Abschlussklausur zu Programmierung 1 im Wintersemester 2014

Geschrieben am 07.02.2014

Vorname _____

Nachname _____

Matrikelnummer _____

Geburtsdatum _____

Studiengang _____

Klausur für 9 Credit Points werten? Ja ☐

Vom Prüfer auszufüllen:

Aufgabe	1	2	3	4	5	6	7	8	9	10	Σ
Punkte	10	10	10	10	10	10	10	10	10	10	100
Davon erreicht											

Klausurpunkte _____

\+ Bonuspunkte _____

Σ _____

Note _____

Generelle Klausurhinweise:

1. Geben Sie auf jedem Blatt (oben rechts) Ihre Matrikelnummer an. Blätter ohne Matrikelnummer können nicht gewertet werden.
2. Schreiben Sie bitte *leserlich*!
3. Kontrollieren Sie Ihre Klausur auf Vollständigkeit. Die Seitenzahlen befinden sich unten rechts.
4. Verwenden Sie die Rückseiten der Klausur ausschließlich für eigene Notizen - diese werden **nicht** gewertet. Die letzte Seite der Klausur ist als *„Schmierpapier“* vorgesehen, oder falls der Platz zum Beantworten einer Frage nicht ausreicht. Zur Benotung muss ein Verweis bei der Aufgabenstellung und eine deutliche Kennzeichnung auf dem Schmierblatt enthalten sein. Benötigen Sie weiteres Papier, melden Sie sich bei der Aufsicht. Selbst mitgebrachtes Papier wird als Täuschungsversuch gewertet!
5. Außer einem dokumentenechten Stift - kein Bleistift - (**nicht Rot**) sind keine weiteren Hilfsmittel zugelassen, wie Handy, Smartphone, Smartwatch, Taschenrechner, Laptop etc. Ein betriebsbereites Handy oder Smartphone wird als Täuschungsversuch gewertet.
6. Die Prüflinge können während der Klausur einzeln die Toilette besuchen. Vor Verlassen des Klausurraumes haben diese bei der Aufsicht ihren Namen anzugeben.
7. Für die Bearbeitung der Klausur stehen 180 Minuten zur Verfügung. In der letzten halben Stunde (30 Minuten) vor Abgabe ist es den Prüflingen untersagt den Raum zu verlassen, um unnötige Unruhe zu vermeiden.

Aufgabe 1: **Multiple Choice** Punkte: ____ / 10

Hinweis: *Die folgenden Fragen sind Multiple-Choice-Fragen, d.h. mehrere Alternativen können richtig sein. Jede Frage erbringt einen Punkt. Es müssen alle richtigen Alternativen und keine falsche ausgewählt sein.*

(a) (1 Punkt) **Formale Sprachen**

Gegeben sei die folgende Grammatik $\mathcal{G}$ in *EBNF* (gemäß ISO 14977) mit dem Startsymbol `S`:

```
S = {A}.
A = A | B | 'a'.
B = AB | B | 'b' | 'ab'.
```

Für welche der folgenden Wörter w gilt $w \in \mathcal{G}$?

- ☐ ε (das leere Wort)
- ☐ BaBBabAB
- ☐ bAbbABab
- ☐ ABAadABA

(b) (1 Punkt) **Pythonanalyse**

Welches Ergebnis liefert die folgende Funktion?

```
def unknown(a):
    for i in range(2, 1 // int(a**0.5)):
        if a % i == 0:
            return i
    return a
```

- ☐ Die Wurzel von `a` für eine Zahl ≥ 2
- ☐ Den Kehrwert des Quadrats für eine positive Zahl `a`
- ☐ Einen ganzzahligen Teiler von `a`
- ☐ Etwas ganz anderes

(c) (1 Punkt) **Python-Ausdrücke**

Welche der folgenden Ausdrücke werden von Python (Version 3.x) zu `False` ausgewertet?

- ☐ `bool(1 // 2 ** 3 % 4)`
- ☐ `True if not False else not False`
- ☐ `bool(3>>2>>1)`
- ☐ `bool(set([1, 2]) - set([2, 3]))`

(d) (1 Punkt) **Built-in-Datentypen**

Über welche primitiven *Built-in*-Datentypen verfügt Python?

