Prof. Dr. Ir. Joost-Pieter Katoen

Jonathan Heinen, Sabrina von Styp

Hinweise:

- Die Übungsblätter sollen in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung bearbeitet werden.
- Die Lösungen müssen bis Montag, den 17. 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!
- Am Freitag den 14.05.2010 findet keine Vorlesung statt.

Aufgabe 1 (Verbesserter Insertionsort): (4 + 5 Punkte)

Der folgende Algorithmus stellt eine Variante von Insertionsort dar, bei der die lineare Suche nach der Einfügeposition durch eine binären Suche ersetzt wurde.

```
int binaereSuche (int E[], int key, int left, int right){

    if (left == right)
        return left;

    int mid = left + ((right - left) / 2);

    if (key > E[mid])
        return binaereSuche (E, key, mid + 1, right);
    else if (key < E[mid])
        return binaereSuche (E, key, left, mid);

    return mid;
}


void sort(int[] E){

    int ins, i, j;
    int tmp;

    for (i = 1; i < E.length; i++) {

        ins = binaereSuche (E, E[i], 0, i);
        tmp = E[i];

        for (j = i - 1; j >= ins; j--)
            E[j + 1] = E[j];

        E[ins] = tmp;
    }
}
```

a) Bestimmen Sie jeweils $\Theta(T_{binaereSuche(E,k,0,n)})$ in Abhängigkeit von `n` im Worst- und Best-case.

b) Nutzen sie das Ergebnis Ihrer Analyse aus **a)** um $\Theta(T_{sort(E)})$ in Abhängigkeit von der Größe des übergebenen Arrays `E.length = n` für den Worst- sowie Best-case zu bestimmen.

Aufgabe 2 (Sortieralgorithmus): (5 + 3 Punkte)

```
sort(int [ ] A, int l, int r) {
  if (A[l] > A[r]) then {
    exchange(A[l], A[r]);
  }
  if (l < r-1) then {
    k:= (r-l+1) div 3;
    sort(A, l, r-k);
    sort(A, l+k, r);
    sort(A, l, r-k);
  }
}
```

a) Bestimmen Sie vom Suchalgorithmus die Komplexitätsklasse Θ im Best-, Worst- und Average-Case für den Aufruf `sort(A,l,r)` in Abhängigkeit von `n`, der Anzahl der Elemente im Array `A`.

b) Der vorgestellte Algorithmus ist nicht stabil. Ändern Sie den Algorithmus, so ab dass er stabil ist.

Aufgabe 3 (Rekursionsbäume): (5 + 6 Punkte)

Erstellen Sie zu den folgenden Rekursionsgleichungen die entsprechenden Rekursionsbäume und lesen Sie eine möglichst kleine obere Schranke der Lösung ab. Überprüfen Sie die Schranke mit Hilfe des Substitutionsverfahrens.

a)

$$T(n) = 4 \cdot T(\frac{n}{3}) + \frac{n}{2} + 3 \text{ mit } T(0) = 1$$

b)

$$T(n) = T(n-2) + T(n-1) + 7 \text{ mit } T(0) = 1 \text{ und } T(1) = 7$$

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

Geben Sie eine möglichst gute untere und obere Schranke für nachfolgende Rekursionsgleichungen an. Verwenden Sie dafür das Mastertheorem und begründen Sie Ihre Antwort.

a) $T(n) = 2T(\sqrt{n}) + \mathrm{ld}n$

b) $T(n) = 4T(\frac{n}{5}) + n \cdot \log n$

c) $T(n) = 8T(\lceil \frac{n}{2} \rceil) + 16n$

d) $T(n) = 47T(\lfloor \frac{n}{47} \rfloor) + 47n$