Prof. Dr. Ir. Joost-Pieter Katoen

Christian Dehnert, Jonathan Heinen, Thomas Ströder, Sabrina von Styp

Hinweise:

- Die Übungsblätter sind in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung zu bearbeiten.
- Die Lösungen müssen bis Montag, den 7. Mai um 11:00 Uhr in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55).
- Namen und Matrikelnummern der Studenten sowie die Nummer der Übungsgruppe sind auf jedes Blatt der Abgabe zu schreiben. Heften bzw. tackern Sie die Blätter!
- Für die Übungsgruppen 7,8,9 und 10 bieten wir in den nächsten Wochen Ersatztermine an. Eine Liste der Ausweichtermine befindet sich auf unserer Webseite.
- Am Freitag den 11. Mai, sowie am Dienstag den 15. findet keine Vorlesung statt.

Aufgabe 1 (Sortieralgorithmus I): (2 + 1 + 2 + 2 + 3 Punkte)

Gegeben sei der folgende Sortieralgorithmus:

```
void sort(int E[]) {
  int i,j,m;
  for (i = 0; i < E.length; i++) {
    m = i;
    for (j = i + 1; j < E.length; j++) {
      if (E[j] <= E[m]) {
          m = j;
      }
    }
    int v = E[i];
    E[i] = E[m];
    E[m] = v;
  }
}
```

a) Nutzen Sie den gegebenen Algorithmus, um das folgende Array zu sortieren. Geben Sie den Zustand des Arrays nach jedem Durchlauf der äußeren Schleife an.

1	3	2	7	0	4	8	5	7	6

b) Geben Sie in wenigen Worten wieder, wie der gegebene Algorithmus funktioniert. In der Vorlesung wurden mehrere Sortierverfahren genannt (siehe Folien). Welcher Name passt zu diesem Algorithmus?

c) Ist der Sortieralgorithmus stabil? Falls nicht geben Sie an wie er angepasst werden muss, damit er stabil wird?

d) Welche Average-Case Laufzeit besitzt der gegebene Sortieralgorithmus für eine Eingabe der Länge n? Geben Sie die Komplexitätsklasse $\Theta(T_{sort}(n))$ für ein Array E mit Länge `n = E.length` an und begründen Sie Ihre Antwort.

e) Implementieren Sie eine Version des obigen Sortieralgorithmus, der die Elemente einer doppelt verketten Liste aufsteigend nach ihren Schlüsseln sortiert. Die Signatur der Sortiermethode sei `void sort(List l)`. Die Klassendefinition eines Listenelements ist unten gegeben. Gehen Sie davon aus, dass der Algorithmus das erste Element einer nicht zyklischen, doppelt verketteten Liste erhält, d.h. der `prev`-Zeiger des ersten Elements ist ein Nullpointer, ebenso der `next`-Zeiger des letzten Elements. Achten Sie darauf tatsächlich die Reihenfolge der Elemente zu ändern und nicht lediglich die Schlüsselwerte zu vertauschen.

```
class List{
    int key;
    List next, prev;
}
```

Aufgabe 2 (Sortieralgorithmus II): (3 + 3 + 3 Punkte)

Betrachten Sie den folgenden Sortieralgorithmus.

```
static void sort(int E[]) {
  int sorted = 0;
  while(sorted < E.length - 1) {
    int pos = 0;
    for(int i = sorted + 1; i < E.length; i++){
      if(E[i] <= E[sorted]){
        pos++;
      }
    }
    if (pos == 0){
      sorted++;
    }else{
      swap(E, sorted, sorted + pos);
    }
  }
}

static void swap(int E[], int i, int j){
  int tmp = E[i];
  E[i] = E[j];
  E[j] = tmp;
}
```

a) Nutzen Sie den gegebenen Algorithmus, um das folgende Array zu sortieren. Geben Sie den Zustand des Arrays nach jedem Durchlauf der `while`-Schleife an.

4	1	7	9	0	6	2	3	5	8

b) Geben Sie die Best-Case sowie Worst-Case Laufzeit des gegebenen Sortieralgorithmus an. Geben Sie hierzu jeweils die Komplexitätsklasse $\Theta(T_{sort}(n))$ für ein Array E mit Länge `n = E.length` an und begründen Sie Ihre Antwort.

c) Ergibt sich eine bessere Worst-Case Laufzeit wenn der Vergleich `E[i] <= E[sorted]` durch den Vergleich `E[i] < E[sorted]` ersetzt wird?

Aufgabe 3 (Rekursionsgleichungen): (2 + 2 + 4 + 3 Punkte)

Geben Sie für die folgenden Rekursionsgleichungen die Komplexitätsklasse (Θ) an. Begründen Sie Ihre Antwort.

a) $T(n) = 23T(\frac{n}{23}) + 42n$

b) $T(n) = 9T(\frac{n}{3}) + 27n$

c) $T(n) = 2T(\frac{n}{2}) + n\log_2 n$

d) $T(n) = 2T(\frac{n}{2}) + T(\frac{n}{3}) + 2n^2 + 5n + 42$