- ☐ Array
- ☐ Liste
- ☐ Record
- ☐ Struct

(e) (1 Punkt) **Wörterbuch**
Was gilt für die Schlüssel eines Wörterbuches?

- ☐ Die Schlüssel sind `mutable`
- ☐ Die Schlüssel sind `unmutable`
- ☐ Sie dürfen Tupel enthalten
- ☐ Sie dürfen nicht aus Strings bestehen, da diese `mutable` sind

(f) (1 Punkt) **Funktionen**
Was ist streng genommen nicht Bestandteil der Signatur einer Funktion?

- ☐ Der Name
- ☐ Die Reihenfolge der Parameter
- ☐ Die Typen der Parameter
- ☐ Der Rückgabewert

(g) (1 Punkt) **Objektorientierung**
Gewinnt ein Programm Erkenntnis über seine eigene Struktur, so reden wir von

- ☐ Intension
- ☐ Initiation
- ☐ Introspektion
- ☐ Interaktion

(h) (1 Punkt) **Objektorientierung**
Welche der folgenden „Deklarationen“ macht in Python die Variable `x` pseudo-privat?

- ☐ `private x = 1`
- ☐ `x__ = 1`
- ☐ `x 1`
- ☐ `__x = 1`

(i) (1 Punkt) **Abstrakte Datentypen**
Welche Eigenschaften muss ein abstrakter Datentyp aufweisen?

- ☐ Präzise Beschreibung
- ☐ Vererbbarkeit
- ☐ Entscheidbarkeit
- ☐ Kapselung

(j) (1 Punkt) **Synchronisation**
Wie lässt sich ein kritischer Abschnitt schützen?

- ☐ Mit Semaphoren
- ☐ Durch Abschalten der Interrupts
- ☐ Auf Monoprozessorsystemen gibt es keine kritischen Abschnitte
- ☐ Mit Hilfe von atomaren Aktionen

Aufgabe 2: **Zahlendarstellung** Punkte: ____ / 10

(a) (2 Punkte) Vervollständigen Sie die nachstehende Tabelle gemäß der in der Tabelle gegebenen Zahlenbasen.

Binär	Hexadezimal
0100 1100 0000 1111	
	2ED

(b) (2 Punkte) Vervollständigen Sie die nachstehende Tabelle gemäß des Zweierkomplementes für eine Wortlänge von 8 Bit.

