Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

Grundlagen der Programmierung & Algorithmen und Datenstrukturen

Übungen zum Part 3 – Lösungsalternativen

Übung 3

Übung 3

- Erstellen Sie neben der main()-Methode drei weitere Methoden denen Sie die Namen funktion1(), funktion2() und funktion3() geben. Im Rumpf der Methode soll lediglich eine Ausgabeanweisung in folgender Weise erfolgen:

```
System.out.print(„Funktion x ");
```

- Alle Methoden erhalten eine leere Parameterliste und erzeugen bei Beendigung kein Rückgabewert.
- Rufen Sie die Methoden innerhalb der main()-Methode nacheinander auf.
- Ändern Sie die Methoden so ab, dass beim Aufruf ein Integer-Wert als Parameter übergeben werden kann. (optional)

Übung 3

Ohne Parameterübergabe

```java
public class Aufgabe3 {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        funktion1();
        funktion2();
        funktion3();
    }

    public static void funktion1() {
        System.out.println("Funktion 1");
    }

    public static void funktion2() {
        System.out.println("Funktion 2");
    }

    public static void funktion3() {
        System.out.println("Funktion 3");
    }

}
```

mit Parameterübergabe

```java
public class Aufgabe3 {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        funktion1(1);
        funktion2(2);
        funktion3(3);
    }

    public static void funktion1(int a) {
        System.out.println("Funktion "+a);
    }

    public static void funktion2(int a) {
        System.out.println("Funktion "+a);
    }

    public static void funktion3(int a) {
        System.out.println("Funktion "+a);
    }

}
```

Übung 4

Übung 4

- Schauen Sie sich das folgende Programmfragment an und überlegen Sie welche Zahl jeweils ausgegeben wird.
- Prüfen Sie ihr Ergebnis mit Hilfe eines Java-Programms?
- Diskutieren Sie ggf. vorhandene Abweichungen

```
int a = 15;
int b = 13;
int c = a%b;
int d = c%b;
System.out.println("mod "+c);
System.out.println("mod "+d);
a = ++a + a++;
System.out.println(a);
a = a + a;
System.out.println(a);
b++;
System.out.println(b);
--b;
System.out.println(b);
b *= b;
System.out.println(b);
a += b;
System.out.println(a);
```

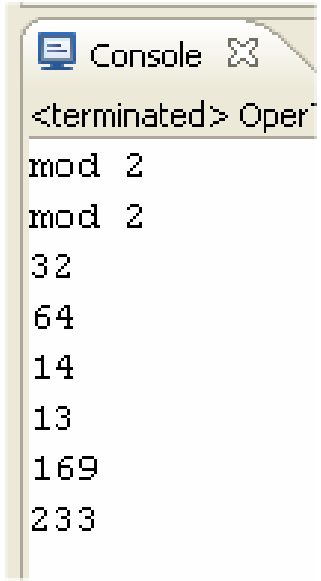
Übung 4

```
public class OperTest1 {

    public static void main(String[] args){
        getTestOper();
    }

    public static void getTestOper() {
        int a = 15, b = 13;
        int c = a % b;
        int d = c % b;
        System.out.println("mod " + c);
        System.out.println("mod " + d);
        a = ++a + a++;
        System.out.println(a);
        a = a + a;
        System.out.println(a);
        b++;
        System.out.println(b);
        --b;
        System.out.println(b);
        b *= b;
        System.out.println(b);
        a += b;
        System.out.println(a);
    }
}
```

Ergebnisse nach Ausführung:



Übung 5

Übung 5

- Geben Sie drei Möglichkeiten an, den Wert 1 zur Variablen `count` zu addieren.
- Welche Werte nimmt die boolsche Variable `result` an, wenn `x = 6` und `y = 2` gesetzt wird?
  - `result = x > y; ?`
  - `result = x < y; ?`
  - `result = x == 0; ?`
  - `result = x != 0; ?`
  - `result = x <= 7; ?`
- Welche Werte nimmt die boolsche Variable `result` an, wenn x = 4, y = 2 und z = 0 gesetzt wird?
  - `result = (x > y && z == 0) ?`
  - `result = (x == 0 || z == 0) ?`
  - `result = !(y == 2 || z < x) ?`

