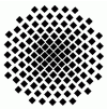


Universität Stuttgart
Fakultät 5, Institut FMI
Einführung in die Informatik II

Sommersemester 2006
Claus, Draskoczy, Schmid
2. Testklausur am 10.07.2006

Name		Matrikel-nummer		Übungs-gruppe	

Lesen Sie alle Aufgaben sorgfältig durch! Die maximal erreichbare Punktzahl ist 31, davon werden aber höchstens 30 Punkte angerechnet.

1 Wissensfragen ($6 \times 0.5 = 3$ Punkte)

Welche der folgenden Aussagen ist richtig, welche falsch? (Bitte zutreffendes ankreuzen)

a) In einer Hashtabelle sollten aus Effizienzgründen mindestens 80% der Plätze leer bleiben. richtig ◯ falsch ◯

b) Falls beim offenen Hashing eine Kollision auftritt, wird keine neue Adresse ermittelt. richtig ◯ falsch ◯

c) Da ein Heap ein Binärbaum ist, sind Arrays nicht als Datenstruktur dafür geeignet. richtig ◯ falsch ◯

d) Die Worst-Case Zeitkomplexität von Quicksort ist $O(n^2)$. richtig ◯ falsch ◯

e) Ein B-Baum der Ordnung $m = 4$ und der Tiefe $t = 2$ enthält mindestens 9 Elemente. richtig ◯ falsch ◯

f) In B-Bäumen kommt *Borgen* vor *Verschmelzen* weil die Wahrscheinlichkeit für *Borgen* größer ist. richtig ◯ falsch ◯

2 Bubblesort (2+1+1 Punkte)

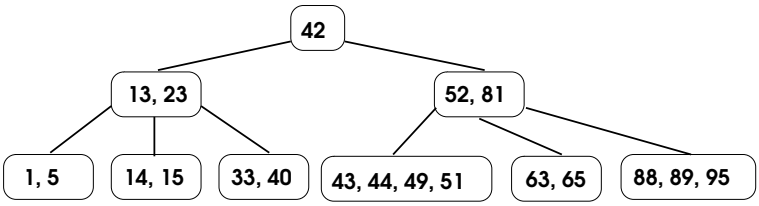
Ergänzen Sie die Prozedur `Bubblesort` (die Prozedur implementiert das Sortierverfahren Bubblesort). Die Variable `A` ist ein Array vom Typ `Integer_Array`.

```
procedure Bubblesort(A: in out Integer_Array) is
Weiter : Boolean [:=          ]; H : Integer;
begin while Weiter loop
        Weiter [:=          ];
        for i in A'First..A'Last-1 loop
          if A(i) [>          ] then Weiter:= True;
            H:= A(i); A(i) [:=          ]; A(i+1):= H;
          end if;
        end loop; end loop;
end Bubblesort;
```

Ist diese Variante von Bubblesort ordnungsverträglich? Ja ◯ Nein ◯

Die Zeitkomplexität in O-Notation lautet: $O(\quad)$

3 B-Baum (3+3 Punkte)



Fügen Sie in den obigen B-Baum der Ordnung $m = 2$ den Schlüssel 45 ein und zeichnen Sie den B-Baum in diesen Kasten ein:

Löschen Sie aus dem Ausgangsbaum (ganz oben) den Schlüssel 13 und zeichnen Sie den B-Baum in diesen Kasten ein:

4 Hashing (7 Punkte)

Fügen Sie die Schlüssel D, E, F, G, H und I in dieser Reihenfolge in die Hashtabelle (mittlere Tabelle ein). Verwenden Sie für die Kollisionsbehandlung *quadratisches Sondieren*. Die Hashwerte der Schlüssel entnehmen Sie der linken Tabelle. Tragen Sie zusätzlich für jeden der einzufügenden Schlüssel und für jede Kollision ein P für Primärkollision und ein S Sekundärkollision in der rechten Tabelle ein!

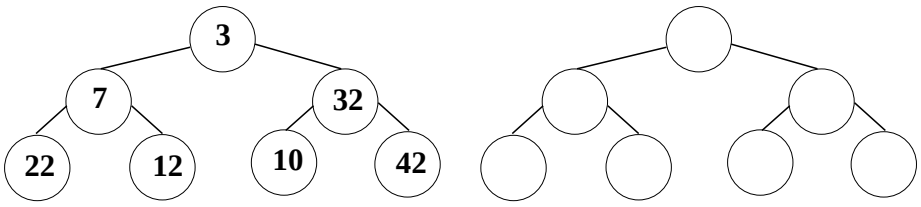
Schlüssel	Hashwert
A	10
B	10
C	10
D	4
E	4
F	3
G	0
H	1
I	1

0	B
1	
2	
3	C
4	
5	
6	
7	
8	
9	
10	A

Schlüssel	1. Koll.	2. Koll.	3. Koll.
D			
E			
F			
G			
H			
I			

5 Heapsort (2+1+1+2 Punkte)

Wandeln Sie den linken Binärbaum zu einem Heap um und tragen Sie das Ergebnis im rechten Binärbaum ein. Sie sollen also den ersten Teil des Heapsort-Algorithmus durchführen, der die absteigende Heap-Eigenschaft im Binärbaum herstellt.



Wie oft wird im obigen Beispiel die `Sink`-Prozedur aufgerufen? Anzahl:

Geben Sie das zu dem obigen Binärbaum korrespondierende Array an (nach dem Herstellen der Heap-Eigenschaft):

1	2	3	4	5	6	7

Beschreiben Sie die Idee des zweiten Teils des Heapsort-Algorithmus. Wie wird aus einem Baum, der die Heap-Eigenschaft erfüllt, eine sortierte Folge?

6 Minimumsuche (3+1+1 Punkte)

Führen Sie im unten stehenden Array die Minimumsuche durch. Tragen Sie das Array nach jedem Sortierschritt in die Tabelle ein.

3	13	42	12	23	1

Die Zeitkomplexität der Minimumsuche in O-Notation lautet: $O(\quad)$

Der zusätzliche Platzbedarf der Minimumsuche in O-Notation ist: $O(\quad)$