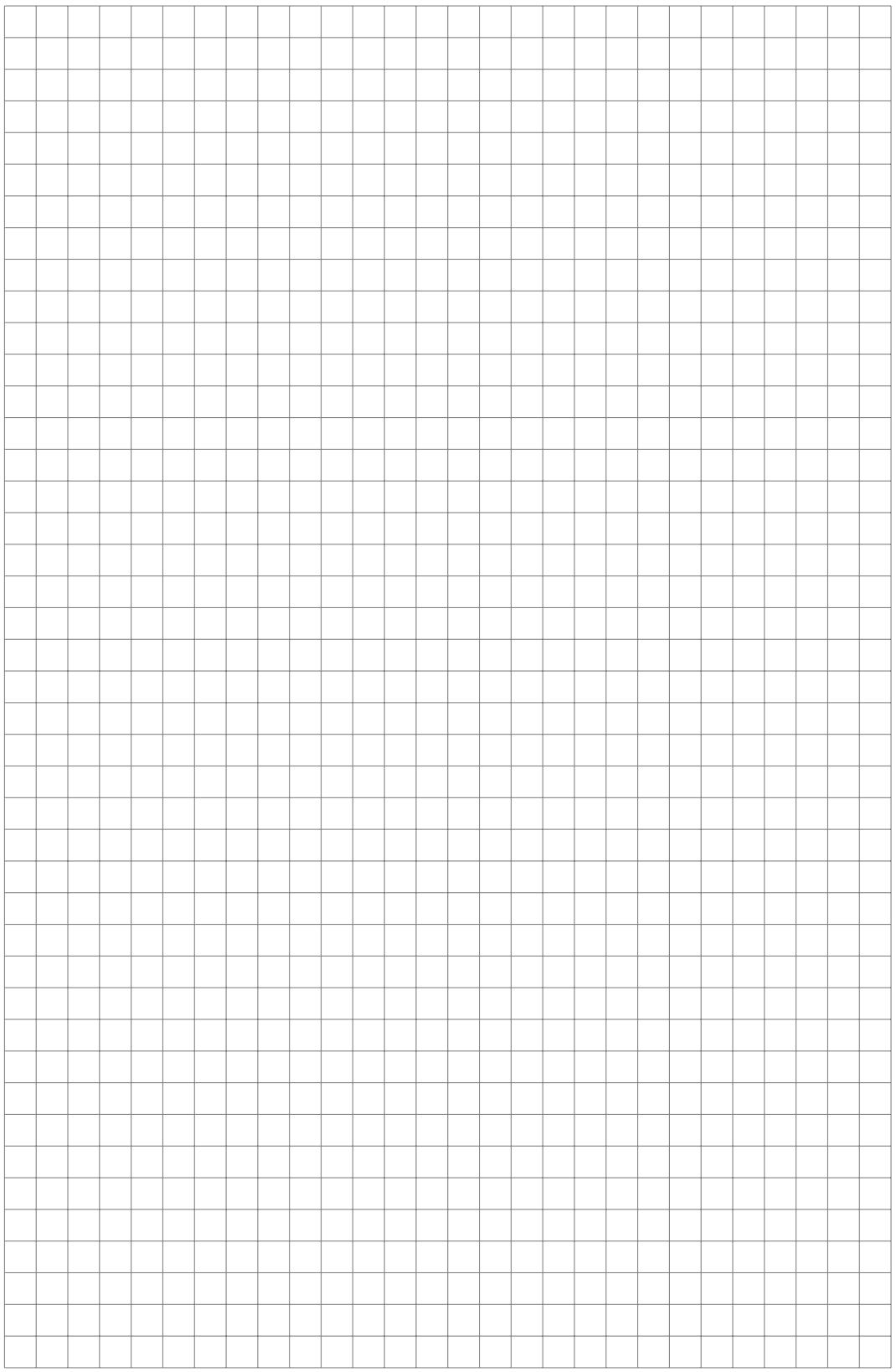
2er-Komplement	Dezimal
0110 1001	
	-86

(c) (2 Punkte) Was ist der Unterschied zwischen dem Einerkomplement und dem Zweierkomplement? Geben Sie ein Beispiel an.

(d) (2 Punkte) Welche Dezimalzahl z wird mit folgenden Bitmuster gemäß *IEEE-754* Codiert? Geben Sie den Rechenweg an.

1 1 0 0 1 0 0 1 1 0 1 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0

(e) (2 Punkte) Geben Sie das Bitmuster und den Rechenweg der Zahl $z = 6.875$ gemäß *IEEE-754* mit einfacher Genauigkeit an.



Matrikelnummer: _____

Aufgabe 3: **Python-Datentypen** Punkte: ____ / 10

Gegeben seien die folgenden Python-Codezeilen:

```
1 a = (2 << 1) / 3 + 2
2 b = 2 ** 3 - 4 * 2.0
3 c = 5 & 3 * (4 >> 2)
4 d = 3 / 2 + 2 ==  3 / 2.0 + 2
```

(a) (2 Punkte) Geben Sie die Auswertungsreihenfolge der Operatoren von links (zuerst) nach rechts (zuletzt) an.

Priorität	1	2	3
Auswertungsreihenfolge der Operatoren aus Zeile 1			
Auswertungsreihenfolge der Operatoren aus Zeile 2			
Auswertungsreihenfolge der Operatoren aus Zeile 3			
Auswertungsreihenfolge der Operatoren aus Zeile 4			

(b) (2 Punkte) Welche Ausgabe erzeugt der folgende Befehl in Bezug auf die Variablen `a`, `b`, `c` und `d` aus der vorherigen Teilaufgabe?
`print(a, b, c, d, end="\t")`

(c) (2 Punkte) Vervollständigen Sie die folgenden Aussagen:

1. Wie viele Zustände lassen sich mit n Bit kodieren?

   1. _____

2. Wie lautet die größte positive Ganzzahl, die sich mit n Bit darstellen lässt?

   2. _____

3. Wie werden *veränderliche* Datentypen in Python genannt?

   3. _____

4. Zählt der Python-Datentyp `str` zu den veränderlichen Datentypen?

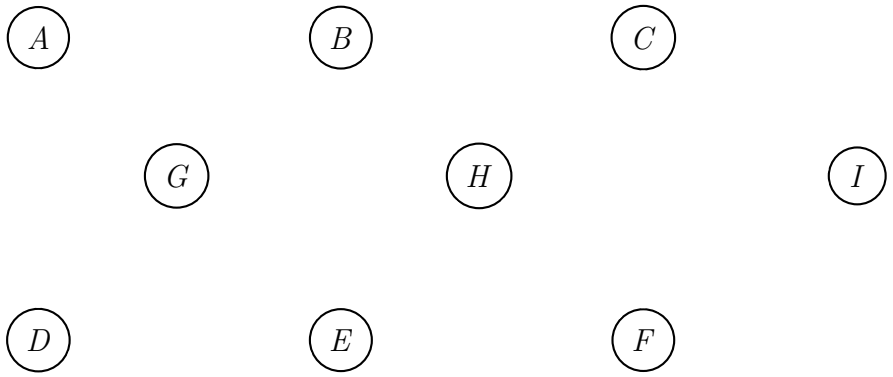
   4. _____

(d) (1 Punkt) Vervollständigen Sie die `print`-Anwseisung mittels *slicing*, so dass der Text `'eipmacscL'` ausgegeben wird!

```
1 s = "Lachsschaumspeise"
2 print(s[                                ])
```

(e) (3 Punkte) Rekonstruieren Sie den Graphen G anhand der Adjazenzliste `L` für die Knoten `A`, `B`, `C`, `D`, `E`, `F`.

```
L = [ [A, [B, D]],
      [B, [A, G, H]],
      [C, [F, H, I]],
      [D, [A, E]],
      [E, [D, G, H]],
      [F, [C, H, I]],
      [G, [B, E]],
      [H, [B, C, E, F]],
      [I, [C, F]] ]
```



Matrikelnummer: _____

Aufgabe 4: **Kontrollstrukturen** Punkte: ____ / 10

(a) (1 Punkt) Überführen Sie die Ackermannfunktion

$$ack(n,m) = \begin{cases} m+1 & , n = 0 \\ ack(n-1,1) & , m = 0 \wedge n \neq 0 \\ ack(n-1, ack(n, m-1)) & \end{cases}$$

in eine rekursive Python Funktion.

```
>>> def ack(n, m):
...
...
...
...
...
...
```

(b) (1 Punkt) Geben Sie eine iterative Implementierung der folgenden Funktion an.

$$f(n) = \begin{cases} n & , n \leq 1 \\ f(n-1) + f(n-2) & \end{cases}$$

```
>>> def f(n):
...
...
...
...
...
...
...
```

(c) (1 Punkt) Überführen Sie den Python-Code in mathematische Notation.

```
def pell(n):
    if n == 0:
        return 0
    if n == 1:
        return 1
    return 2 * pell(n - 1) + pell(n - 2)
```

(d) (1 Punkt) Über welchen sonst oft üblichen **Schleifentyp** verfügt Python nicht?

_____

_____

_____

(e) (2 Punkte) Welche Ausgabe erzeugt folgendes Programm?

```
1  a = 6
2  def foo(bar=1):
3      return bar ** 3
4
5  def bar(b):
6      global a
7      if a<b:
8          a -= 1
9          return foo(b)
10     else:
11         b = b + 1
12         return a + b
13
14 for i in range(5):
15     print(bar(i), end='+')
```

(f) (1 Punkt) Beantworten Sie die folgenden Fragen.

1. Mit welchem Schlüsselwort kann die Ausführung einer Schleife - in Python - übersprungen werden?

   _____

2. Wie lautet das Python-Äquivalent für eine `switch-/case`-Anweisung?

   _____

(g) (3 Punkte) **Merge-Sort**:
Merge-Sort ist ein vergleichsbasiertes Sortierverfahren, dessen Grundannahme darauf aufbaut, dass eine einelementige Menge sortiert ist.