Lösung – Teil1

```
public class TestUE5 {

     * @param args
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        getInkr();
        getBoolA();
        getBoolB();
    }

    public static void getInkr() {
        int count = 0;
        count = count + 1;
        count += 1;
        count++;
        ++count;
        System.out.println(count);
    }

    public static void getBoolA() {

    public static void getBoolB() {

}
```

```
public class TestUE5 {

     * @param args
    public static void main(String[] args) {

    public static void getInkr() {

    public static void getBoolA() {
        boolean result = false;
        int x = 6;
        int y = 2;
        result = x > y;
        System.out.println(result);
        result = x < y;
        System.out.println(result);
        result = x == 0;
        System.out.println(result);
        result = x != 0;
        System.out.println(result);
        result = x <= 7;
        System.out.println(result);
    }
```

Lösung – Teil2

```
public class TestUE5 {

     * @param args
    public static void main(String[] args) {

    public static void getInkr() {

    public static void getBoolA() {

    public static void getBoolB() {
        boolean result = false;
        int x = 6;
        int y = 2;
        int z = 0;
        result = (x > y && z == 0);
        System.out.println(result);
        result = (x == 0 || z == 0);
        System.out.println(result);
        result = !(y == 2 || z < x);
        System.out.println(result);
    }

}
```

Ergebnisse nach Ausführung:

Übung 6

Übung 6

- Testen Sie beide der angegebenen Möglichkeiten zur Eingabe von Werten aus. Nutzen Sie dafür jeweils eine eigene Methode entsprechend der folgenden Signaturen.
  - Eingabe innerhalb eine kleinen Windows!
    - public static void getValueSwing() {…}
  - Eingabe innerhalb der Konsole
    - public static void getValueStream() {…}
- Geben Sie die eingegebenen Werte direkt innerhalb der beiden Methoden auf der Konsole aus.
- Geben Sie die Werte innerhalb der main()-Methode aus, verändern Sie dafür die Signaturen beider Methoden. (optionale Aufgabe)

Lösung

```java
import java.io.*;

public class EingabeTestB {

    /**
     * @param args
     */
    public static void main(String[] args) throws NumberFormatException,
            IOException {
        // TODO Auto-generated method stub
        System.out.println(getValueSwing());
        System.out.println(getValueStream());
    }

    // Wert einlesen via Window
    public static int getValueSwing() {
        String eingabe = JOptionPane.showInputDialog("Zahl eingeben!");
        int zahl = Integer.parseInt(eingabe);
        return zahl;
    }

    // Wert einlesen via Steam
    public static int getValueStream() throws NumberFormatException,
            IOException {
        System.out.print("Zahl eingeben: ");
        BufferedReader din = new BufferedReader(
                new InputStreamReader(System.in));
        int zahl = Integer.parseInt(din.readLine());
        return zahl;
    }
}
```

Lösung inkl. Option

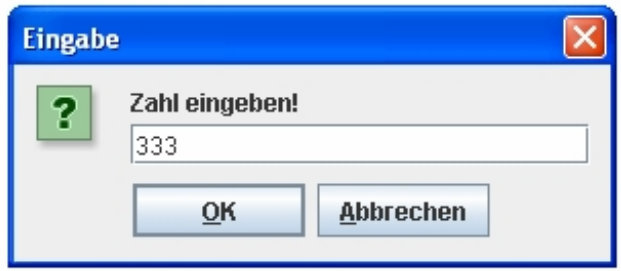
```
import java.io.*;
import javax.swing.JOptionPane;

public class EingabeTest {

     * @param args
    public static void main(String[] args) throws NumberFormatException,
            IOException {
        // TODO Auto-generated method stub
        System.out.println(getValueSwing());
        System.out.println(getValueStream());
    }

    // Wert einlesen via Window
    public static int getValueSwing() {
        String eingabe = JOptionPane.showInputDialog("Zahl eingeben!");
        int zahl = Integer.parseInt(eingabe);
        return zahl;
    }

    // Wert einlesen via Window
    public static int getValueStream() throws NumberFormatException,
            IOException {
        System.out.print("Zahl eingeben: ");
        BufferedReader din = new BufferedReader(
                new InputStreamReader(System.in));
        int zahl = Integer.parseInt(din.readLine());
        return zahl;
    }
}
```



Übung 7

Übung 7

- Schreiben Sie eine Funktion, der eine Jahreszahl übergeben wird und die einen wahren Wert (`true`) zurückliefert, wenn es sich um ein Schaltjahr handelt. Falls der übergebene Wert kein Schaltjahr ist, soll ein falscher Wert (`false`) zurückgeliefert werden.

