Brandenburgische Technische Universität Cottbus

Institut für Informatik
Einführung in die Programmierung
Sommersemester 2012

Endklausur 25. September 2012

Name:	Vorname:
Matrikel:	Studiengang:
Raum: Audimax 2	Sitzreihe: Sitzplatz:

- Für Ihre Lösungen verwenden Sie bitte den freigelassenen Platz hinter der Aufgabenstellung.
- Bei Bedarf kann ein weiteres Blatt bei den Aufsichtspersonen angefordert werden.
- Nebenrechnungen machen Sie bitte auf Ihrem eigenen Papier, das aber nicht eingesammelt und deshalb nicht berücksichtigt wird.
- Verwenden Sie nur dokumentenechte Stifte, keine Bleistifte und keine rotschreibenden Stifte.
- Es sind keine Hilfsmittel (keine Taschenrechner, keine Handys, keine elektronischen Geräte usw.) zugelassen.
- Die Bearbeitungszeit beträgt **120 Minuten.**
- Anzahl der Aufgaben: **10**

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**∑**
Max. Punkte	**5**	**8**									**90**
Erreichte Punkte											

VIEL ERFOLG!

Aufgabe 1

a) Geben Sie alle Programmkonstrukte an, die in der Programmiersprache Java zur Verfügung stehen, um

- Verzweigungen,

- Wiederholungen

im Programmablauf zu programmieren.

Aufgabe 2

a) Für die Parameterübergabe an Unterprogramme gibt es verschiedene Techniken. Welche wird in der Programmiersprache Java verwendet?

b) Erklären Sie anhand der in der Vorlesung eingeführten grafischen Darstellung von Objekten wie die Parameterübergabe im folgenden Beispiel beim Aufruf der Methode `irgendWas` erfolgt.

```
class Auf2 {
  static void irgendWas(int a, int b, int[] c){

      //Methodenrumpf

  }

  public static void main(String[] argv) {
      int z = 2;

      int[] v = new int[2];

      irgendWas(5, z, v);  // Aufruf der Methode irgendWas
  }
} //class Auf2
```

Aufgabe 3

c) Schreiben Sie eine statische Methode `static boolean pruefe(int[][] a)`zum Test einer quadratischen Matrix (a_{ij}) auf Diagonalendominanz.
Eine quadratische Matrix heißt *diagonal dominant*, wenn in jeder Zeile der Betrag des Diagonal-elementes größer ist als die Summe der Beträge der übrigen Matrixelemente derselben Zeile, falls also gilt:

$$\forall i \in \{0,\ldots,n-1\}: \sum_{\substack{j=0 \\ j\neq i}}^{n-1} |a_{ij}| < |a_{ii}|$$

Für die Berechnung des Absolutbetrages können Sie die vordefinierte Methode `abs()` der Klasse `Math` verwenden: z. B. `Math.abs(-5)` liefert den Wert 5.

Aufgabe 4

Betrachten Sie das folgende Java-Programm:

```
import java.util.Scanner;
public class Auf4 {
   public static void main(String[] args){
    int i,b;
    String s;
    Scanner sc = new Scanner(System.in);
    while (true) {
       s = sc.next();
       if (s.equals("X")) break;      // beendet die while-Schleife
       b = sc.nextInt();
       i = Integer.parseInt(s,b);
       System.out.println(s + " base "+b+" = "+i);
    }
   }
} //class Auf4
```

Die verwendete Methode `parseInt` der Klasse `Integer` konvertiert den String `s`, der eine Zahl zur Basis `b` repräsentiert, in eine Dezimalzahl, falls der übergebene String eine gültige Zahl zur angegebenen Basis darstellt. Ist dies nicht der Fall, löst sie eine Ausnahme des Typs `NumberFormatException` aus. Ohne weitere Maßnahmen stürzt das Programm z. B. beim Versuch, den String "50" als Zahl zur Basis 5 anzusehen, ab.

Ändern Sie das Programm und fügen Sie die notwendigen Anweisungen ein, die solche Fehler abfangen und behandeln.

Aufgabe 5
Modifizierer können Sichtbarkeit und Eigenschaften von Klassen, Attributen und Methoden beeinflussen. Erläutern Sie kurz, welche Wirkung die Modifizierer

a) `abstract` bei der Definition einer Klasse,

b) `abstract` bei der Definition einer Methode,

c) `public` bei der Definition einer Klasse haben.

Aufgabe 6
Eine Java-Anwendung soll den Jahresurlaub der Mitarbeiter eines Unternehmens verwalten.
a) Schreiben Sie die Klasse `UrlaubsKonto`. Die Klasse hat

- die privaten Attribute:
  `mitarbeiter` (Name des Mitarbeiters)
  `urlaubsTage` (positive ganze Zahl ≥ 0)
  `vertreter` (Name des Vertreters)
  `genehmigt` (ja/nein)

- die private Klassenkonstante:
  `ANSPRUCH` mit dem Wert 30

- einen öffentlichen parametrisierten Konstruktor:
  `UrlaubsKonto (String name, int vorjahr):` Der Konstruktor initialisiert die Attribute. Der Parameter `name` enthält den Namen des Mitarbeiters, der Parameter `vorjahr` den Resturlaub aus dem Vorjahr und wird zum Initialisieren des Attributes `urlaubsTage` verwendet. Das Attribut `vertreter` soll mit der Zeichenkette "N.N.", das Attribut `genehmigt` mit `false` initialisiert werden.

- die öffentlichen Instanzmethoden:
  `boolean beantragen(int tage):` Sofern `tage` ≤ `urlaubsTage` werden die verbleibenden Urlaubstage neu berechnet und der Urlaub genehmigt. Andernfalls kann der Urlaubsantrag nicht genehmigt werden. Der Wert des Attributs `genehmigt` wird zurückgegeben.

