Prüfung D-CHAB Frühling 2022: Informatik I

03.02.2022 09:30-11:30

HIL E1

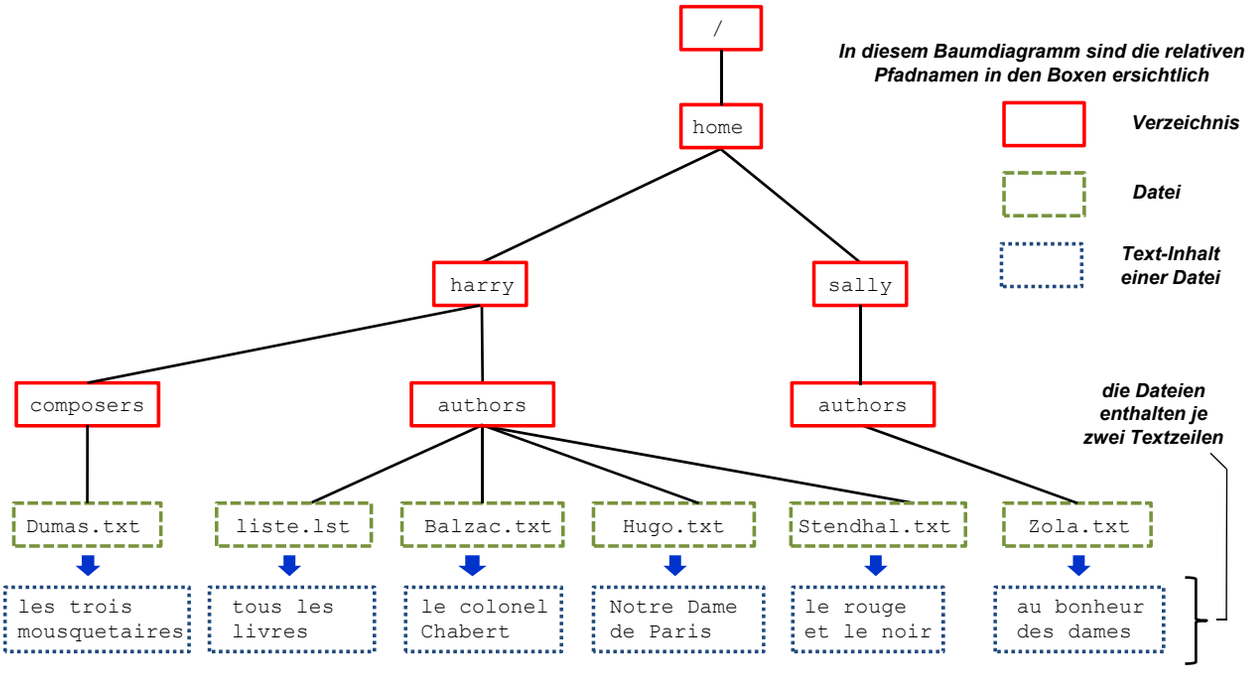
Prof. Philippe H. Hünenberger

Angabe auf Deutsch

- Schreiben Sie Ihren **Namen** und Ihre **Identifikationsnummer (Legi)** auf **jedes** Blatt, welches Sie einreichen.
- Die Benutzung von Laptops, Mobiltelefonen, Taschenrechnern, Büchern, Kursmaterialien *usw.* ist **nicht erlaubt** (Ausnahme: Wörterbücher).
- Sie dürfen Ihre Antworten (oder Teile davon) auf die **Frageblätter** schreiben.
- Bitte **heben Sie** Ihre **abschliessende Antwort** deutlich **hervor**, *z.B.* durch Unterstreichen oder Einrahmen.
- Halten Sie Ihre Antworten **kurz**, aber **klar**.
- Die sechs Aufgaben der Prüfung werden für die Endnote **gleich gewichtet**.

1 UNIX (S2022.1)

Beantworten Sie die unten stehenden Fragen über das *UNIX Betriebssystem* unter Berücksichtigung des folgenden Verzeichnisbaumes



Das Stammverzeichnis (root) ist "/". Ihr Home-Verzeichnis ist "`/home/harry`". Ihr aktuelles Arbeitsverzeichnis ist ebenfalls "`/home/harry`". Erklären Sie die **Wirkungen** der folgenden UNIX Befehle, welche **der Reihe nach ausgeführt werden** (es wird angenommen, dass Sie **sämtliche Berechtigungen** besitzen; beachten Sie auch, dass alle Textdateien jeweils zwei Textzeilen enthalten). Wenn ein Befehl etwas auf den Bildschirm oder in eine Datei schreibt, müssen Sie **exakt angeben**, was geschrieben wird.