  Signatur der Funktion:

  ```
  public static boolean isSchaltjahr(int jahr) { …}
  ```

  Ein Jahr ist kein Schaltjahr, wenn die Jahreszahl nicht durch 4 teilbar ist. Ein Jahr ist ein Schaltjahr, wenn die Jahreszahl durch 4, aber nicht durch 100 teilbar ist. Ein Jahr ist ebenfalls ein Schaltjahr, wenn die Jahreszahl durch 4, durch 100 und durch 400 teilbar ist.

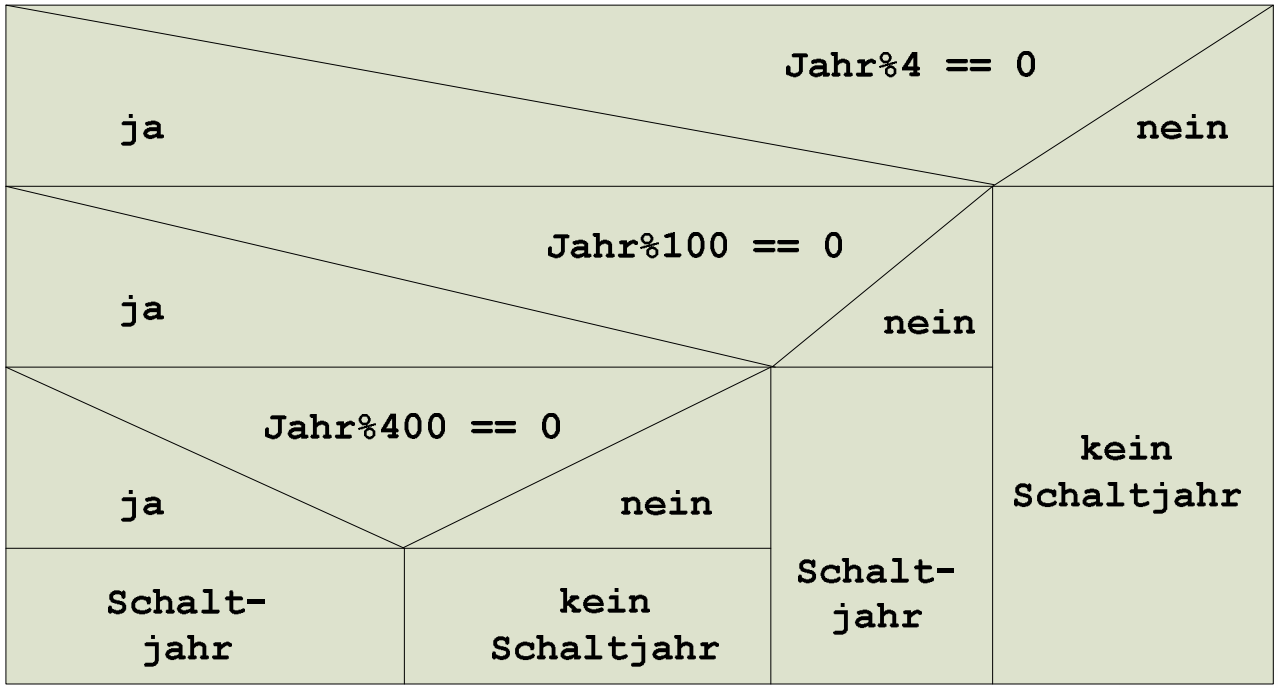
- Schreiben Sie dazu eine main-Funktion, die vom Benutzer eine Jahreszahl einliest, die Funktion `isSchaltjahr()` ausführt und dem Benutzer das Ergebnis mitteilt.