  `int aktuellerStand():` Gibt die aktuelle Anzahl der Urlaubstage zurück.

  `void setzeVetreter(String vName):` Das Attribut `vertreter` wird auf den Wert `vName` gesetzt, wenn der Urlaub genehmigt wird.

```
public class UrlaubsKonto {

}//UrlaubsKonto
```

b) In einer ausführbaren Klasse sollen in der `main`-Methode folgende Anweisungen ausgeführt werden:

- Für Herrn Meier, Max, der noch 5 Urlaubstage aus dem Vorjahr hat, wird das Urlaubskonto angelegt.
- Für Frau Schulz, Jana, die keinen Resturlaub mehr aus dem Vorjahr hat, wird das Urlaubskonto angelegt.
- Herr Meier beantragt 15 Urlaubstage. Wird der Urlaub bewilligt, übernimmt Frau Schmidt, Renate die Urlaubsvertretung.
- Der Vorgesetzte von Frau Schulz, Jana möchte gern wissen, wie viele Tage Urlaub Frau Schulz, Jana noch hat.

```
public class UrlaubsPlanung {

  public static void main(String[] args) {









  }//main
}//UrlaubsPlanung
```

Aufgabe 7

a) Nennen Sie drei wesentliche Merkmale der objektorientierten Programmierung!

b) Gegeben sind die folgenden Klassen:

```
public class Test{
  public static void main(String[] args){
       Kugel k1 = new Kugel(1,2,1,3);
       Punkt p1 = new Kugel(0, 0, 0, 7);
       Punkt p2 = new Punkt(1, 2, 3);
       Kugel k2 = new Punkt(2, 2, 4);
       k1.aufblasen(2);
       p1.verschieben(1,1,1);
       k1.drucken();
       p1.drucken();
       p2.verschieben(1,1,1);
       p2.drucken();
       p2.aufblasen(3);
  }
} // class Test

class Punkt {
    private double x;
    private double y;
    private double z;

    public Punkt (double x, double y, double z){
       this.x = x;
       this.y = y;
       this.z = z;
    }

    public void drucken () {
       System.out.println ("x=" + x + ", y=" + y + ", z=" + z);
    }

    public void verschieben (double dx, double dy, double dz) {
       x= x+dx; y= y+dy; z=z+dz;
    }
}//class Punkt
//-------------------------------------------------------------
class Kugel extends Punkt {
   private double radius;

  public Kugel(double xx, double yy, double zz, double r) {
     super (xx, yy, zz);
     radius = r;
   }

   public void aufblasen (double f) {
       radius = radius*f;
   }

   public void drucken () {
       super.drucken();
       System.out.println ("Radius = " + radius);
   }
} //  class Kugel
```

Welche Anweisungen in der `main`-Methode der Klasse `Test` sind nicht korrekt? Unterstreichen Sie diese und erklären Sie, warum der Compiler sie als Fehler erkennt.

Welche Ausgabe liefert das korrekte Programm (ohne die fehlerhaften Anweisungen)?

Aufgabe 8

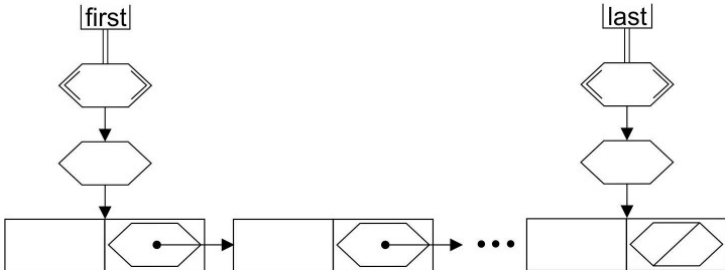
a) Es gibt drei grundlegende Klassen von Sortierverfahren:

(1) elementare Sortierverfahren
(2) spezielle Sortierverfahren
(3) höhere Sortierverfahren

Geben Sie für jede Klasse den Namen und Aufwand (i. A.) eines typischen Verfahrens an.

b) Mittels *Bubble-Sort* soll die Zahlenfolge **6, 11, 1, 4, 3, 9, 5, 2** aufsteigend sortiert werden. Geben Sie nach jedem Durchlauf die dabei entstehende Folge an.

Aufgabe 9
Gegeben sind folgende Listenstruktur, in der ganze Zahlen verwaltet werden, und zwei Klassen `Knoten` und `Liste`.




```
class Knoten {
  int data;
  Knoten next;
 Knoten(int i) {
    data = i;
    next = null;
  }
}

class Liste {
  Knoten first;
  Knoten last;
  Liste(){
    first = null;
    last = null;
  }
}
```

Ergänzen Sie die Klasse `Liste` und schreiben Sie folgende Methoden:

`boolean istSortiert()`: Die Methode gibt den Wert `true` zurück, wenn die Liste bez. `data`-Komponente aufsteigend sortiert ist, sonst `false`.

`int laenge()`: Die Methode gibt die Anzahl der Listenelemente zurück.

Aufgabe 10
Gegeben ist der vollständig geklammerte arithmetische Ausdruck **(1+(2*3))+(((4*5)+6)/7)**.

a) Skizzieren Sie den zugehörigen Formelbaum.

b) Markieren Sie den Operator mit der höchsten Priorität.

c) Durchlaufen Sie den Formelbaum im Pre-, In- und Postorder-Verfahren und geben Sie den sich jeweils ergebenen Ausdruck an.

Preorder-Verfahren:

Inorder-Verfahren:

Postorder-Verfahren: