

## Klausur Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

### Hinweise

- Bevor Sie mit der Bearbeitung der Aufgaben beginnen, müssen Sie auf allen Blättern Ihren Namen und Ihre Matrikelnummer eintragen. Dafür bekommen Sie zusätzlich fünf Minuten Zeit.
- Der Klausurtext enthält ausreichend Platz zur Lösung der Aufgaben. Sie können auch die Rückseiten der Blätter für Ihre Lösungen nutzen. Sofern Sie zusätzliches Papier benötigen, wenden Sie sich an die Aufsicht. Die Nutzung eigenen Papiers ist nicht gestattet.
- Sollte Ihre Lösung nicht unmittelbar unter oder neben der Aufgabenstellung stehen, machen Sie bitte einen entsprechenden Hinweis. Streichen Sie diejenigen Teile der von Ihnen geschriebenen Texte deutlich durch, die **nicht** in die Bewertung eingehen sollen.
- Die Klausur ist zusammengeheftet. Die Heftung darf nicht geöffnet werden.
- Die Aufsicht gibt Ihnen keine Hilfestellung beim Lösen der Aufgaben.
- Teilnehmer, die eine Einzelklausur über **ein Teilgebiet** schreiben, müssen zum Bestehen **mindestens 20 Punkte** erreichen. Maximal können 50 Punkte erreicht werden.
- Teilnehmer, die die Gesamtklausur über **beide Teilgebiete** schreiben, müssen zum Bestehen in beiden Teilgebieten **jeweils mindestens 20 Punkte** erreichen. Maximal können jeweils 50 Punkte erreicht werden.

*Viel Erfolg!*

	a)	b)	c)	d)	e)	f)	insg.	
1							10	
2	2	1	1	2	2	2	10	
3	1	4	4	1	2		12	
4							5	
5	2	1	6	4			13	
<b>Summe</b>							<b>50</b>	

# Klausur - Teilgebiet: Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

## Aufgabe 1: Multiple Choice-Test

### Hinweise:

Zu jeder Frage gibt es **genau eine** richtige Antwort.

Bewertung je Frage:

Richtiges Kreuz: 1 Punkt

Falsches Kreuz: 0,5 Punkte Abzug

Mehr als ein Kreuz: 0 Punkte

Maximal zu erreichende Gesamtpunktzahl: 10 Punkte

Minimal zu erreichende Gesamtpunktzahl: 0 Punkte

Sollten Sie Ihre Antwort korrigieren, machen Sie bitte **unmissverständlich** deutlich, welche Antwort gilt, andernfalls wird diese Frage nicht bewertet.

- Welche der folgenden Aussagen gilt **nicht** für jeden Baum, der ein Heap ist?
  - Der Knoten mit der größten Beschriftung im Baum ist ein Blatt.
  - Für beliebige Knoten a und b des Baumes gilt: Ist die Beschriftung von a kleiner als die Beschriftung von b, so muss b ein Sohn von a sein.
  - Die Heap-Bedingung ist in allen Teilbäumen einer Tiefe  $\leq 2$  erfüllt
- In einem schwach zusammenhängenden Graphen mit mindestens zwei Knoten
  - gibt es zwischen zwei beliebigen Knoten a und b immer eine Kante von a nach b oder von b nach a
  - gibt es zwischen zwei Knoten a und b immer einen Semiweg
  - gibt es zwischen zwei Knoten a und b immer einen Pfad von a nach b oder einen Pfad von b nach a
- Sei S ein nichtleerer binärer Suchbaum. Welche Aussage über S ist wahr?
  - Die Beschriftung der Wurzel ist größer als die Beschriftung jedes anderen Knotens.
  - Der Knoten mit der kleinsten Beschriftung muss ein Blatt sein.
  - Jeder Unterbaum von S ist wieder ein binärer Suchbaum.
- Sei B ein Binärbaum, t die Tiefe von B, k die Zahl der Kanten von B, n die Zahl der Knoten von B und b die Zahl der Blätter von B. Welche der folgenden Aussagen trifft immer zu?
  - $n = 2^t - 1$
  - $n = k + 1$
  - $b = 2^{t-1}$

# Klausur - Teilgebiet: Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

- Wenn eine Java-Klasse ein Interface implementiert, aber nicht alle Methoden des Interfaces überschreibt
  - muss sie ein weiteres Interface implementieren, in dem die nicht überschriebenen Methoden implementiert sind
  - wird beim Aufruf der Methode die im Interface enthaltene Implementierung genutzt
  - muss sie als abstrakt deklariert werden
  
- Welche der folgenden Aussagen ist für die Sprache Java **falsch**?
  - Eine Klasse kann von mehreren anderen Klassen erben.
  - Ein Interface kann von mehreren anderen Interfaces erben.
  - Eine Klasse kann mehrere Interfaces implementieren.
  