Struktogramm - grob

Jahreszahl vom Nutzer einlesen `einlesenJahr()`
Jahreszahl auf Schaltjahr prüfen `isSchaltjahr (int jahr)`
Ergebnis der Schaltjahrprüfung ausgeben `ausgabeErgebnis(boolean result, int jahr)`

Struktogramm - fein

Algorithmus zum prüfen auf Schaltjahr



Übersicht

```
import javax.swing.*;

public class Scaltjahr {

     * @param args

    public static void main(String[] args) {

    // Funktion zum Einlesen einer Jahrszahl
    public static int einlesenJahr(){

    // Funktion zum Pruefen auf Schaltjahr
    public static boolean isSchaltjahr(int jahr){

    // Funktion zur Ergebnisausgabe
    public static void ausgabeErgebnis(boolean result, int jahr){
}
```

main-Funktion

```
import javax.swing.*;

public class Scaltjahr {

    /**
     * @param args
     */

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        // Funktion zum Einlesen einer Jahrszahl aufrufen
        int jahr = einlesenJahr();
        // Funktion zum Pruefen auf Schaltjahr aufrufen
        boolean result = isSchaltjahr(jahr);
        // Funktion zur Ausgabe des Ergebnisses aufrufen
        ausgabeErgebnis(result, jahr);


    }
```

einlesenJahr-Funktion

```
public class Scaltjahr {

     * @param args

    public static void main(String[] args) {

    // Funktion zum Einlesen einer Jahrszahl
    public static int einlesenJahr(){
        String eingabe = JOptionPane.showInputDialog("Zahl eingeben!");
        int zahl = Integer.parseInt(eingabe);
        return zahl;
    }

    // Funktion zum Pruefen auf Schaltjahr
    public static boolean isSchaltjahr(int jahr){

    // Funktion zur Ergebnisausgabe
    public static void ausgabeErgebnis(boolean result, int jahr){
}
```

isSchaltjahr-Funktion

```
// Funktion zum Pruefen auf Schaltjahr
public static boolean isSchaltjahr(int jahr){
    boolean erg = false;
    if ( jahr % 4 == 0 ){
        if ( jahr % 100 == 0 ){
            if ( jahr % 400 == 0 ){
                erg = true;
                // System.out.println( "" + jahr + " ist ein Schaltjahr!");
            }
            else {
                erg = false;
                // System.out.println( "" + jahr + " ist kein Schaltjahr!");
            }
        }
        else {
            erg = true;
            // System.out.println( "" + jahr + " ist ein Schaltjahr!");
        }
    }
    else {
        erg = false;
        // System.out.println( "" + jahr + " ist kein Schaltjahr!");
    }

    return erg;
}
```

isSchaltjahr-Funktion (Alternative)

```
    // Funktion zum Einlesen einer Jahrszahl
    public static int einlesenJahr(){..

    // Funktion zum Pruefen auf Schaltjahr
    public static boolean isSchaltjahr1(int jahr){..

    public static boolean isSchaltjahr2(int jahr){
        boolean result = false;
        result=((jahr%4==0)&&(jahr%100!=0))||((jahr%4==0)&&(jahr%100==0)&&(jahr%400==0));
        return result;
    }

    // Funktion zur Ergebnisausgabe
    public static void ausgabeErgebnis(boolean result, int jahr){..
}
```

ausgabeErgebnis-Funktion

```
import javax.swing.*;

public class Scaltjahr {

     * @param args

    public static void main(String[] args) {

    // Funktion zum Einlesen einer Jahrszahl
    public static int einlesenJahr(){

    // Funktion zum Pruefen auf Schaltjahr
    public static boolean isSchaltjahr(int jahr){

    // Funktion zur Ergebnisausgabe
    public static void ausgabeErgebnis(boolean result, int jahr){
        if (result){
            System.out.println( "Das Jahr " + jahr + " war ein Schaltjahr");
        }else{
            System.out.println( "Das Jahr " + jahr + " war kein Schaltjahr");
        }
    }
}
```

Übung 8

Übung 8

- Programmieren Sie eine Zählschleife, die die ersten einhundert geraden Zahlen in absteigender Reihenfolge ausgibt!
- Geben Sie innerhalb der Zählschleife auch die zur geraden Zahl gehörige Quadratzahl aus.
- Geben Sie darüber hinaus die jeweilige 2-er Potenz aus, dabei handelt es sich um das Ergebnis von 2^x, mit x als Wert korrespondierenden geraden Zahl!