a. `mv ~sally/authors/* authors`

b. `cd authors`

c. `cat *.txt | grep le | sort > ../le.lst`

d. `cd`

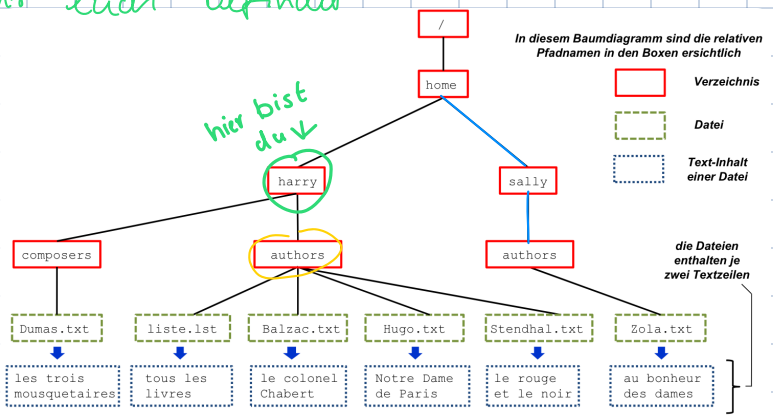
e. `chmod 751 ./le.lst`

f. `cp * ~sally`

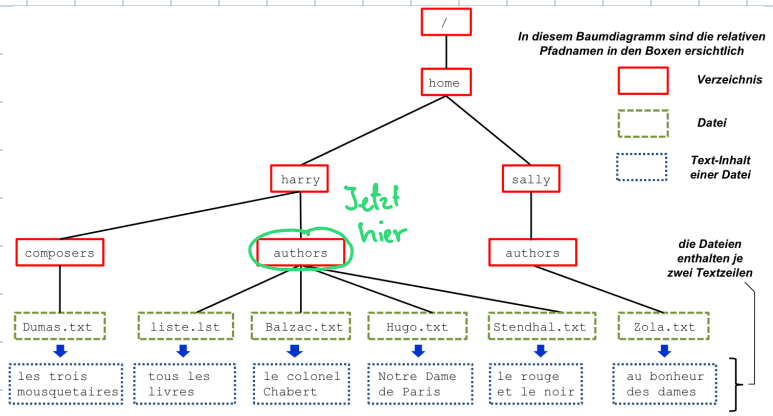
Stellt euch genau vor wo ihr euch befindet

a) mv ~sally/authors/* authors

von wo — was wird bewegt — Zielort



b)



c) cat *.txt | grep le | sort > ../le.list

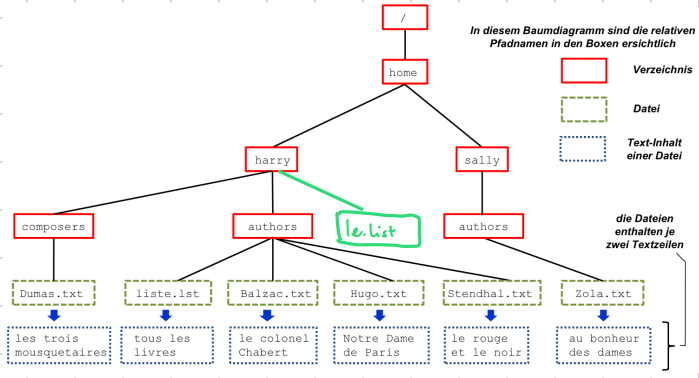
cat: gibt file Inhalt aus

*.txt: alle files in der directory, die auf .txt enden. Also Balzac.txt, Hugo.txt und Stendhal.txt.

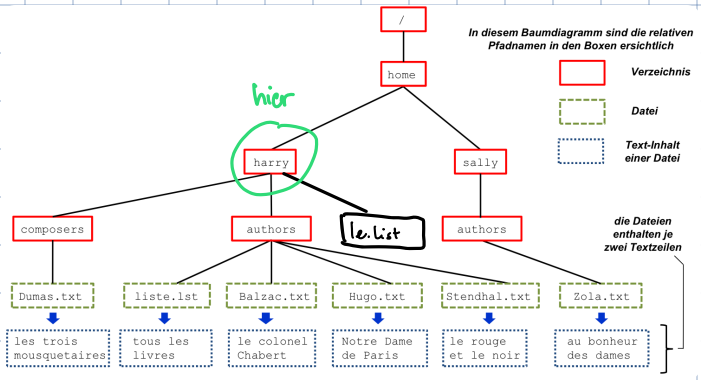
| grep le: filtert den Inhalt der .txt files und gibt jede Zeile die le enthält weiter. Das wäre "le colonel", "le rouge" und "et le noir".

| sort: sortiert alphabetisch: "et le noir", "le colonel", "le rouge"

> ../le.list: erzeugt ein neues file mit dem Namen le.list in der parent directory



d) cd: gehe zum home Verzeichnis



user group others

e) chmod 751 ./le.list

7: rwx = read, write, execute

5: rx = read, execute

1: x = execute

"." : bedeutet in der aktuellen directory.

Statt "./le.list" ginge auch "le.list"

f) cp * ~sally

Da ohne die flag "-r" gearbeitet wird, werden nur files der aktuellen directory nach /home/sally kopiert.