- Sei G ein zusammenhängender gerichteter Graph. Welche der folgenden Aussagen ist **falsch**?
  - Enthält G keine Zyklen, ist G bereits selber Spannbaum.
  - Zu G gibt es genau einen Spannbaum.
  - Jede Kante, die in einem Spannbaum von G enthalten ist, ist auch in G enthalten.
  
- Von abstrakten Klassen
  - werden keine Instanzen erzeugt
  - können keine anderen Klassen erben
  - dürfen nur Klassen erben, die ausschließlich abstrakte Methoden definieren
  
- Für welchen Datentyp ist eine Operation Inhalt: Integer → Element definiert, die das Element an der durch den Integer definierten Position liefert?
  - Stack
  - Liste
  - Queue
  
- Eine Java-Klasse `Loslauf` sei definiert als `class Loslauf extends Thread {...}`  
Wie wird erreicht, dass ein neuer Thread dieser Klasse gestartet wird?
  - `Loslauf a = new Thread(); a.run();`
  - `Loslauf a = new Loslauf(); a.run();`
  - `Loslauf a = new Loslauf(); a.start();`

**Klausur - Teilgebiet: Programmierung und  
Programmiersprachen**

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

**Aufgabe 2: Algorithmen auf Graphen**

a) Gegeben sei ein Graph  $G=(V,E)$ . Definieren Sie formal die transitive Hülle von  $G$ . **(2 Punkte)**

b) Wie heißt ein Ihnen bekannter Algorithmus, der zu einem gegebenen Graphen dessen transitive Hülle berechnet? **(1 Punkt)**

c) Welche Laufzeitkomplexität (ausgedrückt in O-Schreibweise) hat der in b) gefragte Algorithmus? (Hinweis: Erklären Sie die Bedeutung der in Ihrer Lösung verwendeten Variablennamen!) **(1 Punkt)**

# Klausur - Teilgebiet: Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

d) Ein gerichteter Graph  $G$  sei durch die folgende Adjazenzmatrix gegeben:

	1	2	3	4	5
1	falsch	falsch	falsch	falsch	falsch
2	falsch	falsch	wahr	falsch	falsch
3	falsch	wahr	falsch	falsch	falsch
4	wahr	falsch	falsch	falsch	falsch
5	falsch	wahr	falsch	falsch	wahr

Zeichnen Sie den so definierten Graphen! **(2 Punkte)**

e) Benenne Sie alle schwachen Zusammenhangskomponenten des durch die Adjazenzmatrix in d) definierten Graphen. **(2 Punkte)**

f) Ändern Sie die Adjazenzmatrix in d) so ab, dass die entstehende Matrix die Adjazenzmatrix der transitiven Hülle von  $G$  ist. **(2 Punkte)**

	1	2	3	4	5
1					
2					
3					
4					
5					

# Klausur - Teilgebiet: Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

## Aufgabe 3: Datenstruktur

Gegeben seien die folgenden Codefragmente:

```
class Zelle{
    Object inhalt;
    Zelle verweis;
    Zelle(Object o) {inhalt = o;}
    Zelle(Object o, Zelle z) {inhalt = o; verweis = z;}
}

class A{
    private Zelle start;
    private Zelle aktuell;
    // Konstruktor und weitere Methoden weggelassen...
    // ...
    void EinfügenAnPositionP(int p, Object o) {
        aktuell = start;
        for (int i=1; i<p; i++) {aktuell=aktuell.verweis;}
        Zelle neu = new Zelle(o, aktuell.verweis);
        aktuell.verweis = neu;
    }
}
```

Bei den folgenden Aufgaben reicht es, die neu hinzukommenden Codeabschnitte zu notieren; der bereits gegebene Code muss nicht wiederholt werden.

