



17.4.2016

Handbuch OOP

Übungen zur Objektorientierter
Programmentwicklung

Patrick Seidel

[FIRMENNAME]

INHALT

Aufgabe 1: Lösung einer quadratischen Gleichung.....	2
A: Spezifikation des Problems.....	2
B: Analyse des Problems.....	2
C: Entwurf des Programms	2
D: Erstellung des Quellcodes	3
Aufgabe 2: Verarbeitung von Messwerten	4
A: Spezifikation des Problems.....	4
B: Analyse des Problems.....	4
C: Entwurf des Programms	4
D: Erstellung des Quellcodes	5
Aufgabe 3: Berechnung des GGT	5
A: Spezifikation des Problems.....	5
Aufgabe 4: Berechnung der Summe von natürlichen Zahlen	5
Aufgabe 5: Bestimmung eines Minimums	5
A: Spezifikation des Problems.....	5
B: Analyse des Problems.....	5
C: Entwurf des Programms	5
D: Erstellung des Quellcodes	6
Aufgabe 6: Berechnung des GGT mit einem weiteren Algorithmus	7
Aufgabe 7: Fakultäten berechnen.....	7
Aufgabe 8: Darstellung von e^x durch eine Taylor Reihe.....	7
Aufgabe 9: Darstellung von $\sin(x)$ durch eine Taylor Reihe	7
Aufgabe 10: Insertion Sort	7
A: Spezifikation des Problems.....	7
B: Analyse des Problems.....	7
C: Entwurf des Programms	7

AUFGABE 1: LÖSUNG EINER QUADRATISCHEN GLEICHUNG

A: SPEZIFIKATION DES PROBLEMS

Beschreiben Sie einen Algorithmus zur Lösung einer quadratischen Gleichung.

B: ANALYSE DES PROBLEMS

Die quadratische Gleichung kann mithilfe der pq-Formel gelöst werden. Hierzu müssen zuerst die Variablen p und q aus der Gleichung herausgefunden werden. Anschließend werden p und q in die Formel eingesetzt und ein Ergebnis für x1 und x2 ermittelt.

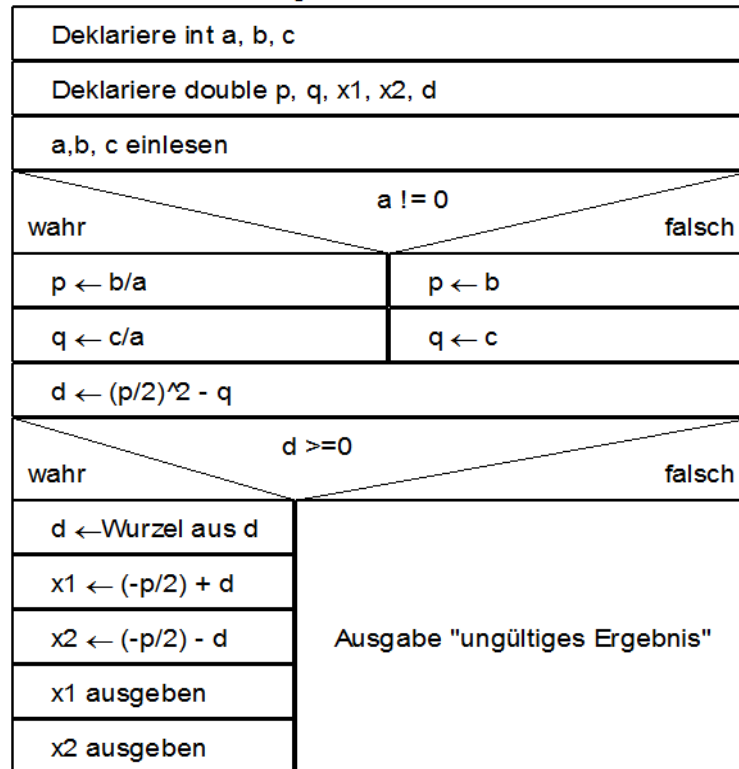
Beispielhafter Ablauf

- 1: $2x^2 + 3x + 40 = 0 \quad | :2$
- 2: $\Rightarrow x^2 + 1,5x + 20 = 0$
- 3: $p = 1,5; q = 20;$
- 4: $x_{1,2} = (-p/2) \pm \sqrt{(p/2)^2 - q}$
- 5: $x_1 = (-1,5/2) + \sqrt{(1,5/2)^2 - 20}$
- 6: $x_2 = (-1,5/2) - \sqrt{(1,5/2)^2 - 20}$

C: ENTWURF DES PROGRAMMS

Pq-Formel

Quadratische Gleichung lösen



D: ERSTELLUNG DES QUELLCODES

```
public class Pq_Formel {
    public static void main(String[] args) {
        int a,b,c;
        double p,q,d,x1,x2;           //Deklariere Variablen
        a = 2;                         // Initialisierung a,b,c
        b = 13;
        c = 7;
        if(a != 0){                    // Prüfe ob a ungleich 0
            p = b/a;
            q = c/a;
        }else{
            p = b;
            q = c;
        }
        d = (p/2)*(p/2) - q;
        if(d >= 0){
            d = Math.sqrt(d);
            x1 = (-p/2)+ d;
            x2 = (-p/2) - d;
        }
    }
}
```

```

        System.out.println("x1:" + "\t" + x1);
        System.out.println("x2:" + "\t" + x2);
    }else{
        System.out.println("ungültiges Ergebnis");
    }
}
}

```

AUFGABE 2: VERARBEITUNG VON MESSWERTEN

A: SPEZIFIKATION DES PROBLEMS

Bildung des arithmetischen Mittels von Zahlen

B: ANALYSE DES PROBLEMS

Übergebe eine Liste von Zahlen:

Bsp: 2, 6, 7, 10, 23

Bilde das arithmetische Mittel

$$\Rightarrow (2 + 6 + 7 + 10 + 23) / 5 = 9,6$$

C: ENTWURF DES PROGRAMMS

Arithmetisches Mittel

Deklariere zahlenliste

deklariere int count, summe

count ← 1

übergebe zahlenliste

für int i = 0; i to Länge zahlenliste; i++;

count += i;

summe += zahlenliste[i];

summe / count ausgeben

D: ERSTELLUNG DES QUELLCODES

```
public static double Mittelwert(){
    double [] zahlenliste = new double [10];
    double count = 0;
    double summe = 0;
    double result;
    for (int i = 0; i < zahlenliste.length; i++){
        zahlenliste [i] = (Math.random() * 100);
        System.out.println(zahlenliste[i]);
    }
    for (int i = 0; i < zahlenliste.length; i++){
        summe += zahlenliste[i];
        count += 1;
    }
    result = summe / count;

    return result ;
}
```

AUFGABE 3: BERECHNUNG DES GGT

A: SPEZIFIKATION DES PROBLEMS

Berechne den größten gemeinsamen Teiler zweier Zahlen mithilfe des Euklidischen Algorithmus.

AUFGABE 4: BERECHNUNG DER SUMME VON NATÜRLICHEN ZAHLEN

AUFGABE 5: BESTIMMUNG EINES MINIMUMS

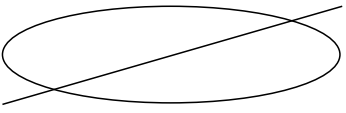
A: SPEZIFIKATION DES PROBLEMS

Schreiben Sie ein Programm das das Minimum einer Zahlenreihe am Bildschirm ausgibt. Weiterhin soll die Nummer der Zahl, die das Minimum darstellt, ausgegeben werden. Die Anzahl der Zahlen und die Zahlenreihe sollen im Quellcode fest vorgegeben werden.

B: ANALYSE DES PROBLEMS

C: ENTWURF DES PROGRAMMS

BerechneMinimum

Deklariere zahlenliste	
deklariere int min;	
deklariere int pos = 0;	
übergebe zahlenliste	
für (i = 1; i to Länge zahlenliste; i++)	
wahr	zahlenliste[pos] < zahlenliste [min]
	falsch
	
min ← zahlenliste[i]	
pos ← i	
Ausgabe: "Die kleinste Zahl der Liste ist" + min + " und befindet sich an Position" + pos	

D: ERSTELLUNG DES QUELLCODES

```

public static void Minimum(){
    int [] zahlenliste = new int [10];
    int min = 0;
    int pos = 0;

    for (int j = 0; j < zahlenliste.length; j++){
        zahlenliste[j]= (int) (Math.random()* 100);
        System.out.println(zahlenliste[j]);
    }
    System.out.println("");
    for (int i = 1; i < zahlenliste.length; i++){
        if (zahlenliste[i] < zahlenliste[pos]){
            min = zahlenliste[i];
            pos = i;
        }
    }
    System.out.println("Die kleinste Zahl der Liste ist " + min + " und befindet"
        + " sich an Position Nr: " + pos + ".");
}

```

AUFGABE 6: BERECHNUNG DES GGT MIT EINEM WEITEREN ALGORITHMUS

AUFGABE 7: FAKULTÄTEN BERECHNEN

AUFGABE 8: DARSTELLUNG VON e^x DURCH EINE TAYLOR REIHE

AUFGABE 9: DARSTELLUNG VON $\sin(x)$ DURCH EINE TAYLOR REIHE

AUFGABE 10: INSERTION SORT

A: SPEZIFIKATION DES PROBLEMS

B: ANALYSE DES PROBLEMS

C: ENTWURF DES PROGRAMMS

insertionsort

Sortieralgorithmus

Deklariere int [] liste

Deklariere int puffer

Deklariere int i,j

für i \leftarrow 0; to Länge liste; i++

erzeuge Zufallszahl für liste[i]

für i \leftarrow 1; to Länge liste; i++

puffer \leftarrow liste[i]

für j \leftarrow i; j > 0 und liste[j-1] > puffer; j--

liste[j] \leftarrow liste[j-1]

liste[j] \leftarrow puffer

für i \leftarrow 0; to Länge liste; i++

liste[i] ausgeben

D: ERSTELLUNG DES QUELLCODES

```
public static void insertionsort(){
    int [] liste = new int [10];           // Deklaration Variablen
    int puffer = 0;
    int i,j,k;

    for (i = 0; i < liste.length; i++){
        liste[i] = (int) (Math.random() * 100); // Initialisierung Array
        System.out.print(liste[i] + " ");
    }
    for (j = 1; j < liste.length; j++){
        puffer = liste[j];                 // Übergabe des zu sortierenden Wertes in
den puffer
        for(k = j; k > 0 && liste[k-1] > puffer; k--){
            liste[k] = liste[k-1];
        }
        liste[k] = puffer;
    }
    for(i = 0; i < liste.length; i++){
        System.out.println(liste[i]);
    }
}
```