 \wedge x_1 && (617)\\
&\vee \neg x_3 \wedge x_2 \neg \wedge x_0 && (618)\\
&\vee x_3 \wedge \neg x_1 \wedge x_0 && (619)\\
&\vee \neg x_3 \wedge x_1 \wedge x_0 && (620)\\
& && (621)
\end{aligned}
$$

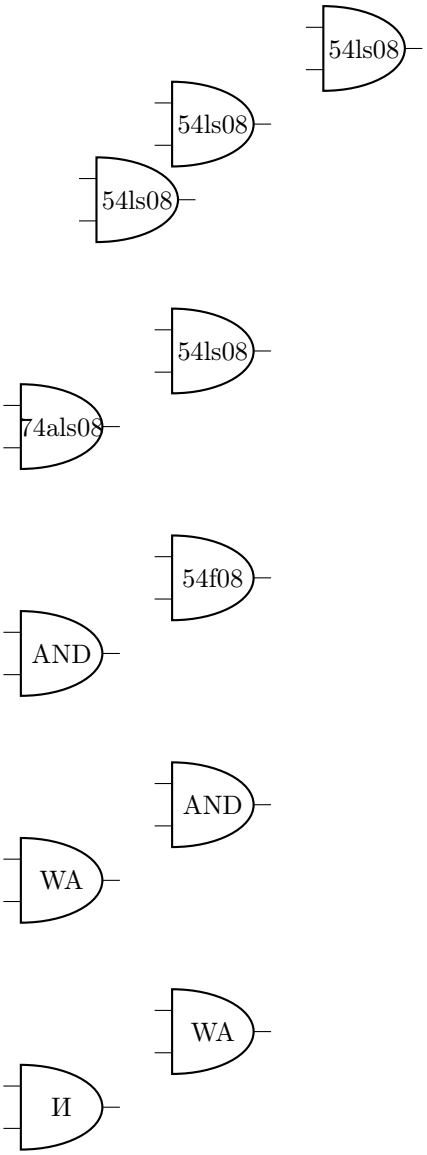
diese gleichung in galasm

```
gal16v8
module gal16v8quineDNF

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc    nc nc nc nc vcc

y =       x3*/x2*x1*/x0 +
          /x3*x2*x1 +
          /x3*x2*/x0 +
          x3*/x2*x0 +
          /x3*x1*x0

description: Die DNF + Quine
```



jetzt geht es weiter.... das sind so viele AND Gatter. aber ok

mal kyrillisch zur abwechslung + russisch.

- 0 – ноль / нуль (nol / nul)
- 1 – один (adin) (oder раз / ras beim reinen Aufzählen)
- 2 – два (dwa)
- 3 – три (tri)
- 4 – четыре (tschetyre)
- 5 – пять (pjat)
- 6 – шесть (schest)
- 7 – семь (sem)
- 8 – восемь (wosem)
- 9 – девять (dewjat)

und logik:

- И (Konjunktion / logisches UND)
- ИЛИ (Disjunktion / logisches ODER)
- НЕ (Negation / logisches NICHT)

ноль И ноль равно ноль

before, we go on, we find relevant expressions (??? my words in german) in - google. like the russians, numbers and arabian in german in sounds like

sifre, wahid, itnan, talata, arba'a hamsa sitta saba'a tamania tissa. ok: and like: wa/aw/la

in russia: ili, i, nje

ok, we start!

we have to bind in the library.

```
\usepackage{fontawesome5}
\usepackage[T2A]{fontenc}          % Kyrillische Schriftkodierung
\usepackage[utf8]{inputenc}
```

we can put it in - this is a little like german, directly in - cyrillic and in ...arabian - ana arabia - ich bin araberin. ok.

mixed: german/russian:

	А	Б	, А ILI Б = (raswe) Ц
nol	istina	istina	istina
ras	istina	losch	losch
dwa	losch	istina	losch
tri	losch	losch	losch

н	А	Б	, А ILI Б = Ц
nol	Истина	Истина	Истина
ras	Ложь	Истина	Ложь
dwa	Истина	istina	Ложь
tri	Ложь	Ложь	Ложь

auf arabisch.

А	Б	, alif wa ba = ya	
sifre	sahih	sahih	sahih
wahid	sahih	hata	hata
itnan	hatah	sahih	hatah
talata	hatah	hatah	hata

wa/aw also aw

ahmad wa ali
with arabian characters:

this is not a problem for working with LaTeX. You have to take bibtex, or lualatex. this is problem for me for all poeple. this works, no problem. and: i not use lualatex, but the table is there. this doesn't work very good, because - the AND like on the IBM-8851-9 i think Latin sth. doesn't work with the other extra signs in table. but i put it in a text file and printeted out,. This is not the destination to make it good with latex make all people languages like indian speaking - we have feelings - but we will learn it for our self, sime time we speak in arabian, indian and all languages .

```
ي= وَ اب
وَ =
\begin{tabular}{|c|c|c|}
\hline
\\ضَطاَ&ضَطاَ&ضَطاَ&\\
ضَجيج&ضَطاَ&ضَجيج\\
&ضَطاَ&ضَجيج&ضَطاَ\\
&ضَجيج&ضَجيج &ضَجيج\\
ي &ب &ا &\\
\end{tabular}
```

now, we go on!

