Technische Universität Berlin **WS 2018/19**
Fakultät II, Institut für Mathematik
Sekretariat MA 6–2, Antje Schulz
Prof. Dr. Thorsten Koch
Holger Eble, Dr. Frank Lutz, Felix Schröder

Zwischentest zur Computerorientierten Mathematik I

10.12.2018, Beginn: 10:15 Uhr, Bearbeitungszeit: 90 Minuten

Es sind keine Hilfsmittel erlaubt. Bitte nicht mit Bleistift oder mit roter Farbe schreiben!

Nachname, Vorname: _____

Matrikelnummer: _____

Studiengang: _____

1	2	3	4	5	6	Summe
10	8	10	8	10	8	54

Sie haben den Test bestanden, wenn Sie mindestens 50 % der Punkte erreichen.

1. Aufgabe (2 + 2 + 2 + 2 + 2 Punkte)

Geben Sie die asymptotische Laufzeit folgender Codefragmente in Abhängigkeit von der positiven ganzen Zahl n an. Die Laufzeit soll dabei **möglichst knapp** in Θ-Notation ausgedrückt werden. Nehmen Sie dazu an, dass elementare Operationen wie `print`, `+`, `-`, `*`, `/`, `//` und `%` sowie Zuweisungen und Vergleiche jeweils eine konstante Laufzeit haben.

(a)
```
k = 1
while k < n//2:
    k += k
```

Antwort: $\Theta($_____$)$

(b)
```
i = 1
while i < n:
    i -= -2
```

Antwort: $\Theta($_____$)$

(c)
```
i = 1
while n % i != 0:
    i += 1
```

Antwort: $\Theta($_____$)$

(d)
```
for i in range(1,n):
    for j in range(i):
        for k in range(i,j):
            print(k-i)
        print(j)
```

Antwort: $\Theta($_____$)$

(e)
```
for i in range(n):
    print(i)
    if i > 7:
        j = 1
        while j < n:
            j *= 2
```

Antwort: $\Theta($_____$)$

2. Aufgabe (2 + 2 + 2 + 2 Punkte)

Es ist jeweils genau eine Auswahlmöglichkeit anzukreuzen.

(a) Die Ternärdarstellung (zur Basis 3) der Zahl 100 ist:

☐ 10201 ☐ 11021 ☐ 20012 ☐ 10220

(b) In `python3` ergibt der Ausdruck `0xaf + 0b11 + 23` :

☐ 211 ☐ 201 ☐ 189 ☐ 198

(c) Die 8-stellige 2-Komplementdarstellung der Zahl -52 ist:

☐ 10101011 ☐ 10110111 ☐ 11000010 ☐ 11001100

(d) Welcher der folgenden Sortieralgorithmen bietet die beste asymptotische Worst-Case-Laufzeit? (Geben Sie entweder einen der drei genannten Algorithmen an oder, falls mehrere zutreffen, die vierte Option.)

☐ InsertionSort
☐ MergeSort
☐ SelectionSort
☐ Mehr als eine der drei oben genannten sind asymptotisch am besten.

Platz für Nebenrechnungen:

3. Aufgabe

(3 + 3 + 4 Punkte)

Nachfolgend sind einige **python3**-Funktionen gegeben. Geben Sie zu konkreten Eingaben an, was die jeweilige Funktion zurückgibt.

(a)

```
def func1(n,m):
    k = n+m
    while k < n**2:
        k += 2*m
    return k
```

Input: (n, m) = (2,3) Output: _____

Input: (n, m) = (5,3) Output: _____

Input: (n, m) = (3,2) Output: _____

(b)

```
def func2(num):
    if num <= 4:
        return num
    elif num % 5 == 2:
        return func2(num-2)+func2(num-1)
    else:
        return func2(num-5)+5
```

Input: num = 7 Output: _____

Input: num = 8 Output: _____

Input: num = 22 Output: _____

(c)

```
def func3(num, word):
    while num != 0:
        if word[num] == '7':
            return True
        word = word[num//3:num//2]
        num = num//10
    return False
```

Input: num = 4, word = "73737373" Output: _____

Input: num = 3, word = "x7ng34" Output: _____

Input: num = 1, word = "Sieben" Output: _____

Input: num = 10, word = "876543210987" Output: _____

4. Aufgabe (2 + 2 + 2 + 2 Punkte)

Geben Sie bei jeder der folgenden `python3`-Funktionen an, ob die Funktion

(i) bei einer gültigen Eingabe in eine Endlosschleife oder Endlosrekursion laufen kann,

(ii) bei einer gültigen Eingabe einen Laufzeitfehler produzieren kann,

(iii) keinen der vorigen Fehler hat, aber nicht immer das gewünschte Ergebnis berechnet,

(iv) für jede gültige Eingabe terminiert und das korrekte Ergebnis berechnet.

Geben Sie bei jedem Fehler einen Input an, der den Fehler produziert bzw. nicht das gewünschte Ergebnis liefert.