  Bem.: mit Hilfe der Methode `pow(double basis, double exponent)`, diese steht im Rahmen der Klasse `Math` zur Verfügung, lässt sich die Zweierpotenz berechnen.

Lösungsansatz

```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    int j = 0;
    for (int i=200; i>1; i-=2){
        if (i%2==0){
            System.out.println("Zahl-Nr.= "+(++j));
            System.out.println("gerade Zahl= "+i);
            System.out.println("2er Potenz= "+Math.pow(2,i));
            System.out.println("Quadratzahl= "+i*i);
        }
    }
}
```

Übung 8b

Aufgabe

Schreiben Sie ein Programm, welches die folgende Zahlenausgabe

erzeugt:

5678910

67891011

789101112

8910111213

91011121314

Lösungsansatz

```
package zahlen;

public class ZahlMatrix {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        {

                for ( int i = 5;  i < 10; i++)
                {
                        for ( int j = i;  j <= i+5; j++)
                        {
                                System.out.print(j);
                        }
                        System.out.println();
                }


        }
    }

}
```

Übung 9

Übung 9

Kreiszahlberechnung nach Leibniz:

- Der Mathematiker Leibniz hat herausgefunden, dass man die Kreiszahl Pi auf die folgende Art und Weise berechnen kann:

$$4\sum_{v=0}^{\infty}\frac{(-1)^v}{2v+1} = (1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\frac{1}{9}-\frac{1}{11}+\ldots)*4 = \pi$$

- Entwerfen (Struktogramm) und implementieren (Java) Sie ein Programm, welches nach diesem Verfahren Pi berechnet. Da sich das Ergebnis immer weiter der tatsächlichen Zahl Pi annähert, je weiter die Reihe berechnet wird, soll das Programm vor der Berechnung fragen, wie weit die Reihe gebildet werden soll, damit keine unendliche Schleife entsteht.

Lösungsansatz

```
public class PIBerechnung {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("PI-Berechnung");
        double pi = leibniz_pi(1000000);
        System.out.println("PI-Berechnung " + pi);

    }
```

Lösungsansatz

```
public static double leibniz_pi(int durchlauf) {

        double pi   = 1; double eins = 1; int nenner  = 1;

        for (int i = 1; i <= durchlauf; i++) {
            nenner = nenner + 2;
            if ((i % 2) == 1) { // ist die Zahl gerade?
                pi = pi - (eins/nenner);
            } else {
                pi = pi + (eins/nenner);
            }
        }

        return 4 * pi;
}
```

Projektübersicht – alle Methoden

```
public class PIBer {

    /**
     * @param args
     */
    public static void main(String[] args) {..

    public static double getPI_A(int durchlauf) {..

    public static double getPI_B(int durchlauf) {..

    public static double getDiff(double pi, int durchl){..

    public static void setAusgabe(double pi, int durchl, double diff){..
```

main-Methode

```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    int durchl = 20;
    double erg = 0, diff = 0;

    System.out.println("PI-Berechnung A");
    erg = getPI_A(durchl);
    diff = getDiff(erg, durchl);
    setAusgabe(erg, durchl, diff);

    System.out.println("PI-Berechnung B");
    erg = getPI_B(durchl);
    diff = getDiff(erg, durchl);
    setAusgabe(erg, durchl, diff);
}
```

Lösungsalgorithmus - A

```
public static double getPI_A(int durchlauf) {
    double pi    = 1;
    double eins = 1;
    int nenner   = 1;

    for (int i = 1; i <= durchlauf; i++) {
        nenner = nenner + 2;
        if ((i % 2) == 1) { // ist die Zahl gerade?
            pi = pi - (eins/nenner);
        } else {
            pi = pi + (eins/nenner);
        }
    }
    //Rückgabe des Ergebnisse
    return 4 * pi;
}
```