## Klausur - Teilgebiet: Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

a) Welche Ihnen bekannte Datenstruktur wird durch die Klasse A implementiert? **(1 Punkt)**

b) Durch das folgende Interface ist eine Methode count() definiert, die die Zahl der in einer Datenstruktur enthaltenen Elemente liefern soll.

```
public interface Countable {  
    public int count();  
}
```

Notieren Sie für die Klasse A den Code, der nötig ist, um das Interface zu implementieren! **(4 Punkte)**

## Klausur - Teilgebiet: Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

c) Die Methode `EinfügenAnPositionP` der Klasse `A` enthält noch keine Prüfungen, ob die übergebenen Parameter sinnvoll sind. Ergänzen Sie die nötigen Prüfungen. Falls die Operation „Einfügen von `o` an Position `p`“ nicht möglich ist, soll eine `IllegalArgumentException` geworfen werden. **(4 Punkte)**

d) Begründen Sie, warum die gegebene Implementierung der Methode `EinfügenAnPositionP` der Klasse `A` nicht in mehreren parallel laufenden Threads verwendet werden sollte. **(1 Punkt)**

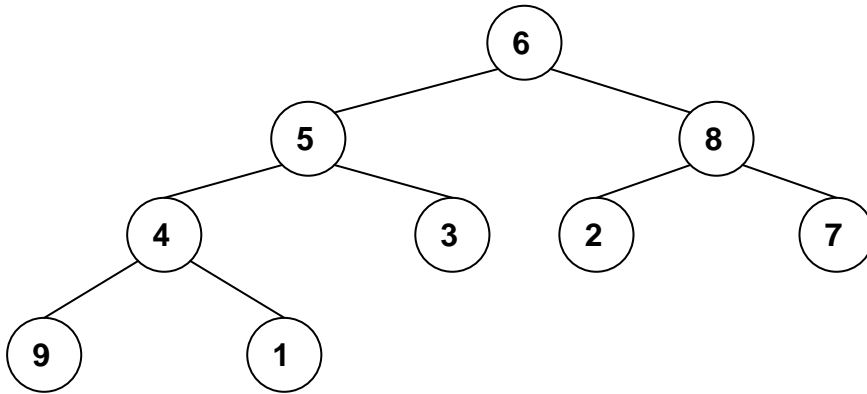
e) Ergänzen Sie in der Methode `EinfügenAnPositionP` den Code, der nötig ist, um die Ausführung dieser Methode in mehreren parallel laufenden Threads problemlos zu ermöglichen! **(2 Punkte)**

**Klausur - Teilgebiet: Programmierung und  
Programmiersprachen**

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

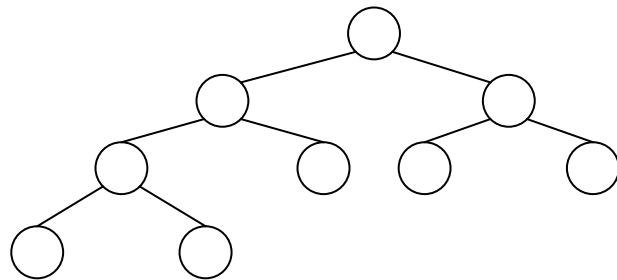
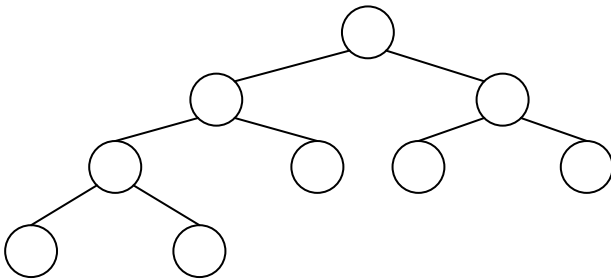
**Aufgabe 4:**

Benutzen Sie den in der Vorlesung behandelten Algorithmus, um den folgenden Binärbaum in einen Heap umzuformen. Notieren Sie die Ergebnisse der einzelnen Zwischenschritte in den unten gegebenen Diagrammen. **(5 Punkte)**



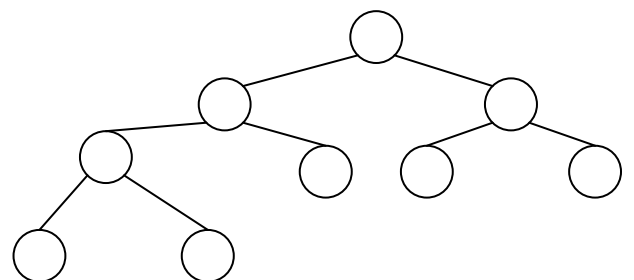
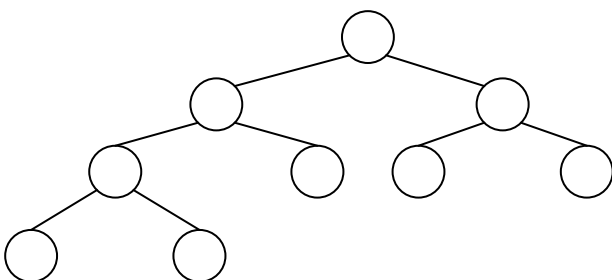
nach dem 1. Schritt:

nach dem 2. Schritt:



