Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

Klausur
Algorithmen und Datenstrukturen
WS 2021/22 – 07.02.2022

Hinweise:

- Die Bearbeitungszeit beträgt **90 Minuten!**
- Zum Bestehen der Klausur sind 50 Punkte erforderlich.
- Erlaubte Hilfsmittel: keine
- Notieren Sie Ihre Lösungen auf eigenem Papier. Versehen Sie jedes Blatt mit Namen, Matrikelnummer, Studiengang und Seitenzahl. **Geben Sie bei jeder Lösung die Aufgabennummer und Teilaufgabe an.** Bitte beginnen Sie für jede Aufgabe ein neues Blatt, d.h. auf einem Blatt sollten nicht Lösungen zu mehreren Aufgaben stehen.
- Bitte schreiben Sie deutlich mit schwarzem Stift.
- Alle vorgegebenen und zu erstellenden Programmtexte beziehen sich auf die Programmiersprache Java.
- Scannen Sie am Ende der Klausur Ihre Lösungsblätter ein. Auf dem ersten Blatt scannen Sie zusätzlich die Vorderseite, auf dem letzten Blatt die Rückseite Ihres Studierendenausweises mit ein.

Viel Erfolg!

Aufgabe	**Maximalpunkte**	**Erreichte Punkte**
1 (Wissen)	16	
2 (Komplexität, Rekursion)	14	
3 (Listen)	20	
4 (B-Bäume)	25	
5 (Sortieren)	12	
6 (Graphen)	18	
Summe	105	

Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

Aufgabe 1 (Wissen) — *16 Punkte*

a) Notieren Sie zu jeder Aussage, ob sie richtig oder falsch ist.

1. $2^n = O(n^2)$
2. $\log(n) = O(n)$
3. $n^3 + 3n^2 + 3n + 1 = O(n^2)$

b) Ergänzen Sie in der Tabelle die Zeitkomplexität bei n enthaltenen Objekten in O-Notation. Die ArrayList habe noch mindestens ein freies Element.

	ArrayList	LinkedList
Einfügen in der Mitte		
Zugriff auf das i-te Element		

c) Wie lautet die Suchbaumeigenschaft für einen Binärbaum?

d) Wann heißt ein Sortierverfahren stabil? Nennen Sie ein Beispiel für ein stabiles Sortierverfahren.

e) Sie haben ein Array mit unsortierten Werten gegeben. Welches elementare Suchverfahren setzen Sie ein, um festzustellen, ob ein Wert im Array vorhanden ist (mit Begründung).

f) Sie haben ein Array mit sortierten Werten gegeben. Welches elementare Suchverfahren setzen Sie ein, um auch im schlechtesten Fall effizient festzustellen, ob ein Wert im Array vorhanden ist. Welche Zeitkomplexität in O-Notation besitzt dieses Verfahren?

NOTIEREN SIE DIE LÖSUNGEN AUF IHREM LÖSUNGSBLATT!

Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

Aufgabe 2 (Komplexität, Rekursion) *14 Punkte*

Bestimmen Sie für die nachstehenden Prozeduren folgende Informationen:

```
static void proz1(int n)
{
    for (int a=0; a<=n; a++)
        for (int b=1; b<n; b++)
            tuwas();

}
```

```
static void proz2(int n)
{
    tuwas();

    if (n > 0)
    {
       proz2(n-1);
    }

    tuwas();
}
```

- Berechnen Sie die Anzahl der Aufrufe von tuwas() für n=4.
- Bestimmen Sie die Anzahl der Aufrufe von tuwas() als Funktion von n.
- Bestimmen Sie die asymptotische Zeitkomplexität in O-Notation unter der Annahme, dass die asymptotische Zeitkomplexität von tuwas() O(1) ist.

a) Notieren Sie die Lösung als Tabelle auf Ihrem Lösungsblatt.

Methode	**Anzahl Aufrufe für n=4**	**Anzahl Aufrufe als Funktion von n**	**O-Notation**
`proz1`			
`proz2`			

b) Bestimmen Sie die Rekursionstiefe von proz2 für `n=4`.

c) Bestimmen Sie die Rekursionstiefe von proz2 in Abhängigkeit von `n`.

NOTIEREN SIE DIE LÖSUNG AUF IHREM LÖSUNGSBLATT!

Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

Aufgabe 3 (Listen) *20 Punkte*

Betrachten Sie die folgende teilweise Implementierung einer einfach verketteten Liste:

```
public class Link
{
    public String daten;
    public Link naechster;

    public Link(String daten, Link naechster){
        this.daten = daten;
        this.naechster = naechster;
    }
}

public class Liste
{
   private Link anfang, ende;

    public Liste(){
       anfang = ende = null;  // Leere Liste
    }
    …
    public void verschieben(){
       // Aufgabenteil a)
    }

    public boolean istDoppelt(String s){
        assert(s!=null);
       // Aufgabenteil b)
    }
}
```

a) Vervollständigen Sie die Methode `verschieben`, die das erste Element der Liste an das Ende der Liste verschiebt (umhängt). Wenn die Liste höchstens 1 Element enthält, soll nichts passieren.

```
public void verschieben()
{
   // ToDo !!!
}
```

b) Vervollständigen Sie die Methode `istDoppelt`, welche überprüft, ob die Zeichenkette `s` genau zweimal in der Liste enthalten ist. **Hinweis**: Sie können davon ausgehen, dass der Parameter `s` ungleich null ist.

```
public boolean istDoppelt(String s)
{
   assert(s!=null);
   // ToDo !!!
}
```

NOTIEREN SIE DIE LÖSUNG AUF IHREM LÖSUNGSBLATT!

Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

Aufgabe 4 (B-Bäume) *25 Punkte*

Betrachten Sie die folgende teilweise Implementierung eines B-Baumes:

```
public class BKnoten
{
   public int[] schluessel;
   public BKnoten[] kinder;

   // Für Blätter
   public BKnoten(final int[] schluessel)
   {
      assert(schluessel!=null);

      this.schluessel = schluessel;
      this.kinder = new BKnoten[schluessel.length + 1];
   }

   // Für innere Knoten
   public BKnoten(final int[] schluessel, final BKnoten[] kinder)
   {
      assert(schluessel != null);
      assert(kinder != null);
      assert(schluessel.length+1 == kinder.length);

      this.schluessel = schluessel;
      this.kinder = kinder;
   }
}

public class BBaum
{
   private BKnoten wurzel;

   public boolean hoeherAls(int n) {
      // Aufgabenteil a)
   }

   public boolean istBinaer() {
      return istBinaer(wurzel);
   }

   private boolean istBinaer(BKnoten knoten) {
      // Aufgabenteil b)
   }
}
```

Wichtiger Hinweis:
Das Array `schluessel` ist für jeden Knoten vollständig mit Schlüsseln gefüllt, kann aber für verschiedene Knoten eine unterschiedliche Länge besitzen!

Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

a) Vervollständigen Sie die Methode `hoeherAls`, die iterativ überprüft, ob die Höhe des B-Baumes größer als `n` ist.

```
public boolean hoeherAls(int n)
{
  // ToDo !!!
}
```

b) Vervollständigen Sie die Methode `istBinaer`, die rekursiv überprüft, ob alle Knoten im Teilbaum mit der Wurzel `knoten` maximal zwei Kinder besitzen.

```
private boolean istBinaer(BKnoten knoten)
{
  // ToDo !!!
}
```

NOTIEREN SIE DIE LÖSUNG AUF IHREM LÖSUNGSBLATT!

Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

Aufgabe 5 (Sortieren) *12 Punkte*

Sortieren Sie das folgende Feld nach dem Heapsort-Verfahren aufsteigend. Markieren Sie vor jedem Versickerungsvorgang das zu versickernde Element durch Einkreisen. Schreiben Sie nach jedem Versickerungsdurchlauf die neue Reihenfolge auf. Markieren Sie die Zeile, in der der erste Teil des Verfahrens, d.h. das Erstellen des Heaps, beendet ist. Trennen Sie danach bereits sortierte Teilfelder durch einen senkrechten Strich voneinander.

3	8	4	9	2	7
⋮	⋮	⋮	⋮	⋮	⋮

NOTIEREN SIE DIE LÖSUNG AUF IHREM LÖSUNGSBLATT!

Name, Vorname, Matrikelnummer	Studiengang	Sitzplatznummer

Aufgabe 6 (Graphen) — *18 Punkte*

Folgender Graph ist durch seine Adjazenzmatrix gegeben:

		Nach				
	A	**B**	**C**	**D**	**E**	**F**
Von **A**		1	1			
B	1			1	1	
C	1					
D		1				1
E		1				
F				1		

a) Begründen Sie anhand der Adjazenzmatrix, ob der Graph gerichtet oder ungerichtet ist.

b) Zeichnen Sie den gegebenen Graphen.

c) Führen Sie für den Graphen eine Tiefensuche beginnend ab A durch. Notieren Sie die Reihenfolge in der die Knoten besucht werden.

d) Führen Sie für den Graphen eine Breitensuche beginnend ab A durch und notieren Sie für jeden Schritt den Zustand der Warteschlange.

NOTIEREN SIE DIE LÖSUNG AUF IHREM LÖSUNGSBLATT!