2 Data representation and processing (S2022.2)

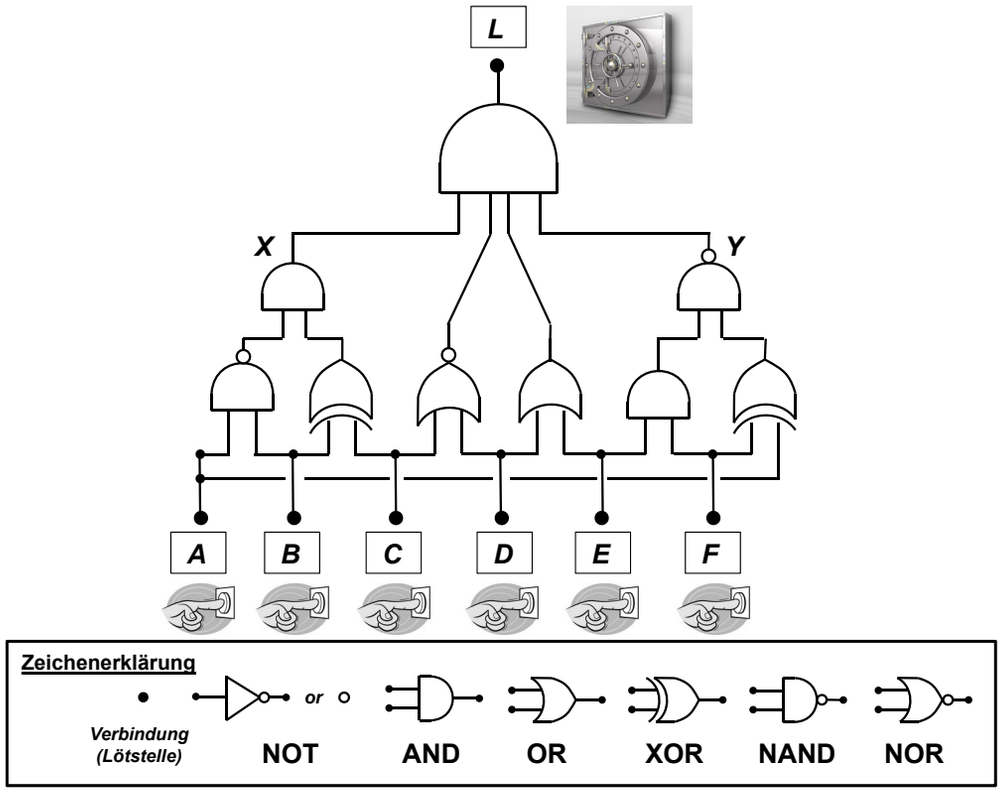
a. Schreiben Sie die *Oktalzahl* $[332]_8$ im **dezimalen** $[\ldots]_{10}$, **binären** $[\ldots]_2$ und **hexadezimalen** $[\ldots]_{16}$ **Zahlenformat** nieder.

b. Ein Computer verwendet die folgende *normalisierte Gleitkommadarstellung* für reelle Zahlen

$$N = (-1)^S \cdot B^{E-7} \cdot [1.M]_2 \qquad 0 < E < 15$$

wobei $B = 2$, die Werte $E = 0$ und $E = 15$ für denormalisierte Zahlen reserviert sind (hier nicht beachtet), und S, E, M durch einen binären 9-Bit String R verschlüsselt sind (1 Bit für S, dann 4 Bits für E, dann 4 Bits für M). **Schreiben Sie die binären 9-Bit Strings** $R_1 = [\ldots]_2$ und $R_2 = [\ldots]_2$ nieder, welche die reellen Zahlen N_1 und N_2 mit den Dezimalwerten $+5.3000$ bzw. -0.1875 verschlüsseln.
Hinweis: Versuchen Sie zuerst, die Zahlen mit einer ganzzahligen Potenz von zwei zu multiplizieren oder zu dividieren, so dass das Ergebnis zum binären Format $[1.M]_2$ passt.

c. Betrachten Sie den folgenden *elektronischen Schaltkreis*, welcher es ermöglicht, das Schloss L eines Safes durch das Drücken einer eindeutigen Kombination der sechs Eingangsschalter A - F aufzuschliessen



Geben Sie die **Anzahl der möglichen Eingangskombinationen** für die sechs Schalter an. Finden Sie dann die **einzige Kombination der Schalter** heraus, welche gleichzeitig gedrückt werden müssen, um das Schloss aufzuschliessen.

Hinweis: Angesichts der Anzahl möglicher Eingangskombinationen ist das Schreiben der Wahrheitstabelle des Schaltkreises ein extrem zeitraubender Weg, um das Problem zu lösen. Versuchen Sie, geschickter zu sein!

a) Schreibe die Zahl in binär um.

① $[3\ 3\ 2]_8 = [011\ 011\ 010]_2$

Du kannst jede der Zahlen in $[]_8$ durch drei bits in $[]_2$ ausdrücken.