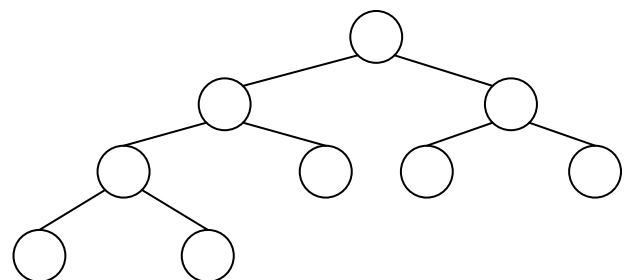
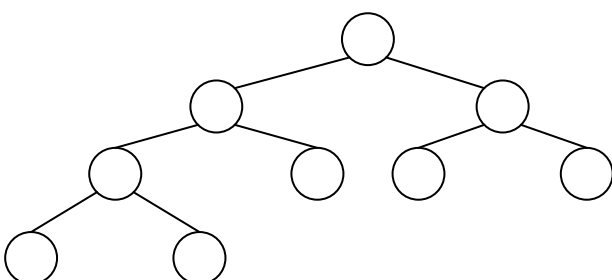
nach dem 3. Schritt:

nach dem 4. Schritt:



nach dem 5. Schritt:

nach dem 6. Schritt:



**Klausur - Teilgebiet: Programmierung und  
Programmiersprachen**

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

**Aufgabe 5: Vererbung und Interfaces**

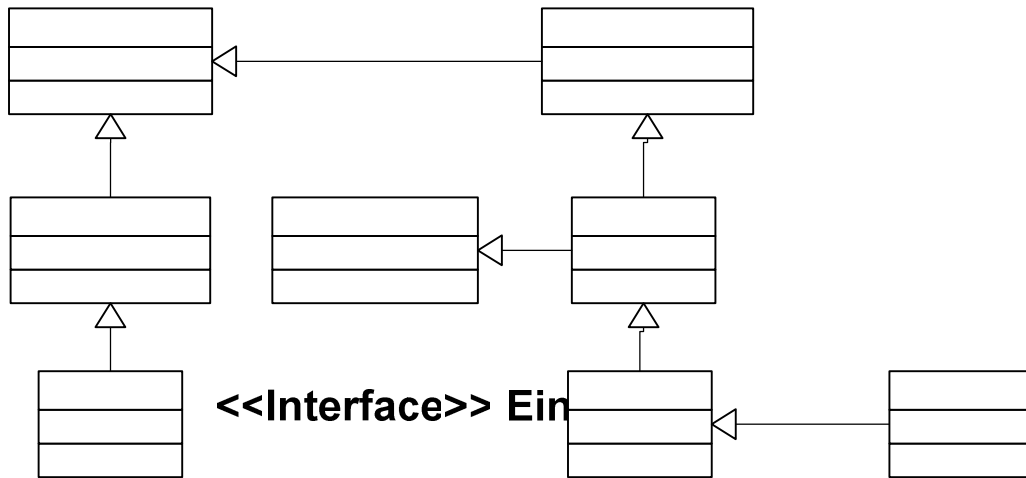
a) Erläutern Sie den Begriff Interface und grenzen Sie ihn vom Begriff der abstrakten Klasse ab. **(2 Punkte)**

b) Was verstehen Sie unter Überschreiben einer Methode? **(1 Punkt)**

# Klausur - Teilgebiet: Programmierung und Programmiersprachen

Vor- und Nachname: \_\_\_\_\_ Mat.-Nr.: \_\_\_\_\_

Gegeben seien die folgenden Klassen und Interfaces:



c) Notieren Sie die Coderaum-Coderahmen der Klassen Gamma und Delta. Ergänzen Sie dabei die Signaturen aller Methoden, die in der Klasse implementiert werden müssen. Nehmen Sie an, dass alle Methoden `public` sind und den Rückgabotyp `void` haben. **(6 Punkte)**

**<<Interface>> Drei**

**+BBB()**

**<<Interface>> Zwei**

**+DDD()**

d) Geben Sie für die folgenden Code-Codesequenzen an, ob sie korrekt oder falsch sind. Wenn Sie „falsch“notiert haben, geben Sie den Grund dafür an. **(4 Punkte)**

Alpha a = new Alpha(); a.CCC();

Alpha a = new Beta(); Beta b = (Beta) a;

Gamma g = new Gamma(); g.AAA(); g.BBB();

Beta b = new Beta(); Gamma g = new Gamma(); b = g;