In der *Merge-Phase* (`merge(A, B)`) werden zwei Listen `A`, `B` zu einer sortierten Liste zusammen gesetzt.

```
1  def merge(A, B):
2      merged_list = []
3      while len(A) > 0 and len(B) > 0:
4          if A[0] < B[0]:
5              merged_list.append(A.pop(0))
6          else:
7              merged_list.append(B.pop(0))
8      merged_list.extend(A)
9      merged_list.extend(B)
10     return merged_list
```

Implementieren Sie die **rekursive** Funktion `merge_sort(A)` so, dass diese die übergebene Liste `A` sortiert. Verwenden Sie dabei die Funktion `merge(A, B)`.

```
1  def merge_sort(A):
```

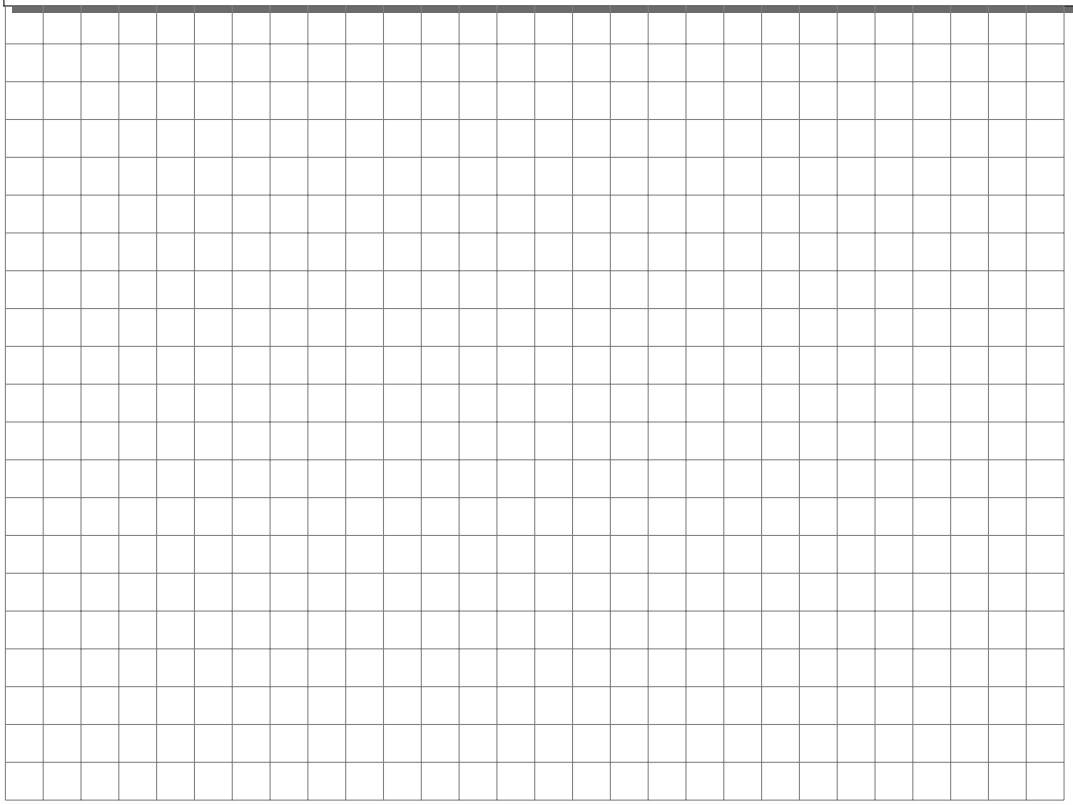
Aufgabe 5: **OOP & OOAD** Punkte: ____ / 10

(a) (3 Punkte) Erstellen Sie aus dem gegebenen Python-Code das zugehörige UML-Klassendiagramm.

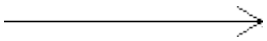
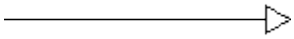
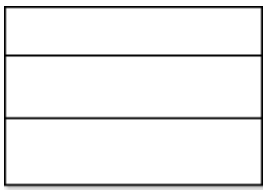
```
class Compress(object):
    def __init__(self):
        self._compression = 0.0
    def encode(self, string:str) -> str:
        pass
    def decode(self, string:str) -> str:
        pass
    def verbose(self) -> str:
        pass

class Huffman(Compress):
    def __init__(self):
        Compress.__init__(self)
        self.frequency, self.character_count = {}, 0
    def H(self, char:str) -> float:
        pass
    def I(self, char:str) -> float:
        pass
    def encode_text(self, text:str) -> str:
        pass
    def compute_entropy(self, length:int) -> float:
        pass
    def count_frequency(self, char:str) -> int:
        pass

class RunLengthEncoding(Compress):
    def __init__(self):
        Compress.__init__(self)
    def write_file(self, path=:str) -> None
        pass
```



(b) (5 Punkte) Benennen Sie die folgenden Elemente der UML.

