Grundlagen der Programmierung
& teilweise Lösungen

Wintersemester 2018/2019

Aufgabe 1 (10 Punkte)

1. Nennen Sie 3 Merkmale, die den Modellbegriff kennzeichnen.
2. Geben Sie die formale Definition des Begriffs gerichteter Graph an.
3. Was versteht man unter einem Datentyp?
4. Nennen Sie einen Vor- und einen Nachteil von Brute-Force-Algorithmen.

Aufgabe 2 (10 Punkte)
Gegeben ist ein ungerichteter Graph mit der Knotenmenge {a, b, c, d, e, f} durch folgende Adjazenzliste:
adj[a]=[b,c,e]
adj[b]=[a,c,d]
adj[c]=[a,b]
adj[d]=[b,e,f]
adj[e]=[a,d]
adj[f]=[d]

1. Skizzieren Sie die graphische Darstellung des Graphen.
2. Geben Sie die Adjazenzmatrix des Graphen an.
3. Ist der Graph zusammenhängend?
4. Geben Sie einen Kreis an, der sowohl a als auch d enthält.
5. Welchen Abstand haben die Knoten c und f?
6. In welcher Reihenfolge werden die Knoten durch den Algorithmus Breitensuche für ungerichtete Graphen aufgesucht, wenn die Suche bei b beginnt?
7. Welche Reihenfolge ergibt sich, wenn man stattdessen Tiefensuche verwendet?

Aufgabe 3 (10 Punkte)
Der geforderte Algorithmus kann in Python- oder Pseudocode aufgeschrieben werden. Bei Pseudocode muss der Algorithmus soweit verfeinert werden, dass eine Übertragung in eine imperative Programmiersprache ohne weitere kreative Schritte beim Algorithmusentwurf möglich ist.

1. Gegeben ist eine nicht-leere Liste L ganzer Zahlen. Geben Sie einen Algorithmus an, der ausgibt, wie oft die Liste ihr erstes Element (also den Wert L[0]) enthält.
2. Bestimmen Sie die Laufzeit ihres Algorithmus (möglichst detailliert). Erklären Sie die Herleitung.

Aufgabe 4 (5 Punkte)

1. Welche der folgenden Aussagen gelten? Bitte ohne Begründung ankreuzen.
   - $-801 + 10^{-10000}n^2 - 2019n \in O(n^2)$
   - $n \in \Omega(2^n)$
   - $81000000n + 4700000000 \in O(n^2)$
   - $n^{10000} \in O(2^n)$
   - $n + 2^n \in \Theta(2^n)$
   - $n^{10000} \in \Omega(2^n)$
   - $n \in O(1)$
2. Zwei Algorithmen A und B lösen das gleiche Problem. Die Zeitkomplexitäten von A und B sind t_a bzw. t_b. Welcher der beiden Algorithmen ist der effizientere (hinsichtliche Zeitkomplexität) falls $t_a \in O(t_b)$ und $t_a \notin \Theta(t_b)$ gilt?
   A

Aufgabe 5 (5 Punkte)
Betrachten Sie die folgende Klassendefinition, die noch nicht ganz komplett ist:
class Konto:

```
kontonummer: 12345678
kontostand: 0            # Euro
def __init__(this, kn, kohle):          # Kohle in Euro
    this.kontonummer = kn
    this.kontostand = kohle
def einzahlen (this, betrag):           # betrag in Euro
```

1. Ergänzen Sie dide Klassendefinition so, dass bei Aufruf der Methode einzahlen der Kontostand um betrag erhöht wird.
2. Nennen Sie alle Datenelemente der Klasse Konto.
3. Schreiben Sie einen Python Code, den Sie benötigen, um eine Variable myKonto anzulegen, die ein Konto-Exemplat mit der Kontonummer 7654321 und einem Anfangsguthaben von 100€ referenziert. Zahlen Sie dann auf dieses Konto 60€ ein. Verändern Sie die Klassendefinitin dazu nicht weiter.

Aufgabe 6 (12 Punkte)

1. Schreiben Sie folgendes Programm nur unter Verwendung von Zuweisungen und einer while-Schleife!

```
s=1
for i in range (2,12,2):
    s*=i
```

2. Betrachten Sie folgenden Pythoncode.

```
def f(x):
    y=x//2
    return g(y, y+1)
def g(x,y):
    z=x**y
    return z/x x=1 x=f(5) print (x)
```

(a) Wie lauten der formale und der aktuelle Parameter der Funktion f in diesem Programm?
aktuell: 5 formal: x

(b) Welche Ausgabe erfolgt durch die print-Anweisung?
4.0

(c) Zeichnen Sie den Aufrufstack mit allen Stackframes, der unmittelbar nach dem Aufruf der Funktion g entstanden ist (nur ein Stack!). Stellen Sie dabei in den Stackgrames jeweils die Variablen mit ihren Werten dar, wie sie zu diesem Zeitpunkt dort verfügbar sind.

Hinweis: Überlegen Sie sorgfältig, welche Variablen zu diesem Zeitpunkt in welchem Stackframe bekannt sind und welche nicht.

Aufgabe 7 (15 Punkte)

1. Nennen Sie 2 Merkmale der funktionalen Programmierung.

2. Welchen Rückgabewert hat folgende Funktion, wenn sie mit dem aktuellen Parameter 5 aufgerufen wird?

```
def explist(n):
    if (n==0):
        return [1]
    else:
        return explist(n-1) + [2**n]
```

3. Betrachten Sie folgenden Pythoncode:

```
from functools import reduce
le=reduce(lambda x,y: x+len(y),[[1,1],[1],[1,1,1],[1,1]],0)
```

Welchen Wert hat die Variable le?
8

4. Ersetzen Sie jeweils f durch einen lambda-Ausdruck, sodass das Ergebnis des Funktionsaufrufs die Liste [5,6,7,8,9] ist.

   (a) list(map(f, [10,12,14,17,19]))
   f=...

   (b) list(filter(f, [1,2,3,4,5,6,7,8,9,10,11,12,13]))
   f=...

5. Schreiben Sie die Liste [1,8,2,12] als Links- und als Rechtssequenz.

6. Zur Erinnerung: Für eine Rechtssequenz xs geben die Funktionen last(xs) und rest(xs) das letzte bzw. die Rechtssequenz z mit allen Elementen außer dem letzten Element zurück.
   Schreiben Sie Pythoncode mit der Definition einer Funktion addfirst, die folgenden Algorithmus im *funktionalen Stil* realisiert.
   Eingabe: Rechtssequenz ganzer Zahlen xs, ganze Zahl x
   Ausgabe: Rechtssequenz, die aus xs durch Einfügen von x als neues erstes Element entsteht.

Aufgabe 8 (8 Punkte)

1. Nennen Sie eine Gemeinsamkeit von Compilern und Interpretern.
2. Nennen Sie zwei Unterschiede von Compilern und Interpretern. Ordnen Sie dabei die Eigenschaften richtig zu.
3. Schreiben Sie ein AASS-Programm, das den folgenden Pythoncode realisiert:
   if (x < y):
   x=y
   else:
   y=x
   Die Speicherbelegung sei dabei durch folgende Tabelle gegeben:

x	y
4	8

Aufgabe 9 (5 Punkte)

1. Spezifizieren Sie ein Problem, das nicht entscheidbar ist (Eingabe/Ausgabe).
2. Spezifizieren Sie ein Problem, das entscheidbar ist (Eingabe/Ausgabe).
3. Geben Sie die formale Definition des Begriffs abzählbar unendliche Menge an.