

## Script generated by TTT

Title: groh: profile1 (03.06.2016)

Date: Fri Jun 03 14:10:36 CEST 2016

Duration: 88:27 min

Pages: 42

The screenshot shows the Eclipse IDE interface with the Demo.java file open in the editor. The code defines three methods: `exp`, `power`, and `powerMitwhile`. The `exp` method uses a loop to calculate the result. The `power` and `powerMitwhile` methods both use loops to calculate the result.

```
28     }
29
30     public double exp(double x){
31         double result = 1.0d;
32         for(int n=1; n<10; n++){
33             result = result + power(x, n) / (double)(faculty(n));
34         }
35         return result;
36     }
37
38     public double power(double argument, int m){
39         double result = 1;
40         for(int i=0; i<m; i++){
41             result = result * argument;
42         }
43         return result;
44     }
45
46     public double powerMitwhile(double argument, int m){
47         double result = 1.0d;
48         int i = m;
49         while(i > 0){
50             result = result * argument;
51             i = i - 1;
52         }
53         return result;
54     }

```

The screenshot shows the Eclipse IDE interface with the Demo.java file open in the editor. The code defines two methods: `minimum` and `expectation`. The `minimum` method is annotated with `@Override` and contains logic to find the minimum of two integers. The `expectation` method is also defined.

```
6     /**
7      * @param a
8      * @param b
9      * @return
10     */
11    public int minimum(int a, int b){
12        int result;
13        if(a < b){
14            result = a;
15            //System.out.println("kjxhhjhh");
16            //System.out.println("hhhhh");
17        } else {
18            result = b;
19            return result;
20        }
21    }
22
23    /**
24     * @param a
25     * @param b
26     * @return
27     */
28    public double expectation(){
29        public int minimum(int a, int b){
30            int result;
31            if(a < b){
32                result = a;
33                //System.out.println("kjxhhjhh");
34                //System.out.println("hhhhh");
35            } else {
36                result = b;
37                return result;
38            }
39        }
40    }

```

Screenshot 1: Eclipse IDE showing a Java code editor with a tooltip for a method signature.

The code in Demo.java is:

```
6    public static void main(String[] args) {
7        // TODO Auto-generated method stub
8        int result = 0;
9        Demo demo = new Demo();
10       result = demo.minimum(5, 10*2);
11       System.out.println("das ergebnis ist: " + String.valueOf(result));
12       //long result2 = demo.faculty(-3);
13       //double result2 = demo.powerWithile(2.0d, 3);
14       double result2 = demo.exp(2.0d);
15       System.out.println("das ergebnis ist: " + String.valueOf(result2));
16   }
17
18   public double expectation(double[] a){
19   }
20
21   public int minimum(int a, int b){
22       int result;
23       if(a < b){
24           result = a;
25           //System.out.println("kjxhjhhi");
26           //System.out.println("hhhhh");
27       } else
28           result = b;
29       return result;
30   }
31 }
```

A tooltip is displayed at line 18, column 18, indicating:

This method must return a result of type double  
2 quick fixes available:  
Add return statement  
Change return type to 'void'

Screenshot 2: Eclipse IDE showing the same Java code editor after applying the 'Add return statement' quick fix.

The code in Demo.java is now:

```
6    public static void main(String[] args) {
7        // TODO Auto-generated method stub
8        int result = 0;
9        Demo demo = new Demo();
10       result = demo.minimum(5, 10*2);
11       System.out.println("das ergebnis ist: " + String.valueOf(result));
12       //long result2 = demo.faculty(-3);
13       //double result2 = demo.powerWithile(2.0d, 3);
14       double result2 = demo.exp(2.0d);
15       System.out.println("das ergebnis ist: " + String.valueOf(result2));
16   }
17
18   public double expectation(double[] a){
19       double result = 0.0d;
20   }
21
22   public int minimum(int a, int b){
23       int result;
24       if(a < b){
25           result = a;
26           //System.out.println("kjxhjhhi");
27           //System.out.println("hhhhh");
28       } else
29           result = b;
30   }
31 }
```

Screenshot 3: Eclipse IDE showing the Java code editor with a tooltip for a division operation.

The code in Demo.java is:

```
6    public static void main(String[] args) {
7        // TODO Auto-generated method stub
8        int result = 0;
9        Demo demo = new Demo();
10       result = demo.minimum(5, 10*2);
11       System.out.println("das ergebnis ist: " + String.valueOf(result));
12       //long result2 = demo.faculty(-3);
13       //double result2 = demo.powerWithile(2.0d, 3);
14       double result2 = demo.exp(2.0d);
15       System.out.println("das ergebnis ist: " + String.valueOf(result2));
16   }
17
18   public double expectation(double[] a){
19       double result = 0.0d;
20       double oneOverN = 1.0d / a.length;
21   }
22
23   public int minimum(int a, int b){
24       int result;
25       if(a < b){
26           result = a;
27           //System.out.println("kjxhjhhi");
28           //System.out.println("hhhhh");
29       } else
30           result = b;
31   }
32 }
```

A tooltip is displayed at line 20, column 18, indicating:

No consoles to display at this time.

