3.1 Binärer Suchbaum (2+3+1+1+1 Punkte)

a) Lesen Sie sich den Beispielcode `Quicksort.java` von der Homepage durch. In der `main`-Methode des Beispiels steht, dass alle Permutationen die gleiche Wahrscheinlichkeit haben. Begründen Sie dies.

b) Schreiben Sie eine Klasse `Node`, die einen Knoten eines binären Suchbaums repräsentiert. Die Klasse soll eine Methode
`Node insert(int k)`
besitzen, die eine ganze Zahl `k` naiv in den Baum einsortiert und den eingefügten Knoten zurück gibt.[1] Schreiben Sie weiterhin eine Methode `int maxDepth()`, die die maximale Tiefe des Baums ermittelt und zurück gibt.

c) Schreiben Sie eine `main`-Methode, die zwei Zahlen $n, m \in \mathbb{N}$ von der Tastatur einliest, alle ganzen Zahlen von 0 bis $(n-1)$ erzeugt und wie im Quicksort-Beispiel permutiert. Führen Sie dieses Experiment m mal durch und ermitteln Sie die durchschnittliche maximale Tiefe. Geben Sie diese auf die Konsole aus.

d) Modifizieren Sie die Erzeugung der Zufallszahl zur Permutation dabei wie folgt: Ersetzen Sie den Aufruf von `Math.random()` durch `random.nextDouble()`, wobei die Variable `random` zu Beginn des Programms einmalig mit
`Random random = new Random(0);`
initialisiert werden soll.[2]

e) Falls $m = 1$, durchlaufen Sie den Baum in In-Order Reihenfolge und geben Sie die Zahlen aus.[3]

3.2 Selektionsproblem, Teil 1 (5 Punkte)

In der Vorlesung wurde im Beweis des Selektionsproblems folgende Abschätzung von (1) nach (2) gemacht:

$$T(n) \leq cn + \frac{a}{n} \sum_{i=1}^{n} \max(i-1, n-i) \tag{1}$$

$$\leq cn + \frac{2a}{n} \sum_{j=\frac{n}{2}}^{n-1} j \tag{2}$$

$$\leq cn + \frac{2a}{n} \sum_{j=1}^{n-1} j \tag{3}$$

Womit im Weiteren bewiesen wurde, dass das Problem in $O(n)$ lösbar ist. Die Abschätzung selbst wäre auch korrekt gewesen, wenn man die Summe bei $j = 1$ hätte starten lassen (3). Führen Sie die Rechnungen aus der Vorlesung mit eben dieser Abschätzung weiter und er-

[1] Duplikate treten in dieser Aufgabe nicht auf.
[2] Hierdurch wird sicher gestellt, dass immer wieder die gleichen "zufälligen" Zahlen erzeugt werden und das Ergebnis somit vergleichbar ist.
[3] geben Sie hierbei zuerst die Elemente des Baums aus und erst danach das Ergebnis aus c)

läutern Sie, weshalb der Beweis dann nicht funktioniert.

3.3 Selektionsproblem, Teil 2 (4+3 Punkte)

In der Vorlesung wurde ein deterministischer Algorithmus für das Selektionsproblem vorgestellt, der zu Beginn alle Zahlen in 5er-Gruppen aufteilte und auf eine Laufzeit in $\Theta(n)$ kam. Berechnen Sie analog die Laufzeit für einen Algorithmus, der

a) 3er-Gruppen verwendet
b) 7er-Gruppen verwendet