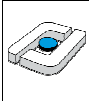


Thema:	Grundlagen Programmierung			
Dozent:	Prof. Dr. Rainer Roosmann	Seitennummer:	Seite 1 von 12	
Studiengang:	Informatik - Medieninformatik	Bearbeitungszeit:	120 Minuten	

Dies ist eine Probeklausur, die keine formalen Schlüsse auf die Form, die Struktur oder den Inhalt der endgültigen Klausur zulässt. Die Probeklausur wurde ursprünglich von Prof. S. Kleuker herausgegeben.

Es werden 50 Punkte zum Bestehen benötigt.

Aufgabe 1	/8	Note:
Aufgabe 2	/8	
Aufgabe 3	/12	
Aufgabe 4	/10	
Aufgabe 5	/38	
Aufgabe 6	/9	
Aufgabe 7	/15	
Gesamt	/100	

Aufgabe 1) Ausdrücke auswerten (8 Punkte)

Gegeben seien folgende Variablen.

Integer x = 42;

Double d = 42.42;

String s = "Hai";

Integer z = null;

Schreiben Sie auf, was die Auswertung der folgenden Ausdrücke ergibt.

Ausdruck	Auswertung
<code>x>12</code>	
<code>x>=d x<=d</code>	
<code>s+"fisch"</code>	
<code>42+s+x</code>	
<code>x==null z==null</code>	
<code>x.equals(z)</code>	
<code>x*x*x*d>2570000 && d-1>x</code>	
<code>x%41</code>	

Aufgabe 2) Analyse von if (8 Punkte)

Gegeben sei die folgende Klasse mit den angegebenen Methoden. Geben Sie die Werte der Objektvariablen `this.x` und `this.y` an, nachdem ein Objekt der Klasse erzeugt und die genannte Methode jeweils ausgeführt wurde.

```
public class Aufgabe02{
    private Integer x = 6;
    private Integer y = 7;

    public void m1(Boolean a, Boolean b){
        if(a){
            x=y;
            y=x;
        }
    }

    public void m2(Boolean a, Boolean b){
        if(a){
            if (b) {
                x=x*y;
            }
        }
        else {
            y=x*y;
        }
    }

    public void m3(Boolean a, Boolean b){
        if( a || b) {
            if(a || (!b)) {
                if(!a || b) {
                    x=x*y;
                }
            }
        }
    }

    public void m4(Boolean a, Boolean b){
        if(a && x>5) {
            y=8;
        }
        if(b && y<8) {
            x=5;
        }
        if( a && !a) {
            x=42;
        }
    }
}
```

a	b	m1(a,b)		m2(a,b)		m3(a,b)		m4(a,b)	
		x	y	x	y	x	y	x	y
true	true								
true	false								
false	true								
false	false								

Aufgabe 3) Analyse von Konstruktoren und Methodenaufrufen (12 Punkte)

Gegeben sei die folgende Klasse. Schreiben Sie auf, welche Ausgaben in welcher Reihenfolge ausgegeben werden, wenn ein Objekt der Klasse K erzeugt und die Methode doIt() aufgerufen wird. (gerne neben das Programm).

```
public class K{
    private Integer x;
    private Integer y;
    private EinUndAusgabe io = new EinUndAusgabe();

    public K(){
        this.x = 2;
        this.y = 2;
    }

    public K(Integer w){
        this.x = w;
        this.y = this.x;
    }

    public void mach(){
        this.x = this.x +this.x;
        this.x = this.x +this.x;
        this.x = this.x +this.x;
    }

    public Integer mult(){
        return this.x*this.y;
    }

    public K nouvo(Integer a){
        return new K(this.x + a);
    }

    public void show(){
        io.ausgeben("x:"+this.x+" y:"+this.y+"\n");
    }

    public void doIt(){
        K k1 = new K();
        k1.show();
        k1.mach();
        k1.show();
        K k2 = new K(3);
        k2.show();
        k2.mach();
        k2.show();
        K k3 = new K();
        K k4 = new K(k3.mult());
        k4.show();
        K k5 = k3.nouvo(4);
        k5.show();
    }
}
```

Aufgabe 4) Analyse von while (10 Punkte)