1. 
2. 
3. 
4. 
5. 

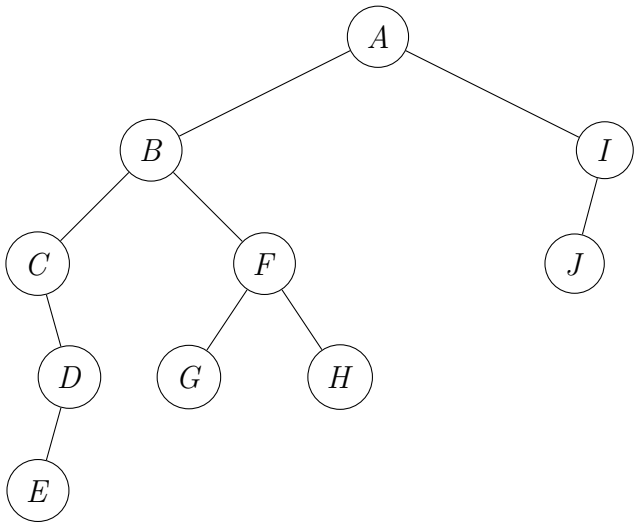
1. _____
2. _____
3. _____
4. _____
5. _____

(c) (1 Punkt) Erläutern Sie den Begriff der *Mehrfachvererbung* im Kontext der Objektorientierung. Welche Probleme können hierbei auftreten?

(d) (1 Punkt) Was ist ein „alternatives“ Konzept in objektorientierten Sprachen, die keine Mehrfachvererbung erlauben? Welcher Nachteil entsteht hierdurch?

Aufgabe 6: **Datenstrukturen** Punkte: ____ / 10

Gegeben sei die folgende ungerichtete Graph $\mathcal{B}$:



(a) (2 Punkte) Stellen Sie den Binärbaum $\mathcal{B}$ in einer **Adjazenzmatrix** dar.

(b) (3 Punkte) Traversieren Sie den Baum und nennen Sie die Reihenfolge, in der die Knoten betrachtet werden für:

Preorder:

_____

Inorder:

_____

Postorder:

_____

(c) (2 Punkte) Mit ineinander geschachtelten Listen lassen sich zweidimensionale Datenstrukturen realisieren. Eine solche zweidimensionale Struktur kann immer auf eine eindimensionale Struktur abgebildet werden. Geben Sie für eine Matrix mit `rows` Zeilen und `cols` Spalten eine Funktion `f(x, y)` an, welche den Index des Elementes in einer eindimensionalen Datenstruktur an Position x aus Zeile y ermittelt und zurück gibt.

```
1 def f(x:int, y:int) -> int:
2     global rows, cols # Anzahl an Zeilen und Spalten
3
4
5
6     return
```

(d) (1 Punkt) Nach welchem Prinzip arbeitet der *Stack*?

(e) (1 Punkt) Nach welchem Prinzip arbeitet eine *Queue*?

(f) (1 Punkt) Wie wird ein Baum bezeichnet, für den in jedem Knoten gilt, dass sich die Höhe des linken und des rechten Teilbaums jeweils um maximal eins unterscheiden?

Matrikelnummer: _____

Aufgabe 7: **Daten – Information – Wissen** Punkte: ____ / 10

Gegeben sei die folgende 32 Zeichen lange Nachricht:

