Name: | Vorname: | Matr.–Nr.:

Technische Universität München
Fakultät für Informatik, I-16
Dr. Stefanie Demirci

SoSe 2018
9. Mai 2018

Probeklausur zu Algorithmen und Datenstrukturen

Allgemeine Hinweise

- Die Angabe besteht aus 9 Seiten. Stellen Sie sicher, dass Ihr Exemplar vollständig ist, und **prüfen Sie alle Seiten auf Aufgaben!**
- Sie haben 90 Minuten Zeit für die Bearbeitung der Aufgaben, von 09:45 bis 11:15 Uhr.
- Ein einziges DIN A4 Blatt mit handschriftlichen Notizen ist als Hilfsmittel erlaubt. **Computer, Mobiltelefone und vergleichbare Geräte sind verboten und müssen während der Klausur ausgeschaltet sein.**
- Sollte Deutsch nicht Ihre Muttersprache sein, dürfen Sie zusätzlich ein Wörterbuch ohne Eintragungen oder Änderungen bereithalten.

Aufgabe 1 Rechnen mit Landau-Symbolen (4 Punkte)

Untersuchen Sie, ob folgende Behauptung wahr ist und zeigen Sie Ihre Aussage.

$$\frac{2}{n^3} - \frac{3}{n^2} + \frac{10}{n} = O\left(\frac{1}{n}\right)$$

Aufgabe 2 Boolesche Logik **(5 Punkte)**

Angenommen, Sie sollen folgende Codezeile

if (a && !b)

schreiben. Leider funktioniert aber die Taste '&' Ihrer Tastatur gerade nicht.

- Formulieren Sie das Problem so um, dass die entsprechende Taste nicht verwendet werden muss!

- Zeigen Sie, dass die Äquivalenz beider Ausdrücke gegeben ist. (Hinweis: Wahrheitstabelle)

Aufgabe 3 Zahldarstellung **(3 Punkte)**

Konvertieren Sie die Zahl $-102,9012_{10}$ in eine 32-bit Fließkommazahl (floating point). (Hinweis: eine 32-bit Fließkommazahl setzt sich aus 1-bit Vorzeichen, 23-bit Mantisse und 8-bit Exponent zusammen. Der Bias für den Exponent ist 127.)

Aufgabe 4 Sequentielle Liste **(4 Punkte)**

Gegeben sei ein Feld *A* implementiert als sequentielle Liste (Array) mit 10 Elementen.

a) Sie möchten nun per Operation `elementAt` auf das dritte Element (A[2]) zugreifen. Welche Komplexität hat diese Operation?

b) Was passiert, wenn Sie nun auf A[10] zugreifen?

c) Welche Komplexität haben die Operationen `insert` und `erase` auf der Liste *A*? Gibt es eine alternative Datenstruktur, die für diese Operationen eine effizientere Lösung bietet? Geben Sie den Namen der Datenstruktur sowie die Komplexität der jeweiligen Operationen an.

Aufgabe 5 Stack **(3 Punkte)**

Nennen Sie kurz *eine* Methode, um den abstrakten Datentyp Stack mittels elementarer Datenstrukturen zu implementieren. Was sind dann jeweils die Komplexitäten der Operation `push` und `pop`?

Aufgabe 6 Fibonacci-Folge, Verifikation **(6 Punkte)**

Die Fibonacci-Folge $(f_n)_{n\in\mathbb{N}}$ ist rekursiv definiert als:

$$\begin{aligned} f_1 &= f_2 &&= 1 \\ f_n &= f_{n-1} + f_{n-2} \end{aligned}$$

Nachfolgend ist die aus der Vorlesung bekannte iterative Implementierung nach dem Prinzip der dynamischen Programmierung angegeben als Funktion **fib**(n).

fib(n) :
    $fib =$ leeres Feld;
    $fib[1] = 1$; $fib[2] = 1$; $k = 3$;
    while $(k \leq n)$ {
        $fib[k] = fib[k-1] + fib[k-2]$;
        $k = k + 1$;
    }
    return $fib[n]$;

a) Geben Sie jeweils eine geeignete Vor- und Nachbedingung der Funktion **fib**(n) an!

b) Geben Sie eine geeignete Schleifeninvariante P für die **while** $(k \leq n)$ { ... } Schleife an!

c) Welche drei Bedingungen müssen überprüft werden, damit die Korrektheit der **while** Schleife mittels der Invariante P gezeigt werden kann? Zeigen Sie, dass diese drei Bedingungen

in diesem Fall erfüllt sind!

Aufgabe 7 Sortieren **(9 Punkte)**

a) Nach welchem Algorithmen-Muster funktioniert Merge Sort?

b) Welche der vier bekannten Sortieralgorithmen (Insertion Sort, Selection Sort, Merge Sort, Quick Sort) sortieren 'in place' und welche nicht? Was bedeutet 'in place'?

c) Gegeben sei eine bereits sortierte Liste von natürlichen Zahlen der Länge n. In diese Liste soll nun ein weiteres Element einsortiert werden. Schlagen Sie einen möglichst effizienten Algorithmus für diese Aufgabe vor und geben Sie dessen Komplexität an.

d) Skizzieren Sie (durch Angabe des Zustandes nach jedem Durchlauf der Hauptschleife), wie *Quick Sort* (in-place) die folgende Zahlenreihe sortieren würde! (Hinweis: Als Pivotelement soll jeweils das erste Element von links gewählt werden.)

121 , 7 , 99 , 1 , 2678 , 37 , 10 , 2

Aufgabe 8 Komplexität **(7 Punkte)**

Sehen Sie sich das folgende Code-Beispiel an:

Input: Feld A der Länge n
Algorithm(A):

```
1       do {
2           swapped = false;
3           for i = 1 to n − 1 {
4               if (A[i − 1] > A[i]) {
5                   swap(A,i − 1,i);
6                   swapped = true;
7               }
8           }
9       } while (swapped)
```

Input: Feld A der Länge n, zwei Indizes i, j
swap(A,i,j):

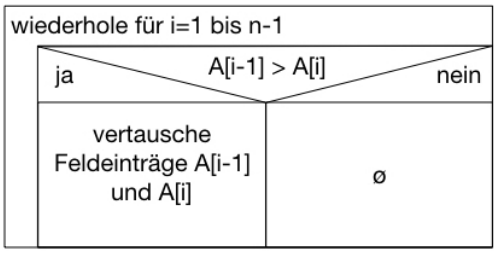
```
k = A[j];
A[j] = A[i];
A[i] = k;
```

a) Was macht dieser Algorithmus? Bitte halten Sie Ihre Beschreibung so kurz wie möglich. (Hinweis: eine Wiedergabe der Funktion einzelner Zeilen ist nicht gefragt!)

b) Welches Entwurfsprinzip liegt diesem Code-Beispiel zugrunde?

c) Schätzen Sie das Wachstumsverhalten des Algorithmus ab. Begründen Sie Ihre Wahl hinreichend z.B. mittels entsprechender Zeilenangaben. (Hinweis: Sie dürfen das RAM-Modell annehmen. Also hat ein einzelner elementarer Verarbeitungsschritt konstantes Wachstum.)

d) Betrachten Sie das folgende Struktogramm:



Ist diese Darstellung äquivalent zu dem obigen Code-Beispiel? Begründen Sie Ihre Entscheidung.