

Aufgabe 1. (10 Punkte)

Programmieren Sie zwei Methoden `int itPartialSum (int i, int j)` und `int rekPartialSum (int i, int j)`, welche beide die Summe aller ganzen Zahlen zwischen i und j berechnen:

$$\sum_{n=i}^j n$$

Implementieren Sie in der Methode `itPartialSum` die partielle Summe mit einer **for**-Schleife. Implementieren Sie die Methode `rekPartialSum` als rekursive Methode, also ohne die Verwendung eines Schleifenkonstrukts.

Bemerkung: Falls $i > j$, so ist der Wert dieser Summe als 0 definiert.

Lösungsvorschlag: Siehe Quellcodepaket.

Aufgabe 2. (10 Punkte)

Das Programm soll ein Array `eingabe` in ein Array `ausgabe` kopieren und dabei jeweils den Wert 23 durch $f(23)$, den Wert 8 durch $f(8)$ und 2 durch den Wert $f(2)$ ersetzen.

```
public class scanArray{

    public static int f(int x)
    {
        return x*x*x+x*x;
    }

    public static void main(String [ ] args) {
        int[] eingabe = new int[args.length];
        for(int ii = 0; ii < args.length; ++ii)
            {eingabe[ii] = Integer.getInteger(args[ii]); }
        int[] ausgabe = new int[eingabe.length];

    } // Ende main
} // Ende class scanArray
```

Lösungsvorschlag: Siehe Quellcodepaket.

Aufgabe 3. (5 Punkte)

Geben Sie für jeden der folgenden 5 Ausdrücke seinen Typ und seinen Wert an, oder begründen Sie warum ein Ausdruck nicht ausgewertet werden kann!

Ausdruck	Typ	Wert
<code>1/2 == 1.0/2.0f && (true false)</code>	Boolean	false
<code>2.5E1+1/2</code>	Double	25.0
<code>3 * 4-5 * 6</code>	Integer	-28
<code>5.0 / (float)2.0</code>	Double	2.5
<code>1.0 == 1 2</code>	Der Operator	kann nicht für boolean und int ausgewertet werden

Aufgabe 4. (10 Punkte)

Was ist die Ausgabe der folgenden drei Schleifenkonstrukte? Geben Sie jeweils die Folge der ausgegebenen Zahlen an!

a)

```
for (int i=1023; i>0; i/=2)
    System.out.println(i%2);
```

Die Ausgabe ist:

1
1
1
1
1
1
1
1
1
1
1
1

b)

```
int n=13;
while (n>1) {
    System.out.println(n);
    if (n%2==0)
        n/=2;
    else
        n=3*n+1;
}
```

Die Ausgabe ist:

13
40

20
10
5
16
8
4
2

c)

```
int i=1;
int j=1;
while (i<=100) {
    System.out.println(i%2==0?i:j);
    j+=2;
    i+=j;
}
```

Die Ausgabe ist:

1
4
5
16
9
36
13
64
17
100

Aufgabe 4. (20 Punkte)

Schreiben Sie eine Klasse **Student**. Die Klasse soll folgende Funktionalitäten aufweisen:

- Jeder Student hat einen Vornamen, Nachnamen und eine Matrikelnummer. Schreiben sie einen passenden Konstruktor.
- Beachten Sie, dass sich der Name, Vorname und Matrikelnummer eines Studenten nicht verändern darf, die hinterlegten Daten aber von anderen Klassen zugreifbar aber nicht veränderbar sind.
- In einer Liste soll abgespeichert werden, welche Prüfungen ein Student bisher bestanden hat. Dabei wird jede Prüfung durch einen String mit dem Prüfungsnamen repräsentiert.
- Schreiben Sie eine Methode, die das hinzufügen von einer bestanden Prüfung erlaubt.
- Ist eine Prüfung bestanden, so kann diese bestanden Prüfung nicht mehr aberkannt werden.
- Schreiben Sie eine Methode die alle bestandenen Prüfungen ausgibt.

Tragen Sie Sorge, dass die obigen Anforderungen unter Anderem durch durch Kapselung gewährleistet sind.