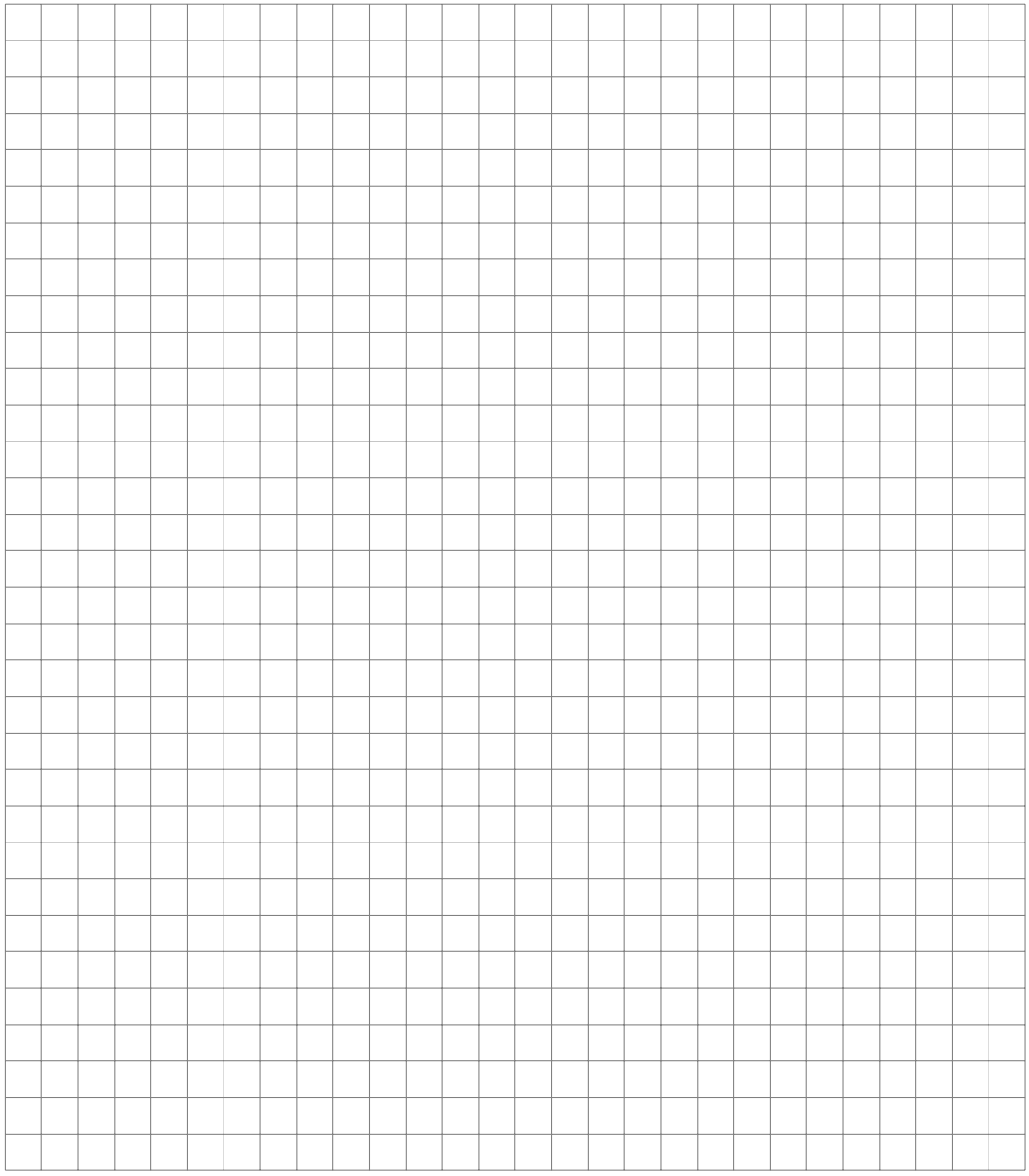
```
OHHHHH_NEIN_ER_HAT_EIN_HANDTUCH!
```

Gehen Sie im Folgenden davon aus, dass alle Zeichen des zugrundeliegenden Alphabets in der Nachricht vorkommen, und dass die Nachricht repräsentativ für die Auftrittswahrscheinlichkeit eines jeden Zeichens ist.

(a) (5 Punkte) Erzeugen Sie aus der Nachricht den Huffman-Code. Zeichnen Sie den zugehörigen Codebaum und geben Sie die Codetabelle an.

	A	!	C	E	D	I
#	2	1	1	3	1	2

	H	O	N	R	U	T	_
#	8	1	4	1	1	2	5



(b) (2 Punkte) Wie groß ist der Informationsgehalt der Zeichen `O`, `A` `N` und `H`? Geben Sie auch den Rechenweg an.

(c) (1 Punkt) Mit was kann die Entropie eines Zeichens einer Informationsquelle gleichgesetzt werden?

(d) (1 Punkt) Wie häufig muss bei der Digitalisierung eines Signals abgetastet werden, um eine fehlerfreie Rekonstruktion zu gewährleisten?

(e) (1 Punkt) Welche Randbedingungen muss das Signal erfüllen um aus dem zeitdiskreten Signal das Ursprungssignal ohne Informationsverlust rekonstruieren bzw. mit endlichem Aufwand, aber beliebig genau, approximieren zu können?