② von $[]_2$ nach $[]_{16}$ einfach vierer Gruppen bilden:

$[0\ 1101\ 1010]_2 = [0DA]_{16} = [DA]_{16}$

③ von $[]_{16}$ in dezimal:

$[DA]_{16} = 13 \cdot 16 + 10 \cdot 1 = [218]_{10}$

b) Benutze den Tip

① $5,3 : 4 = (4 + 1,3) : 4 = 1 + 1,3 : 4 = 1 + 0,325$

$= 1,325$

die Zahl ist positiv $\Rightarrow (-1)^S = 1 \Rightarrow S=0$

$5,3 = 1,325 \cdot 2^2$ da $B=2$ muss $2^{E-7} = 2^2 \Rightarrow Eg$

Jetzt muss nurnoch $[1,325]_{10}$ in $[1,M]$ umgewandelt werden

$[1,325]_{10} = 1 + 0,25 + 0,125 = [1,011]_2$ mit $M = [011]_2$ (da 4 bits für M

②

$-0,1875 \Rightarrow S=1$ da negative Zahl.

$-0,1875 \cdot 2 = -0,375$

$-0,375 \cdot 2 = -0,75$

$-0,75 = 1,5 \Rightarrow -1,5 \cdot 2^{-3} = -0,1875 \Rightarrow E = 4$

$[1,5]_{10} = [1,1000]_2 \Rightarrow M = [1000]_2$

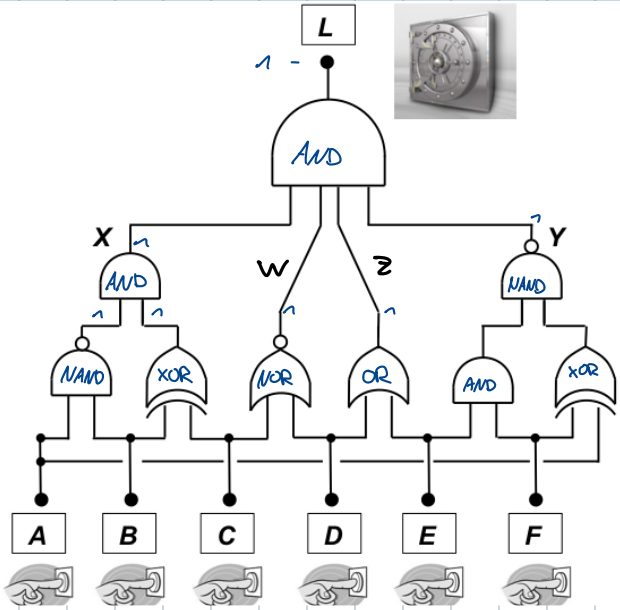
c)

Welche Möglichkeiten haben die Schalter?

Sie sind entweder 0 oder 1, also

2 Stellungen. Wie viel Schalter gibt es? 6!

=> Anzahl möglicher Kombinationen is $2^6 = 64$



Wann ist L true? Fang bei L an und arbeite dich runter. Da vor L ein AND ist, müssen W, X, Y, Z auch true sein. Wann ist X true? Wenn das NAND und XOR true sind

A	B	C	D	E	F	A
0	0	0	0	0	0	0
0	1	1	0	1	1	0
1	0	1	1	0	1	1
1	1	0	1	1	0	1

Muss als NAND true sein

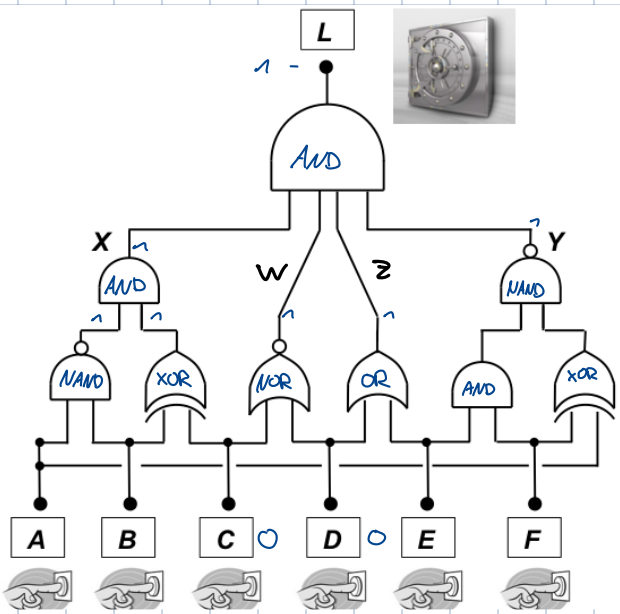
als NOR als OR true sein

↓

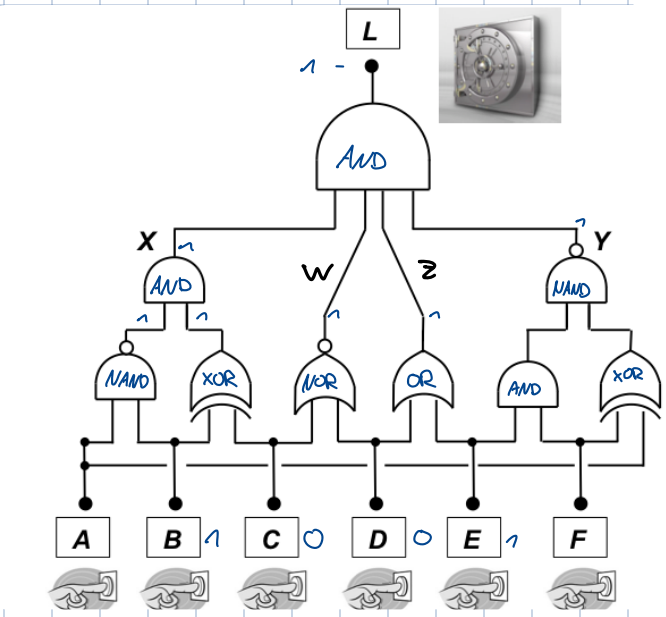
nur true wenn C & D false sind

=) C = D = 0

Warum ich mit NOR angefangen habe? Weil es das einzige ist bei dem es nur eine Kombination gibt