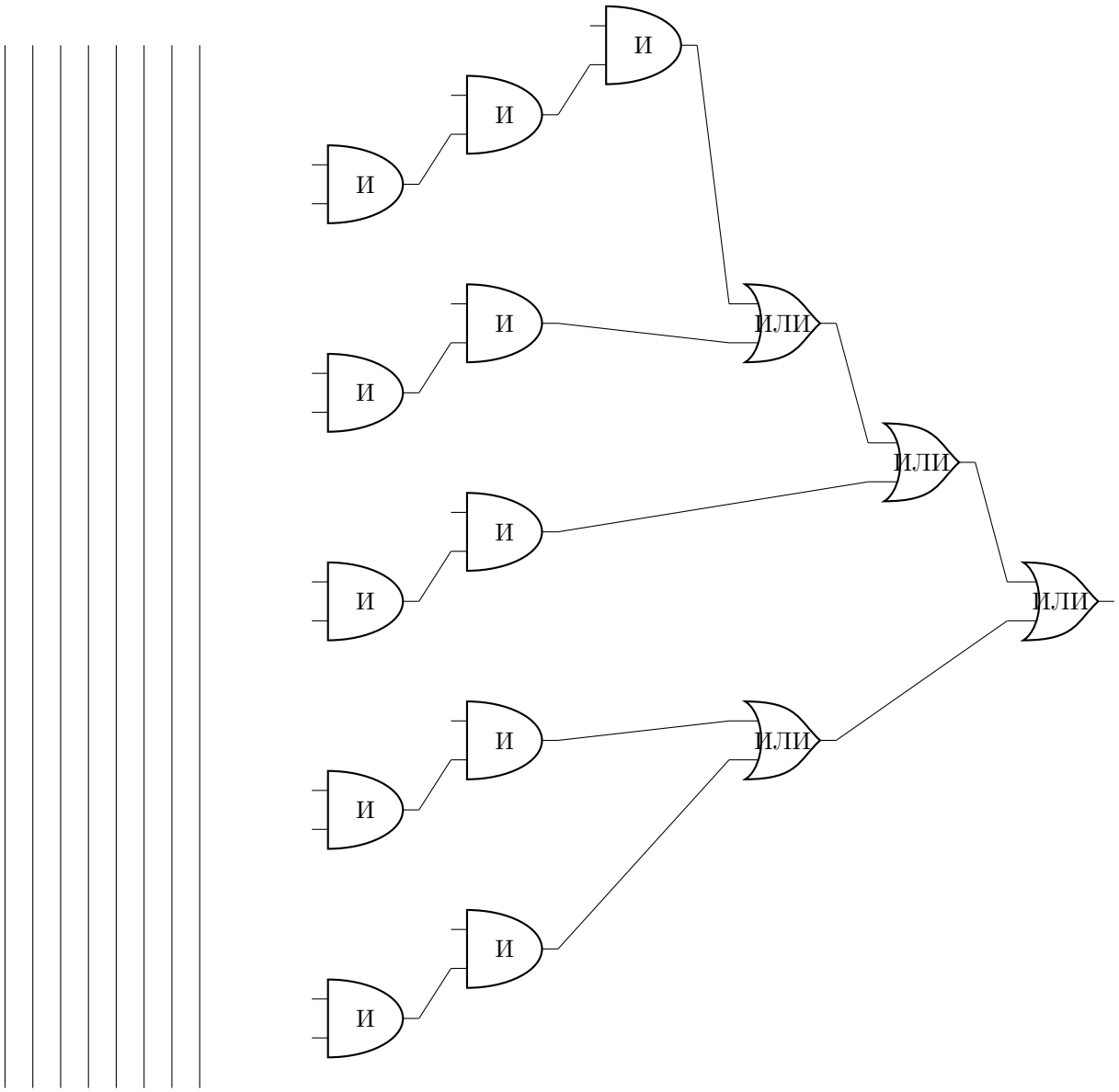
```
gal16v8
module gal16v8quineDNF

clk x3 x2 x1 x0 nc nc nc nc gnd
/oe y nc nc nc   nc nc nc nc vcc
```

```
y =     x3*/x2*x1*/x0 +
        /x3*x2*x1 +
        /x3*x2*/x0 +
        x3*/x2*x0 +
        /x3*x1*x0
```

description: Die DNF + Quine

jetzt geht es weiter.... das sind so viele AND Gatter. aber ok



29 next

```
;; (c) david vajda
;; 11/08/2026
;; m8/code/std exercise

.include "m8def.inc"

ldi r16, HIGH (RAMEND)
out SPH, r16
ldi r16, LOW (RAMEND)
out SPL, r16

ldi r16, 0xff
out DDRB

ldi r16, 0xaa
out PORTB

main01: rjmp main01
```