Screenshot 4: Eclipse IDE showing the Java code editor after applying the 'Add return statement' quick fix from Screenshot 3.

The code in Demo.java is now:

```
6    public static void main(String[] args) {
7        // TODO Auto-generated method stub
8        int result = 0;
9        Demo demo = new Demo();
10       result = demo.minimum(5, 10*2);
11       System.out.println("das ergebnis ist: " + String.valueOf(result));
12       //long result2 = demo.faculty(-3);
13       //double result2 = demo.powerWithile(2.0d, 3);
14       double result2 = demo.exp(2.0d);
15       System.out.println("das ergebnis ist: " + String.valueOf(result2));
16   }
17
18   public double expectation(double[] a){
19       double result = 0.0d;
20       double oneOverN = 1.0d / a.length;
21   }
22
23   public int minimum(int a, int b){
24       int result;
25       if(a < b){
26           result = a;
27           //System.out.println("kjxhjhhi");
28           //System.out.println("hhhhh");
29       } else
30           result = b;
31   }
32 }
```

File Edit Source Refactor Navigate Search Project Run Window Help

Pack... Demo.java

```
6  // TODO Auto-generated method stub
7  int result = 0;
8  Demo demo = new Demo();
9  result = demo.minimum(5, 10*2);
10 System.out.println("das ergebnis ist: " + String.valueOf(result));
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerWithIle(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15
16 }
17
18 public double expectation(double[] a){
19     double result = 0.0d;
20     double oneOverN = 0.0d;
21     if(a.length >= 1)
22         oneOverN = 1.0d / a.length;
23 }
24
25 public int minimum(int a, int b){
26     int result;
27     if(a < b){
28         result = a;
29         //System.out.println("kjxhjhjh");
30         //System.out.println("hhhhh");
31     }
32 }
```

Console

No consoles to display at this time.

Writable Smart Insert 20 : 43

Start

File Edit Source Refactor Navigate Search Project Run Window Help

Pack... Demo.java

```
6  // TODO Auto-generated method stub
7  int result = 0;
8  Demo demo = new Demo();
9  result = demo.minimum(5, 10*2);
10 System.out.println("das ergebnis ist: " + String.valueOf(result));
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerWithIle(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15
16 }
17
18 public double expectation(double[] a){
19     double result = 0.0d;
20     double oneOverN = 0.0d;
21     if(a.length >= 1)
22         oneOverN = 1.0d / a.length;
23     else
24         System.out.println(" das array ist leer");
25     for(int i=0; i < a.length; i++){
26         result = result * a[i];
27     }
28 }
29
30 public int minimum(int a, int b){
31     int result;
32 }
```

Console

No consoles to display at this time.

Writable Smart Insert 26 : 31

File Edit Source Refactor Navigate Search Project Run Window Help

Pack... Demo.java

```
6  // TODO Auto-generated method stub
7  int result = 0;
8  Demo demo = new Demo();
9  result = demo.minimum(5, 10*2);
10 System.out.println("das ergebnis ist: " + String.valueOf(result));
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerWithIle(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15
16 }
17
18 public double expectation(double[] a){
19     double result = 0.0d;
20     double oneOverN = 0.0d;
21     if(a.length >= 1)
22         oneOverN = 1.0d / a.length;
23     else
24         System.out.println(" das array ist leer");
25 }
26
27 public int minimum(int a, int b){
28     int result;
29     if(a < b){
30         result = a;
31     }
32 }
```

Console

No consoles to display at this time.

Writable Smart Insert 24 : 52

Start

File Edit Source Refactor Navigate Search Project Run Window Help

Pack... Demo.java

```
6  // TODO Auto-generated method stub
7  int result = 0;
8  Demo demo = new Demo();
9  result = demo.minimum(5, 10*2);
10 System.out.println("das ergebnis ist: " + String.valueOf(result));
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerWithIle(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15
16 }
17
18 public double expectation(double[] a){
19     double result = 0.0d;
20     double oneOverN = 0.0d;
21     if(a.length >= 1)
22         oneOverN = 1.0d / a.length;
23     else
24         System.out.println(" das array ist leer");
25     for(int i=0; i < a.length; i++){
26         result = result + a[i];
27     }
28     return result * oneOverN;
29 }
30
31 public int minimum(int a, int b){
32 }
```

Console

No consoles to display at this time.

