1 Wissensfragen (6 x 0.5 = 3 Punkte)

Welche der folgenden Aussagen ist richtig, welche falsch? (Bitte zutreffendes ankreuzen)

a) Ein Algorithmus terminiert für eine Eingabe v, wenn der Algorithmus bei Eingabe von v nach endlich vielen Schritten anhält. richtig ◯ falsch ◯

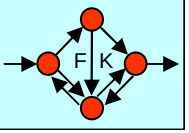
b) Das Halteproblem ist algorithmisch nur mit exponentiellem Aufwand lösbar. richtig ◯ falsch ◯

c) Bei kontextfreien Grammatiken $G = (V, \Sigma, P, S)$ sind nur Regeln der Form $A \rightarrow w$ mit $A \in V$ und $w \in (V \cup \Sigma)^*$ erlaubt. richtig ◯ falsch ◯

d) Mit der Zwei-Komplementdarstellung und und n Bits sind die ganzen Zahlen von -2^{n-1} bis 2^{n-1} darstellbar. richtig ◯ falsch ◯

e) Wenn $L_1 = \{a^n b^m a^n \mid m \geq 1, n \geq 0\}$ und $L_2 = \{b, ab, abaa, abba, a, aba, \varepsilon\}$ sind, dann ist $L_2 \cap L_1 = \{aba, b, \varepsilon, abba\}$. richtig ◯ falsch ◯

f) Gegeben sei folgender Ada-Code:
```
type rf is access float; r : rf;
```
Dann ist die Zuweisung
`begin r:=1.11; end;` erlaubt. richtig ◯ falsch ◯



2 Zwei-Komplementdarstellung (3 x 2 = 6 Punkte)

Wandeln Sie folgende ganze Zahlen in die Zwei-Komplementdarstellung um (verwenden Sie eine 8 Bit Darstellung).

a) hier die Zahl $(-44)_{10}$ eintragen →

b) hier die Zahl $(131)_{10}$ eintragen →

c) hier die Zahl $(-97)_{10}$ eintragen →

3 Gleitpunktdarstellung (3 + 3 = 6 Punkte)

Wandeln Sie $(27.75)_{10}$ in die Gleitpunktdarstellung um. Verwenden Sie 12 Stellen zur Basis 2, wobei die Mantissenlänge 8 und die Exponentenlänge 4 betragen soll. Verwenden Sie für Exponent und Mantisse die Zwei-Komplementdarstellung. Berechnen Sie die Summe der beiden Zahlen.

a) hier die Zahl $(27.75)_{10}$ eintragen →												
Die zweite Zahl →	0	1	1	0	0	0	0	0	1	1	1	0
b) hier die Summe eintragen →												

4 Kontextfreie Grammatiken (3 + 3 = 6 Punkte)

Geben Sie jeweils eine kontextfreie Grammatik an.

a)

In $L(G_1)$ sollen genau die Wörter aus $\{a, b, c\}^*$ liegen, die gleich viele a's wie b's enthalten und nur an erster und letzter Stelle genau ein c besitzen.

$L(G_2) = \{a^n b^m a^n \mid m \geq 1, n \geq 0\}$

5 Syntaxdiagramm (2 + 4 = 6 Punkte)

Ergänzen Sie zeichnerisch die Syntaxdiagramme, für die kontextfreien Grammatiken $G_i = (V_i, \Sigma, P_i, S)$ mit $\Sigma = \{a, b, c\}$.

a) $P_1 = \{(S, aX), (X, aXc), (X, b)\}$ und $V_1 = \{S, X\}$

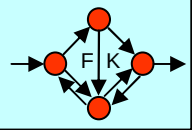
b) $P_2 = \{(S, aXbbYcZ), (X, aXb), (X, \varepsilon), (Y, bYc), (Y, \varepsilon), (Z, aZ), (Z, bZ), (Z, \varepsilon)\}$ und $V_2 = \{S, X, Y, Z\}$

6 Realisierte Abbildung (2 + 2 = 4 Punkte)

Geben Sie die realisierten Abbilungen von P1 und P2 an.