Mit C und D bekannt, können die Nachbarn ausgewertet werden. Damit das OR bei D, E Eins ergibt, muss E = 1 sein. Gleiches Argument für das XOR bei B und C, somit folgt B = 1



Alle anderen Buchstaben folgen analog.

Lösung: A = 0, B = 1, C = D = 0, E = 1, F = 0

Da es nie mehrere Optionen gibt, ist die Lösung einzigartig.

Code sorgfältig durch gehen und auf "Standartfehler" besonders achten

3 Algorithms and programming (S2022.3)

a. Die folgende C++ *Funktion* `list_primes` wurde von einem (nicht sehr bewandten!) Studenten geschrieben, um die Primzahlen in einem Array `arr[]` von `N` Integern (Typ `int`) zu finden, und diese in ein Array `pri[]` der Dimension `M` zu schreiben. Die Funktion soll entweder die Anzahl `m` von Integern zurückgeben, welche in `pri[]` geschrieben wurden, oder einen Fehlerwert von `-1` zurückgeben, falls `M` zu klein ist, um alle diese Werte in `pri[]` zu speichern.

muss int funktion sein

```
01:  void list_primes(int arr, int N, int pri, int M) {
02:      int m = 0;
03:      for ( n = 1; n <= N; ++n) {
04:          div = num = arr[n];
05:          while ( !( num % --div ) ) ;
06:          if ( !div ) { // we found a prime number
07:              if ( m <= M ) // the array mul is big enough
08:                  pri[++m] = num;
09:              else
10:                  return -1;
11:          endif
12:      }}
13:      return n;
14:  }
```

int — nicht declared, int — arr [] — pri[] muss array sein — arr geht von 0 bis N-1 — nicht declared, Typ int — Vergleich — muss weg — !div ist nur bei div = 0 true, soll aber bei div = 1 ins if gehen — wie oben — m++ — gibt es nicht — } } für den for loop — n soll m returnen.

Oft fehlen ";" oder arrays werden nicht korrekt deklariert. Achtet auch darauf in welchem scope die Variable lebt! Fehler sind oft ähnlich in jeder Prüfung

Syntax-Fehler erkennt der Compiler.

Semantic-Fehler werden nicht erkannt, sondern erst bemerkt wenn man den Code executed. Ein out of bounds Zugriff, oder Teilen durch 0.

Logic-Fehler erkennt man erst bei der Auswertung des Ergebnis.

- Erstellen Sie eine Liste **aller Fehler**, die in diesem Code vorkommen, und geben Sie jeweils an, ob es sich um einen **Syntaxfehler**, **Semantikfehler** oder **Logikfehler** handelt. Hinweis: Die Nummerierung zu Beginn jeder Zeile ist nicht Teil des Programmcodes, kann jedoch verwendet werden, um die gefundenen Fehler zu referenzieren.

b. **Erläutern Sie** die folgenden **Konzepte** (nur auf Englisch gegeben) im Zusammenhang mit **Computerarchitektur** (kurz und klar!)

(1) A register and a flag
(2) A coprocessor (+ give two examples)
(3) A bus (+ name the 3 main types of buses in a computer)
(4) Shared *vs.* distributed memory

auswendig lernen

c. Betrachten Sie die folgende C++ *Funktion* `func`

```
int func (int m, int &n) {
  int a = ( m += 2 );
  int b = ++n;
  int c = n++;
  return a + b + c;
}
```

lokale kopie — by referenz, wird global geändert

Schreiben Sie den **Output** von

```
int f = 2, g = 3;
int k = func(f, g);
cout << f << g << k << endl;
```

auf, und **erläutern Sie kurz** Ihren Gedankengang.

durch gehen und jede Wert jeder Variablen aufschreiben

c)

f	g	m	k
2	3		

→ Funktionsaufruf von func, g wird zu n umbenannt m ist (durch Kopie von f

	f	n	m	k	a	b	c
	2	3	2				
Zeile 01 passiert							
	2	3	2+2		4		
" " 02 " - "							
	2	4	4		4	4	
" - " 03 " : "							
	2	5	4		4	4	4
return	f	g	m	4			
	2	5	4	12	/	/	/