Gegeben sei die folgende Klasse. Schreiben Sie auf, welche Ausgaben in welcher Reihenfolge ausgegeben werden, wenn ein Objekt der Klasse L erzeugt und die Methode doIt() aufgerufen wird. (gerne neben das Programm, bei einer ArrayList werden alle Elemente innerhalb von eckigen Klammern ausgegeben).

```
import java.util.ArrayList;

public class L{
    private ArrayList<Integer> z = new ArrayList<Integer>();
    private EinUndAusgabe io = new EinUndAusgabe();

    public L(){
        for(Integer i=0; i<13; i=i+2){
            this.z.add(i);
        }
        this.io.ausgeben(this.z+"\n");
    }

    public Integer drei(){
        Integer ergebnis = 0;
        Integer zaehler=0;
        while(zaehler<this.z.size()){
            if(z.get(zaehler) % 3 == 0){
                io.ausgeben(z.get(zaehler)+"\n");
            }
            zaehler = zaehler + 1;
        }
        return ergebnis;
    }

    public void sum(){
        Integer zaehler=0;
        while(zaehler<this.z.size()-1){
            Integer tmp = z.get(zaehler)+z.get(zaehler+1);
            io.ausgeben(tmp+"\n");
            zaehler = zaehler + 1;
        }
    }

    public void mult(){
        Integer zaehler=1;
        while(zaehler<this.z.size()-4){
            Integer zaehler2=5;
            while(zaehler2<this.z.size()){
                Integer tmp = z.get(zaehler)*z.get(zaehler2);
                io.ausgeben(tmp+"\n");
                zaehler2 = zaehler2+1;
            }
            zaehler=zaehler+1;
        }
    }

    public void doIt(){
        L x= new L();
        x.drei();
        x.sum();
        x.mult();
    }
}
```

Aufgabe 5) Programmierung mit Sammlungen (2+2+2+6+9+11+6 = 38 Punkte)

Gegeben sei folgende Klasse, ergänzen Sie die geforderten Objektmethoden. Sie dürfen vereinfachend davon ausgehen, dass weder in den Objektvariablen, noch den Elementen der Sammlung null-Werte stehen.

```
public class Messreihe {  
  
    private String messort;  
    private ArrayList<Integer> messwerte;  
  
    public Messreihe(String messort) {  
        this.messort = messort;  
        this.messwerte = new ArrayList<Integer>();  
    }  
}
```

- a) Schreiben Sie für alle Objektvariablen get-Methoden.
- b) Schreiben Sie für alle Objektvariablen set-Methoden.
- c) Schreiben Sie eine Methode, mit der ein Element zur Sammlung hinzugefügt werden kann.
- d) Schreiben Sie eine Methode, die ein Integer-Objekt übergeben bekommt und die als Rückgabe liefert, wie oft der Wert dieses Objektes in this.messwerte vorkommt.
- e) Schreiben Sie eine Methode, die ein neues Messreihe-Objekt als Rückgabe liefert, in dem sich nur das erste, dritte, usw- Element aus this.messreihe befindet.
- f) Schreiben Sie eine Methode, die ein Messreihen-Objekt übergeben bekommt und ein neues Messreihe-Objekt als Rückgabe liefert, das die Produkte aller Werte der einen Messreihe mit allen Werten der anderen Messreihe enthält (aus [2,3] und [5,6] wird [10,12,15,18], dabei ist die Reihenfolge im Ergebnis egal).
- g) Schreiben Sie zwei JUnit-Tests mit denen Sie Ihre Methode aus e) einmal für eine Sammlung mit gerader und einmal mit ungerader Anzahl auf Korrektheit prüfen können (Hinweis: Die Aufgabe ist auch ohne eine vollständige Lösung zu e) bearbeitbar).

(Wiederholung)

- a) Schreiben Sie für alle Objektvariablen get-Methoden.
- b) Schreiben Sie für alle Objektvariablen set-Methoden.
- c) Schreiben Sie eine Methode, mit der ein Element zur Sammlung hinzugefügt werden kann.
- d) Schreiben Sie eine Methode, die ein Integer-Objekt übergeben bekommt und die als Rückgabe liefert, wie oft der Wert dieses Objektes in `this.messwerte` vorkommt.
- e) Schreiben Sie eine Methode, die ein neues Messreihe-Objekt als Rückgabe liefert, in dem sich nur das erste, dritte, usw.- Element aus `this.messreihe` befindet.
- f) Schreiben Sie eine Methode, die ein Messreihen-Objekt übergeben bekommt und ein neues Messreihe-Objekt als Rückgabe liefert, das die Produkte aller Werte der einen Messreihe mit allen Werten der anderen Messreihe enthält (aus `[2,3]` und `[5,6]` wird `[10,12,15,18]`, dabei ist die Reihenfolge im Ergebnis egal).
- g) Schreiben Sie zwei JUnit-Tests mit denen Sie Ihre Methode aus e) einmal für eine Sammlung mit gerader und einmal mit ungerader Anzahl auf Korrektheit prüfen können (Hinweis: Die Aufgabe ist auch ohne eine vollständige Lösung zu e) bearbeitbar).