Lösungsvorschlag: Siehe Quellcodepaket.

Aufgabe 5. (20 Punkte)

Betrachten Sie den folgenden Quellcode und beantworten Sie danach die gestellten Fragen:

```
public class a {
    private static int cnt = 0;
    private int x,y;
    private Scanner sc;
    public a(int ii, int jj){
        x=ii;
        y=jj;
    }

    public a(String b) {
        String[] cur = b.split(":");
        for(int ii =0; ii < cur.length; ++ii)
        {
            init(cur[ii]);
            if(ii ==0)
                x =sc.nextInt();
            else
                y =sc.nextInt();
        }
    }

    public a(String b, String c) {
        x = new Scanner(c).nextInt();
        y = new Scanner(b).nextInt();
    }

    private void init(String s) {
        ++cnt;
        sc = new Scanner(s);
    }

    public static void main(String[] argv) {
        a x = new a(14,7);
        a y = new a("3","25");
        a z = new a("10:5");
        a b = new a("9:1:93:8");
    }
}
```

Geben Sie den Wert von `cnt` nach der Initialisierung von `b` an:

Der Wert von `cnt` ist 6.

Geben Sie die Summe der `x`-Werte der vier initialisierten Objekte an: Die Summer aller `x`-Werte ist 58.

Geben Sie die Summe der `y`-Werte der vier initialisierten Objekte an: Die Summer aller `y`-Werte ist 23.

Aufgabe 6. (10 Punkte)

Betrachten Sie den folgenden Quellcode:

```
public class d{
    public static void func(int[] a, int[] x, int y, int z){
        int[] b =a;
        x=a;
        a=b;
        int c = y;
        y =z;
        z=c;
    }

    public static void main(String[] argv){
        int[] a = {4,95,2};
        int[] a1 = {5,34,90};
        int a2 = 8;
        int[] b1 = {8,5,432};
        int b2 = 1;
        int[] c1 = {23,5,8};
        int c2 = 23;
        func(a1,b1,a2,c2);
        func(a1,c1,b2,c2);
        func(c1,b1,b2,c2);
    }
}
```

Geben Sie die Inhalte der Variablen a, a1, a2, b1, b2, c1, c2 nach dem letzten Aufruf von func an.

Lösungsvorschlag: Die Werte der Variablen sind wie folgt:

```
a = {4, 95, 2}
a1 = {5, 34, 90}
b1 = {8, 5, 432}
c1 = {23, 5, 8}
a2 = 8
b2 = 1
c2 = 23
```

Aufgabe 7. (15 Punkte)

Betrachten Sie den folgenden Quellcode und erweitern Sie ihn sinnvoll.

```
public class TCommand {
    final Turtle myTurtle;

    public TCommand(Turtle mT) {
        myTurtle = mT;
    }

    public void execute() { }
}

public class TForw extends _____ { // erweitern Sie sinnvoll
    final double dist;

    TForw(Turtle mT, double d) {
        // Vervollstaendigen sie den Code so, dass myTurtle benutzbar ist
        -----
        dist = d;
    }

    public void execute() {
        myTurtle.goForward(dist);
    }
}

public class TTurn extends _____ { // erweitern Sie sinnvoll
    final double angle;

    TTurn(Turtle mT, double an) {
        // Vervollstaendigen sie den Code so, dass myTurtle benutzbar ist
        -----
        angle = an;
    }

    public void execute() {
        myTurtle.turnLeft(angle);
    }
}

public class TurtleCommandMethod {
    public static void main(String[] args) {
```

```

Turtle mTurtle = new Turtle(0.5, .1, 90);
// Initialisieren Sie hier eine sinnvolle passende Datenstruktur!

----- commands = -----;

commands.add(new TForw(mTurtle, .1));
commands.add(new TTurn(mTurtle, 30));
commands.add(new TForw(mTurtle, .1));
commands.add(new TTurn(mTurtle, 30));
commands.add(new TForw(mTurtle, .1));
        // Fuehren sie jedes Kommando in commands aus:
for (-----) { com.execute();}
}
}

```

Lösungsvorschlag: Siehe Quellcodepaket.