Ergebnis: 2 5 12

Variaben definieren die den Termen der Summe entsprechen für $n=0$

4 Algorithms and programming (S2022.4)

Die Funktion $arsinh(x)$ (Areasinus hyperbolicus) mit $|x| \leq 1$ kann als Taylorreihe entwickelt werden zu

$$arsinh(x) = \sum_{n=0}^{\infty} \frac{(-1)^n (2n)! x^{2n+1}}{2^{2n} (n!)^2 (2n+1)}$$
$$= x - \frac{1}{6}x^3 + \frac{3}{40}x^5 - \frac{5}{112}x^7 + \ldots$$

Steps für jede Iteration ausdenken und im loop einfügen

Schreiben Sie eine **effiziente C++ Funktion** `ar_sinh`, welche *arsinh(x)* auf Basis der oben stehenden Entwicklung zurückgibt. **Brechen Sie die Serie ab**, wenn der letzte hinzugefügte Term im Betrag kleiner als ein gegebener Grenzwert `eps` > 0 ist. Die Funktionsdeklaration lautet

```
double ar_sinh (double x, double eps) {
    double x2 = x * x;                      // for the constant x^2
    int n2p = 1, m1 = 1;                        // for 2n+1 and
(-1)^n
    double facn2 = 1, fac2n = 1;            // for (n!)^2 and (2n)!
    double powx = x, pow2 = 1;               // for x^(2n+1) and 2^(2n)
    double sum = x, term = x;                // first term with n = 0
    int n = 0;
    while (term < -eps || eps < term ) {   // magnitude of last term larger than eps
        n++;
        m1 *= -1;
        n2p += 2;
        facn2 *= n*n;
        fac2n *= 2*n * (2*n-1);
        powx *= x2;
        pow2 *= 4;
        term = fac2n * powx / (pow2 * facn2 * n2p);
        sum += term;
    }
    return sum;
}
```

Ergebnis

Summen Term

oder eps * eps < term * term

5 Handling of chemical structures (F2020.5)

Die Struktur eines Moleküls, welches aus `N` Atomen der Typen C, N, O oder H besteht, kann durch einen *Atomtypenvektor*, gespeichert in einem Array `atoms[0..N-1]` des Typs `int` (mit den Codes `1` für C, `2` für N, `3` für O und `4` für H), zusammen mit einer symmetrischen *Bondtypenmatrix*, gespeichert in einem zweidimensionalen Array `bonds[0..N-1][0..N-1]` des Typs `int` (mit den Codes `0` für nicht gebunden, und `1`, `2` oder `3` für einfach, zweifach bzw. dreifach gebunden) repräsentiert werden.

a. **Zeichnen Sie die Struktur des Moleküls**, welches aus `N=14` Atomen besteht, wobei

`atoms[0..13] = (1, 1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 4, 4, 4)`

und

`bonds[0..13][0..13] =`

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

Zeigen Sie auch die **Atomnummern** (und Elementarladungen falls nötig) in Ihrer Zeichnung.

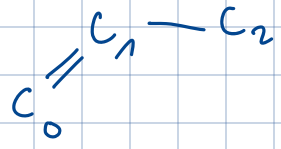
b. Welche Art **topologischer Information**, welche für die Struktur eines Moleküls relevant ist, **umfasst** diese Darstellung **nicht**?

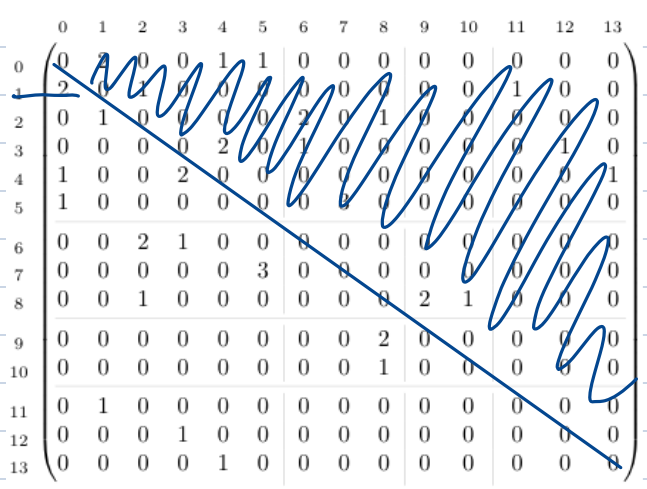
c. In welcher Situation könnte man **zwei unterschiedliche Darstellungen** haben, welche eigentlich dem **gleichen Molekül** entsprechen?