(a) Die folgende Funktion soll genau dann den booleschen Wert `True` zurückgeben, wenn die ganze Zahl `n` in der nichtleeren Liste `L` vorkommt. Andernfalls soll `False` zurückgegeben werden.

```
def occurs(L,n):
    if L[len(L)-1] == n:
        return True
    elif len(L) == 0:
        return False
    else:
        L.pop()
        return occurs(L,n)
```

☐ endlos ☐ Laufzeitfehler ☐ falsches Ergebnis ☐ korrekt

Fehler produzierender Input: _____

(b) Die folgende Funktion soll genau dann den booleschen Wert `True` zurückgeben, wenn die Summe der beiden nichtnegativen Zahlen `n` und `k` durch 6 teilbar ist. Andernfalls soll `False` zurückgegeben werden.

```
def addtozero_mod6(n,k):
    n1 = n-(n//6)*6
    k1 = k-(k//6)*6
    return (n1+k1) % 6 == 0
```

☐ endlos ☐ Laufzeitfehler ☐ falsches Ergebnis ☐ korrekt

Fehler produzierender Input: _____

Teile (c)+(d) auf der nächsten Seite!

(c) Die folgende Funktion soll aus einer nichtleeren Liste `L` die Elemente mit Index `k,k+1,...,len(L)-1` in dieser Reihenfolge in einer neuen Liste ausgeben. Die ganzzahlige Eingabe `k` liege dabei immer zwischen `0` und `len(L)-1`.

```
def extract_sublist(L,k):
    i=k
    result=[]
    while i < len(L):
        item = L.pop(i)
        result.append(item)
        i += 1
    return result
```

☐ endlos ☐ Laufzeitfehler ☐ falsches Ergebnis ☐ korrekt

Fehler produzierender Input: _____

(d) Die folgende Funktion soll eine Kopie der gegebenen Liste `L` erstellen, die beiden Listen konkatenieren (d. h. aneinanderhängen) und die entstehende Liste zurückgeben.

```
def concat(L):
    for i in L:
        L.append(i)
    return L
```

☐ endlos ☐ Laufzeitfehler ☐ falsches Ergebnis ☐ korrekt

Fehler produzierender Input: _____

5. Aufgabe (2 + 2 + 2 + 2 + 2 Punkte)

Geben Sie ohne Begründung an:

(a) Von einem ungerichteten, einfachen Graphen G sei bekannt, dass er n Knoten habe, davon (mind.) einen vom Grad 0 und (mind.) einen vom Grad 1, wobei $n \geq 4$. Wie viele Kanten kann G maximal haben (in Abhängigkeit von n)?

Antwort: _____

(b) Sei G ein einfacher, ungerichteter Graph mit 12 Knoten und 13 Kanten. Was ist die kleinste mögliche Anzahl an Kreisen in G?

Antwort: _____

(c) Zeichnen Sie einen gerichteten Graphen mit 7 Knoten und 7 Kanten, der genau drei starke Zusammenhangskomponenten und keine aufspannende Arboreszenz hat.

(d) Es sei ein gerichteter Graph G durch folgende Adjazenzmatrix gegeben:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

(i) Zeichnen Sie G:

(ii) Geben Sie eine topologische Sortierung von G an.

Antwort: _____

6. Aufgabe (3 + 2 + 3 Punkte)

In dieser Aufgabe sollen Sie selbst `python3`-Code schreiben.

a) Schreiben Sie eine Funktion `get_minidx(intlist)`, die für die Liste `intlist` ganzer Zahlen den Index des kleinsten Eintrags zurückgibt. Die leere Liste als Eingabe soll den Rückgabewert `None` liefern. Die Listeneinträge dürfen als paarweise verschieden angenommen werden. Die Verwendung der `min`- und der `sort`-Funktion ist verboten.

b) Es sei `s` ein String, der aus einer Reihe zusammenhängender Zeichenketten besteht, die jeweils durch ein Leerzeichen voneinander getrennt sind. Schreiben Sie eine Funktion `get_shortest(s)`, welche die kürzeste zusammenhängende Zeichenkette aus `s` zurückgibt. Im Programmverlauf *muss* die obige `get_minidx`-Funktion aufgerufen werden. Für diesen Aufgabenteil dürfen Sie annehmen, dass `get_minidx` korrekt implementiert ist und dass die in `s` vorkommenden Zeichenketten paarweise verschiedene Längen haben.

c) Schreiben Sie eine *rekursive* Funktion `get_primepower(n,p)`, welche den Exponenten ν von p in der Primfaktorzerlegung $n = p_1^{\nu_1} ... p_l^{\nu_l}$ zurückgibt, wobei hier $p_1, ..., p_l$ die Primteiler von n beschreiben. Insbesondere soll 0 zurückgegeben werden, falls n von p nicht geteilt wird.

Die Funktion `get_primepower(n,p)` soll insgesamt $(\nu + 1)$-mal aufgerufen werden.