Aufgabe 8: **Debuggen & Testen** Punkte: ____ / 10

(a) (6 Punkte) Finden Sie in folgendem Code alle Fehler.

```
1 def add_queen(new_row, col, old_solution)
2     new solution = ()
3     foreach solution in old_solutions:
4         for new_col in range(0:len(col):-1):
5             if self.attacks(new_row, new_col, solution);
6                 new_solution.append(Solution % [new_col])
7     raise new_solution
```

Fehler in Zeile 1:

Fehler in Zeile 2:

Fehler in Zeile 3:

Fehler in Zeile 4:

Fehler in Zeile 5:

Fehler in Zeile 6:

Fehler in Zeile 7:

(b) (3 Punkte) Zwischen welchen drei Arten der *Bedingungsüberdeckung* wird unterschieden und was besagen diese jeweils?

(c) (1 Punkt) Worin besteht der Unterschied zwischen Black- und Glass-Box-Test?

Aufgabe 9: **Prozesse & Synchronisation** Punkte: ____ / 10

(a) (4 Punkte) Gegeben sei der folgende unvollständige Programmcode zur Lösung des *Erzeuger-Verbraucher-Problems*. Ordnen Sie die 8 fehlenden Befehle den Zeilen (9, 10, 13, 14, 18, 19, 22, 23) im Code zu.

`s_mutex.acquire()` gehört in Zeile _____
`s_mutex.acquire()` gehört in Zeile _____
`s_mutex.release()` gehört in Zeile _____
`s_mutex.release()` gehört in Zeile _____
`s_empty.acquire()` gehört in Zeile _____
`s_empty.release()` gehört in Zeile _____
`s_full.acquire()` gehört in Zeile _____
`s_full.release()` gehört in Zeile _____

```
1  DEPOT, DEPOT_CAPACITY = [], 5
2
3  s_empty = threading.Semaphore(DEPOT_CAPACITY)
4  s_full  = threading.Semaphore(0)
5  s_mutex = threading.Semaphore(1)
6
7  def consumer():
8      while True:
9         # Zeile 9
10        # Zeile 10
11         shoe = DEPOT.pop()
12         print("Sell {0}, {1} shoes left.".format(shoe, len(DEPOT)))
13         # Zeile 13
14         # Zeile 14
15
16 def producer():
17     while True:
18         # Zeile 18
19         # Zeile 19
20         shoe = "pumps"
21         DEPOT.append(shoe)
22         # Zeile 22
23         # Zeile 23
```

(b) (1 Punkt) Erklären Sie in eigenen Worten - unter Verwendung der Begriffe *Betriebsmittel* und *Prozesse* - was in der Informatik als „kritischer Abschnitt“ verstanden wird.

(c) (1 Punkt) Erklären Sie in eigenen Worten, was der „wechselseitige Ausschluss“ in der Informatik bewirkt!

(d) (2 Punkte) Was wird mit dem Begriff *busy wait* umschrieben?

(e) (2 Punkte) Der folgende Code zeigt, wie man theoretisch ein Semaphor programmieren kann. Bei der Verwendung einer solchen Lösung kann es jedoch zu dem Problem kommen, dass Fehler auftreten können. Woran liegt das und wie lassen sich diese Fehler vermeiden?

```
1  def P(d):
2      d.value -= 1
3      if d.value < 0 :
4          enqueue(my_id, d.list)
5          sleep()
6
7  def V(d):
8      if d.value < 0:
9          wakeup(dequeue(d.list))
10     d.value += 1
```

Matrikelnummer: _____

Aufgabe 10: **Algorithmenentwurf** Punkte: ____ / 10

Falls Sie auf der Titelseite das Kreuz für 9 CPs gesetzt haben, entfällt diese Aufgabe für Sie.

(a) (6 Punkte) In der Vorlesung wurden drei Entwurfsmethoden für Algorithmen vorgestellt. Benennen Sie diese und erklären Sie kurz deren Arbeitsweise.

(b) (4 Punkte) Formulieren Sie einen **Greedy**-Algorithmus als Anweisungsvorschrift (oder Pseudocode), der die Elemente l_i einer n-elementigen Liste `L` in aufsteigender Reihenfolge sortiert.