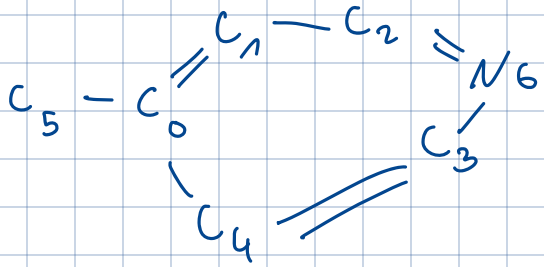
d. Schreiben Sie eine `C++` Funktion `TotalCCBonds`, welche die totale Anzahl Kohlenstoff-Kohlenstoff Bindungen (einfach sowie mehrfach) in einem Molekül zurückgibt. Die Funktionsdeklaration lautet

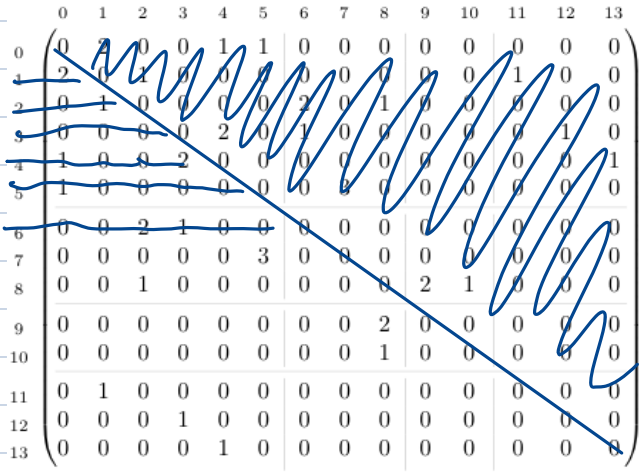
```
int TotalCCBonds (int N, int atoms[], int bonds[][]);
```

a) Fange bei Atom 0 an. Nummeriere die Atome in deiner Zeichnung. Gehe entweder Zeile für Zeile oder Spalte für Spalte durch die Matrix

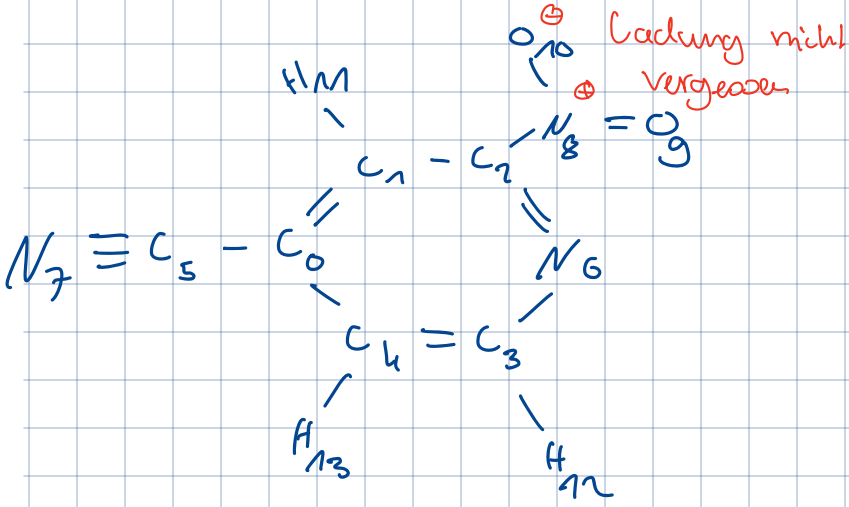


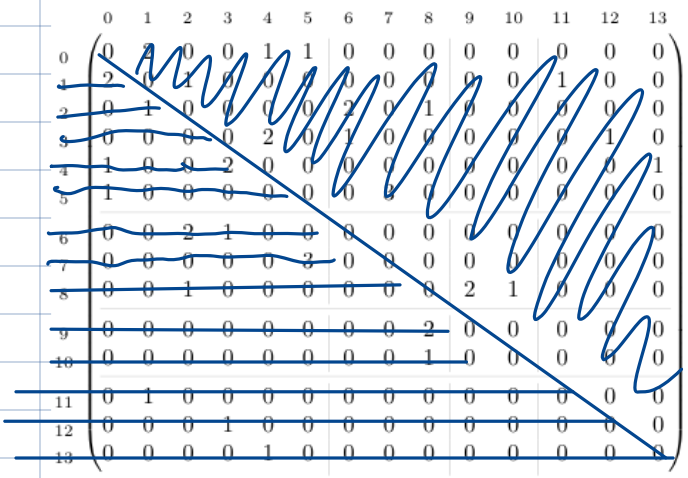






Sieht aus wie ein Phenylring, es kommt sich, seine Zeichnung neu zu machen. Manchmal war man nicht auf dem richtigen Weg





d) alles was die Funktion braucht, steht bereits in ihrem Header.

Es gibt viele Wege diese Aufgabe zu lösen. Man könnte z.B. die ganze Matrix durchgehen und immer 2 Atome anschauen und wenn diese gebunden und Kohlenstoffe sind den counter erhöhen. Dafür bräuchte man aber 4 loops, 2 für jedes Atom. Das würde die Lösung nicht schlechter machen, aber umso mehr code ihr von Hand schreibt, desto höher die Wahrscheinlichkeit das es einen Fehler gibt.

Wie es hier mit "Teilpunkten" aussieht kann ich euch nicht sagen. Es schadet aber nicht sich auch an dieser Aufgabe zu versuchen.

Musterlösung:

d. A possible function `TotalCCBonds` reads

sollte mit expliziter Dimension deklariert werden.