Lösungsalgorithmus - B

```
public static double getPI_B(int durchlauf) {

    // Pi mit Leibnitz-Reihe
    double pi=0, erg=0;
    //Schleife für die Berechnung
    for (int i=0; i<=durchlauf; i++) {
        //Bildungsgesetz
        pi += (1/(2.0*i +1)) * Math.pow(-1, i);
    }
    erg = 4*pi;

    //Rückgabe des Ergebnisse
    return erg;
}
```

Differenz zu Math.PI

```
public static double getDiff(double pi, int durchl){
    double diff = 0;

    //Differenz von Pi mit errechneten Wert von Pi

    diff=Math.PI - pi;

    return diff;
}
```

Ausgabe der Ergebnisse

```
public static void setAusgabe(double pi, int durchl, double diff){
    System.out.print("Leibnitz-Reihe bei " + durchl);
    System.out.println(" Schritten ergibt: " + pi);
    if (diff<0){
        System.out.print("Pi ist um " + -1*diff);
        System.out.println(" kleiner als der errechnete Leibnitz-Wert!");
    }else{
        System.out.println("Pi ist um " + diff);
        System.out.println(" groesser als der errechnete Leibnitz-Wert!");
    }

}
```

Übung 10

Aufgabe

- Schreiben sie zwei Methoden, die Fahrenheit zu Celsius bzw. umgekehrt berechnen.
  - Formel Fahrenheit:Celsius: tempC = (tempF-32.0)*5.0/9.0;
  - Formel Celsius: Fahrenheit : tempF = tempC*9.0/5.0+32.0;
- Ausgabe innerhalb der jeweiligen Methode (inkl. Lehrzeile)
  - tempC + " Grad Celsius ergeben " + tempF + " Fahrenheit"
  - tempF + " Fahrenheit ergeben " + tempC + " Grad Celsius"
- Verwenden Sie die Methoden innerhalb einer Java-Applikation

Lösungsansatz - simple

```
public static void tempFtoC (double temp){
        double tempF = temp;
        double tempC = (tempF-32.0)*5.0/9.0;
        System.out.println(tempF + " Fahrenheit ergeben " +
        tempC + " Grad Celsius \n");
}

public static void tempCtoF (double temp){
        double tempC = temp;
        double tempF = tempC*9.0/5.0+32.0;
        System.out.println(tempC + " Grad Celsius ergeben "
        + tempF + " Fahrenheit \n");
}
```

Lösungsansatz - Parameterübergabe

```
package temperatur;

public class TempBerechnungen {

     * @param args
    public static void main(String[] args) {

    public static double tempFtoC (double temp){
        double tempF = temp;
        double tempC = (tempF-32.0)*5.0/9.0;
        System.out.println(tempF + " Fahrenheit ergeben " + tempC + " Grad Celsius \n");
        return tempC;
    }

    public static double tempCtoF (double temp){
        double tempC = temp;
        double tempF = tempC*9.0/5.0+32.0;
        System.out.println(tempC + " Grad Celsius ergeben " + tempF + " Fahrenheit \n");
        return tempF;
    }
}
```

Übung 11 - Schleifenumformung

Übung 11

- Implementieren Sie in Java eine Zählschleife, die von n=1 auf 23 mit der Schrittweite 8 zählt.
- Formen Sie diese Schleife in eine funktional äquivalente kopfgesteuerte Schleife um.
- Formen Sie diese Schleife in eine funktional äquivalente fußgesteuerte Schleife um.

Lösungsansatz

```
// Zaehlschleife

for (int i=1; i<=23; i+=8) {

        // Ausgabe des jeweiligen i

        System.out.println("i ist"+i);

}
```

Lösungsansatz

```
int j=1;
// Kopfgesteuerte Schleife
while(j<=23){
        // Ausgabe des jeweiligen j;
        System.out.println("j ist"+j);
        // Abbildung der Schrittweite
        j=j+8;
}
```

Lösungsansatz

```
int k=1;
if (k<=23){
        // Kopfgesteuerte Schleife
       do{
              // Ausgabe des jeweiligen k;
              System.out.println("k ist"+k);
              // Abbildung der Schrittweite
              k=k+8;

              } while(k<=23);
       }
else{
              System.out.println("k ist zu gross"+k);
}
```