(Wiederholung)

- a) Schreiben Sie für alle Objektvariablen get-Methoden.
- b) Schreiben Sie für alle Objektvariablen set-Methoden.
- c) Schreiben Sie eine Methode, mit der ein Element zur Sammlung hinzugefügt werden kann.
- d) Schreiben Sie eine Methode, die ein Integer-Objekt übergeben bekommt und die als Rückgabe liefert, wie oft der Wert dieses Objektes in `this.messwerte` vorkommt.
- e) Schreiben Sie eine Methode, die ein neues Messreihe-Objekt als Rückgabe liefert, in dem sich nur das erste, dritte, usw.- Element aus `this.messreihe` befindet.
- f) Schreiben Sie eine Methode, die ein Messreihen-Objekt übergeben bekommt und ein neues Messreihe-Objekt als Rückgabe liefert, das die Produkte aller Werte der einen Messreihe mit allen Werten der anderen Messreihe enthält (aus `[2,3]` und `[5,6]` wird `[10,12,15,18]`, dabei ist die Reihenfolge im Ergebnis egal).
- g) Schreiben Sie zwei JUnit-Tests mit denen Sie Ihre Methode aus e) einmal für eine Sammlung mit gerader und einmal mit ungerader Anzahl auf Korrektheit prüfen können (Hinweis: Die Aufgabe ist auch ohne eine vollständige Lösung zu e) bearbeitbar).

(Wiederholung)

- a) Schreiben Sie für alle Objektvariablen get-Methoden.
- b) Schreiben Sie für alle Objektvariablen set-Methoden.
- c) Schreiben Sie eine Methode, mit der ein Element zur Sammlung hinzugefügt werden kann.
- d) Schreiben Sie eine Methode, die ein Integer-Objekt übergeben bekommt und die als Rückgabe liefert, wie oft der Wert dieses Objektes in `this.messwerte` vorkommt.
- e) Schreiben Sie eine Methode, die ein neues Messreihe-Objekt als Rückgabe liefert, in dem sich nur das erste, dritte, usw.- Element aus `this.messreihe` befindet.
- f) Schreiben Sie eine Methode, die ein Messreihen-Objekt übergeben bekommt und ein neues Messreihe-Objekt als Rückgabe liefert, das die Produkte aller Werte der einen Messreihe mit allen Werten der anderen Messreihe enthält (aus `[2,3]` und `[5,6]` wird `[10,12,15,18]`, dabei ist die Reihenfolge im Ergebnis egal).
- g) Schreiben Sie zwei JUnit-Tests mit denen Sie Ihre Methode aus e) einmal für eine Sammlung mit gerader und einmal mit ungerader Anzahl auf Korrektheit prüfen können (Hinweis: Die Aufgabe ist auch ohne eine vollständige Lösung zu e) bearbeitbar).

Aufgabe 6) Polymorphie (9 Punkte)

Gegeben seien die folgenden drei Klassen.

```
public class K1{
    protected EinUndAusgabe io =
        new EinUndAusgabe();

    public void m1(String s){
        m2(s);
    }

    public void m2(String s){
        this.io.ausgeben(s+"1\n");
    }

    public void m3(String s){
        m1(s);
    }
}

public class K2 extends K1{

    @Override
    public void m1(String s){
        this.io.ausgeben(s+"2\n");
    }

    @Override
    public void m2(String s){
        super.m3(s);
    }

    @Override
    public void m3(String s){
        super.m2(s);
    }
}

public class K3 extends K2{

    @Override
    public void m1(String s){
        m3(s);
    }

    @Override
    public void m2(String s){
        this.io.ausgeben(s+"3\n");
    }

    @Override
    public void m3(String s){
        super.m3(s);
    }
}
```

In einem anderen Programm werden folgende drei Objekte erzeugt.

```
K1 k1 = new K1();
K1 k2 = new K2();
K1 k3 = new K3();
```

Welche Ausgabe liefern die folgenden einzelnen Zeilen, wenn sie jeweils nach den obigen Zeilen ausgeführt werden (falls sie keine Ausgabe angeben können, beschreiben Sie, was passiert)?

Aufruf	Ergebnis
k1.m1("A");	
k1.m2("B");	
k1.m3("C");	
k2.m1("D");	
k2.m2("E");	
k2.m3("F");	
k3.m1("G");	
k3.m2("H");	
k3.m3("I");	

Aufgabe 7) Fragen zur Programmierung (3+ 12 = 15 Punkte) [*Fragen passen zum bisherigen Veranstaltungsinhalt, beziehen sich generell auf die gesamte Veranstaltung*]

- a) In größeren Projekten ist es üblich, mit Coding-Guidelines Vorgaben für den Stil und das Aussehen von Programmen zu machen. Nennen Sie vier möglichst unterschiedliche Regeln für Java, die in solchen Coding-Guidelines stehen können. Nennen Sie einen guten Grund, der für Coding-Guidelines spricht.
- b) Zur Verwaltung von Sammlungen von Objekten kann man in Java u. a. Array-, HashSet, ArrayList und HashMap-Objekte nutzen. Erklären Sie die wesentlichen Unterschiede der genannten Klassen und an Beispielen, wann man Sie einsetzen sollte.

[leeres Blatt, für Lösungen nutzbar]