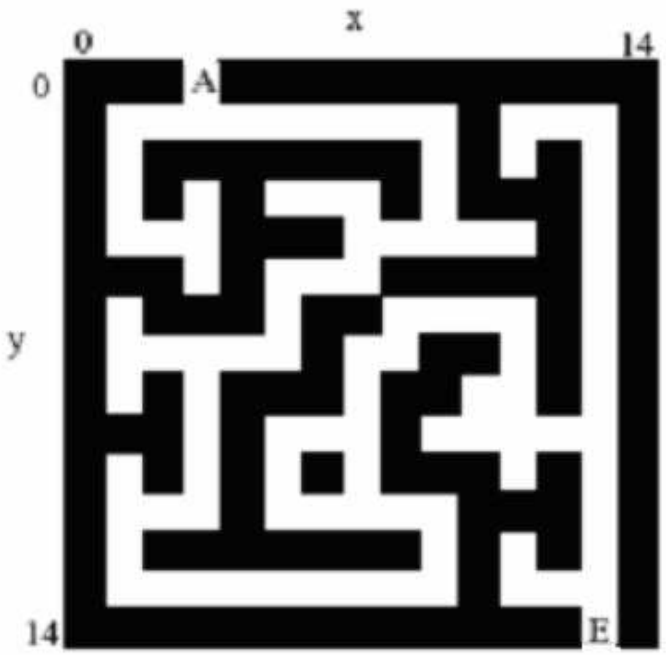
```
program P1 is
declare
a,b : Variablen für ganze Zahlen;
begin
  read(a); read(b);
  if a<0 then a:=a*(-1) fi;
  if b<0 then b:=b*(-1) fi;
  if a<b then
    write(a); write(b) else
    write(b); write(a)
  fi
end
```

```
program P2 is
declare
a,b : Variablen für ganze Zahlen;
v : Variable für eine reelle Zahl;
begin
  read(a); read(b);
  v:=a/b;
  if v<0 then v:=v*(-1) fi;
  write(v)
end
```



Aufgabe 3 (Backtracking)

Finden Sie den Weg aus dem Labyrinth mittels Backtracking.

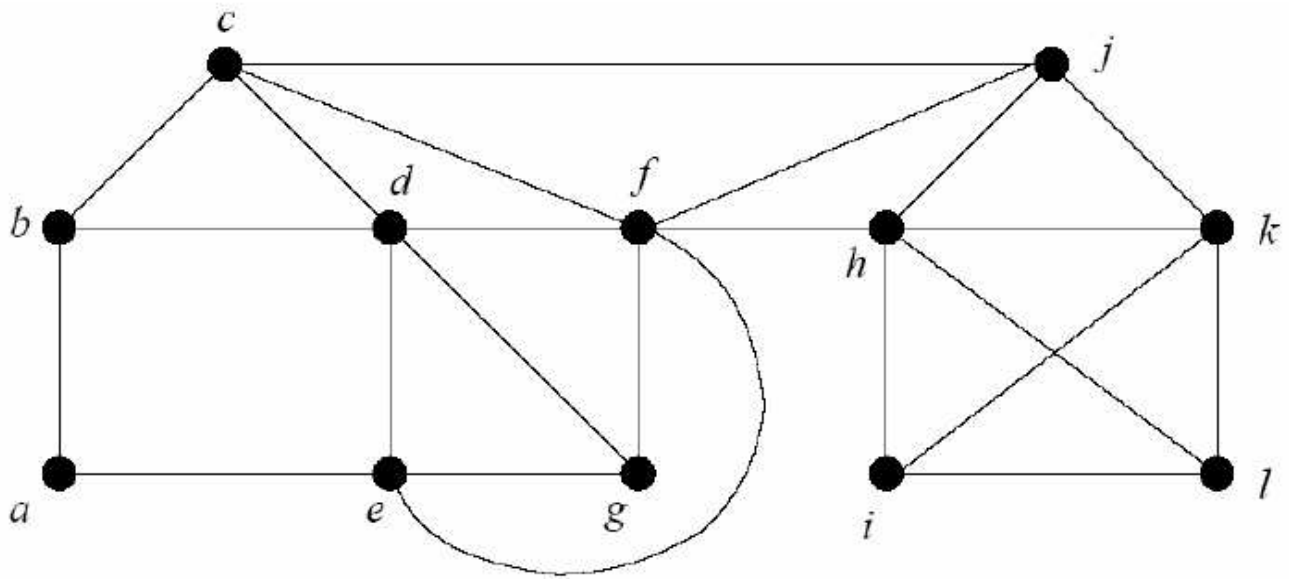


E = Eingang

A=Ausgang

Aufgabe 1 (Tiefen- und Breitensuche)

Führen Sie beim folgenden Graphen eine Tiefen- bzw. Breitensuche durch. Starten Sie die Suche jeweils bei Knoten a. Die Adjazenzlisten sind alphabetisch sortiert.



Aufgabe 2 (Backtracking)

Betrachten Sie folgende Startsituation des 8er-Puzzles:

1	2	3
4	6	
7	5	8

Bestimmen Sie einen Algorithmus, der den Weg zur Zielsituation findet:

1	2	3
4	5	6
7	8	

Versuchen Sie dies mit Mitteln der Graphentheorie zu lösen. Beschreiben Sie einen Erreichbarkeitsgraphen. Beim Verändern von Stellungen im Graphen gehen Sie in der Reihenfolge up, left, down, right vor. Vermeiden Sie direkte Rückzüge (z.B. left - right).

a) Ausgehend von der gegebenen Startsituation expandieren Sie mit der Strategie des Depth-First-Search (Tiefensuche / Backtracking).

b) Ausgehend von der gegebenen Startsituation expandieren Sie mit der Strategie des Best-First-Search (BFS) (Breitensuche). Vergleichen Sie die Anzahl der besuchten Knoten.

Aufgabe 4 (Backtracking)

a) Schreiben Sie ein Programm in einer beliebigen Programmiersprache, welches unter Verwendung von Backtracking alle n - dimensionalen Vektoren mit Komponenten aus {0,1} erzeugt, welche die Eigenschaft besitzen, dass maximal drei aufeinander folgende Komponenten identisch sind. Beispiel für n=6: (0, 0, 0, 1, 1, 0) ist zulässig, (1, 0, 0, 0, 0, 1) dagegen nicht.

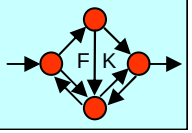
b) Schreiben Sie ein Programm in einer beliebigen Programmiersprache, welches unter Verwendung von Backtracking alle Permutationen der Zahlen (1, 2, ... , n) erzeugt, für die gilt, dass zwei benachbarte Komponenten sich um höchstens 2 voneinander unterscheiden. Beispiel für n=5: (1, 3, 5, 4, 2) ist zulässig, (1, 3, 4, 2, 5) dagegen nicht.

Aufgabe 5

Mit Hilfe von 3 LKW mit Ladevermögen $t1$, $t2$, $t3$ Tonnen sollen t Tonnen ($t \leq t1 + t2 + t3$) der
(nicht partitionierbaren) Güter (mit den Gewichten $g1$, .., gk wobei $g1 + .. + gk = t$) einer Lagerhalle abtransportiert werden. Aus Sicherheitsgründen dürfen manche Güter nicht mit anderen Gütern gemeinsam auf einem LKW transportiert werden. Das Ladevermögen der LKW darf nicht überschritten werden. Wie können die Güter auf die 3 LKW verteilt werden?

Aufgabe 6 (ADT Graph)

Definieren Sie ‚gerichteter Graph' und entwickeln Sie dazu einen ADT.



Iterative Tiefensuche, wobei auf dem Stack Kanten liegen

```
Procedure Tiefensuche (S,Z:Knoten) is
begin
  push (S, S);    (* Speichere Knoten und Vorgänger (S hat keinen) *)
  while ( stack nicht leer ) do begin
    pop ( (x', x) );
    if (not besucht [x]) then begin
      besucht[x] := true;
      if X = Z then fertig;
      for alle y ist Nachbar von x do push (x, y);
    end if;
  end while;
end procedure;
```