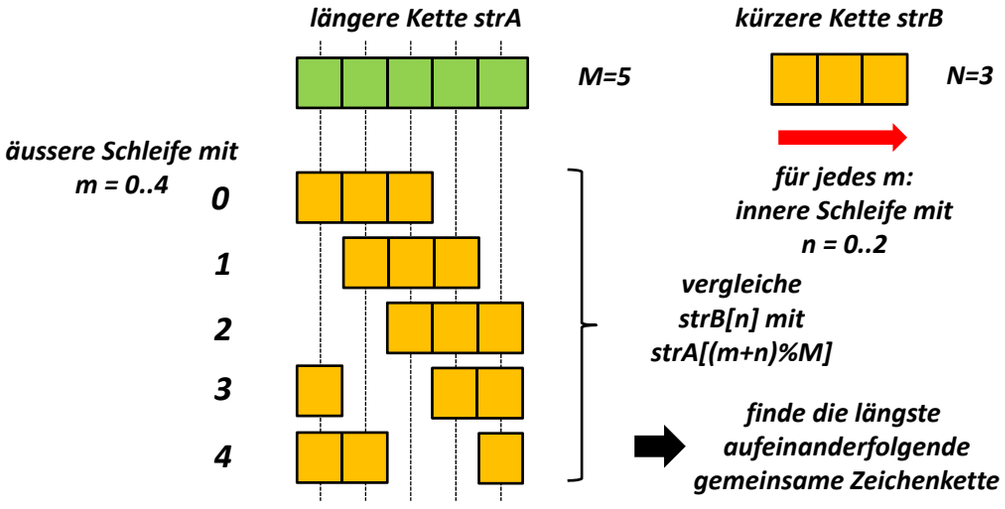
```
int TotalCCBonds ( int N, int atoms[], int bonds[][]) {
    int num = 0; // variable fürs Zählen
    for (n = 0; n < N; n++) { // geht den Atomtyp Vektor durch
        if ( atoms[n] == 1 ) { // sucht nach Kohlenstoffen
            for (i = n+1; i < N; i++) { // schaut sich jede Bindung 1 mal an
                if ( atoms[i] == 1 && bonds[n][i] ) num++;
            }                                  ↳ Mehrfachbindung zählt als 1 Bindung
        }
    }
    return num;
}
```

6 Algorithms and programming (S2022.6)

Schreiben Sie eine `C++` *Funktion* `MaxCommonSubstringLength`, welche die *Länge* des *längsten gemeinsamen Sub-Strings* von zwei Text Strings zurückgibt. Zum Beispiel, wenn der erste String "ToHaveOrNotToHave" ist und der zweite String "ToBeOrNotToBe" ist, dann ist der längste gemeinsame Sub-String "eOrNotTo". Die Funktion sollte dann den Wert 8 zurückgeben, da dies die Anzahl Zeichen in diesem Sub-String ist.

Die Strings sind der Funktion in den Arrays `strA[0..M-1]` und `strB[0..N-1]` des Typen `char` gegeben. Falls die beiden Strings unterschiedlich lang sind, kann angenommen werden, dass `strA` der längere String ist und `strB` der kürzere String ist, *d.h.* es gilt immer $N \leq M$.

Der anzuwendende Algorithmus ist in der untenstehenden Abbildung schematisch aufgezeigt für den Fall, dass $N = 5$ und $M = 3$.



Er beinhaltet eine äussere Schleife über die Variable `m`, welche einen Startpunkt in `strA` für den Abgleich mit `strB` definiert. Für jeden Wert von `m` läuft eine innere Schleife über die Variable `n` für die Zeichen von `strB`. In dieser Doppel-Schleife werden die Zeichen von `strB` "umgefalten", *d.h.* `strB[n]` wird jeweils mit `strA[(n+m)%M]` verglichen.

In der inneren Schleife muss die maximale Anzahl an aufeinanderfolgenden Zeichen von `strA` gefunden werden, für die es eine Übereinstimmung gibt. In der äusseren Schleife muss der maximale Wert dieser maximalen Anzahlen erfasst werden, welche die Funktion am Ende zurückgeben wird.

Zu Beginn der Funktion sollte die Bedingung $N \leq M$ getestet werden. Falls sie nicht erfüllt ist, soll sofort ein Fehlerwert von -1 zurückgegeben werden.

Die Funktionsdeklaration lautet

```
int MaxCommonSubstringLength (char strA[], int M, char strB[], int N);
```

Ein einfacheres Beispiel finde ich ist "ToBeOrNoT" und "Or".

Die Anleitung gibt gut vor, was gemacht werden muss. Und wenn man sie genau befolgt muss man die Aufgabe nicht ganz verstehen.

```
int MaxCommonSubstringLength (char strA[], int M, char strB[], int N) {
    if (N > M) return -1;
    int res = 0;
    for (int m=0; m < M; m++){
        int count = 0;
        for (int n=0; n < N; n++){
            if ((n+m)%M == 0) count = 0; // schaut ob wir das String A wieder vom Anfang durchgehen
            if (strB[n] == strA[(n+m)%M]) count++;
            else {
                if (count > res) res = count; //hält längste Sequenz fest.
                count = 0;
            }
        }
    }
    return res;
}
```