Writable Smart Insert 28 : 33

The screenshot shows the Eclipse IDE interface with the Demo.java file open in the editor. The code implements two methods: variance and expectation. The variance method calculates the sample variance of an array of floats. The expectation method calculates the expected value of an array of floats. Both methods handle empty arrays by printing a message and returning 0.0f. The console output shows the execution of the program and its results.

```

package zue4Wzw;
public class Demo {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int result = 0;
        Demo demo = new Demo();
        result = demo.minimum(5, 10*2);
        System.out.println("das ergebnis ist: " + String.valueOf(result));
        //long result2 = demo.faculty(-3);
        //double result2 = demo.powerWithWhile(2.0d, 3);
        double result2 = demo.exp(2.0d);
        System.out.println("das ergebnis ist: " + String.valueOf(result2));
        double[] theArray = new double[3];
        theArray[0] = 1.0d;
        theArray[1] = 2.0d;
        theArray[2] = 3.0d;
        double result3 = demo.expectation(theArray);
        System.out.println("das ergebnis ist: " + String.valueOf(result3));
    }
    public double expectation(double[] a){
        double result = 0.0d;
        double oneOverN = 0.0d;
        if(a.length >= 1)
            oneOverN = 1.0d / a.length;
        ...
    }
}

```

```

<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 14:33:12)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0

```

## Aufgabe

```

public class Demo {

    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=1)
            oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
        else
            System.out.println("array is empty");
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1)
            oneOverN = 1.0f / (float)(a.length);
        else
            System.out.println("array is empty");
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
}

```

106

## Aufgabe

```

public class Demo {

    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=1)
            oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
        else
            System.out.println("array is empty");
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1)
            oneOverN = 1.0f / (float)(a.length);
        else
            System.out.println("array is empty");
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
}

```

## Aufgabe

```

public class Demo {

    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=1)
            oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
        else
            System.out.println("array is empty");
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1)
            oneOverN = 1.0f / (float)(a.length);
        else
            System.out.println("array is empty");
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
}

```

106

## Aufgabe

public class Demo {

```
    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=2)
            oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
        else
            System.out.println("array is not long enough");
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1)
            oneOverN = 1.0f / (float)(a.length);
        else
            System.out.println("array is empty");
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
```



## Aufgabe

public class Demo {

```
    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=2)
            oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
        else
            System.out.println("array is not long enough");
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1)
            oneOverN = 1.0f / (float)(a.length);
        else
            System.out.println("array is empty");
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
```



106

## Aufgabe

public class Demo {

```
    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=2)
            oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
        else
            System.out.println("array is not long enough");
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1)
```



```
            oneOverN = 1.0f / (float)(a.length);
        else
            System.out.println("array is empty");
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
```

}

...

## Aufgabe

public class Demo {

```
    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=2)
            oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
        else
            System.out.println("array is not long enough");
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1)
```



```
            oneOverN = 1.0f / (float)(a.length);
        else
            System.out.println("array is empty");
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
```

}

...

106

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
18     double oneOverNMinusOne = 0.0d;
19     if(a.length >=2)
20         oneoverNMinusOne = 1.0f / ((float)(a.length) - 1);
21     else
22         System.out.println("array is not long enough");
23     float expectation = expectation(a);
24     for(int i=0; i<a.length; i++){
25         result = result + (a[i] - expectation) * (a[i] - expectation);
26     }
27     result = oneOverNMinusOne * result;
28     return result;
29 }
30
31
32 public double expectation(double[] a){
33     double result = 0.0d;
34     double oneOverN = 0.0d;
35     if(a.length >= 1)
36         oneOverN = 1.0d / a.length;
37     else
38         System.out.println(" das array ist leer");
39     for(int i=0; i < a.length; i++){
40         result = result + a[i];
41     }
42     return result * oneOverN;
43 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 14:33:12)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
```

Writable SmartInsert 37: 1

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
4
5    public static void main(String[] args) {
6        // TODO Auto-generated method stub
7        int result = 0;
8        Demo demo = new Demo();
9        result = demo.minimum(5, 10*2);
10       System.out.println("das ergebnis ist: " + String.valueOf(result));
11       //long result2 = demo.factury(-3);
12       //double result2 = demo.powerHitWhile(2.0d, 3);
13       double result2 = demo.exp(2.0d);
14       System.out.println("das ergebnis ist: " + String.valueOf(result2));
15       double[] theArray = new double[3];
16       theArray[0] = 1.0d;
17       theArray[1] = 2.0d;
18       theArray[2] = 3.0d;
19       double result3 = demo.expectation(theArray);
20       System.out.println("das ergebnis ist: " + String.valueOf(result3));
21   }
22
23
24 public double variance(double []a){
25     double result = 0.0d;
26     double oneOverNMinusOne = 0.0d;
27     if(a.length >2)
28         oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
29     else
30         System.out.println("array is not long enough");
31     double exoectation = expectation(a);
32 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 14:33:12)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
```

Writable SmartInsert 23: 35

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
1/
2     theArray[1] = 2.0d;
3     theArray[2] = 3.0d;
4     double result3 = demo.expectation(theArray);
5     System.out.println("das ergebnis ist: " + String.valueOf(result3));
6 }
7
8 public double variance(double []a){
9     double result = 0.0d;
10    double oneOverNMinusOne = 0.0d;
11    if(a.length >2)
12        oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
13    else
14        System.out.println("array is not long enough");
15    double expectation = expectation(a);
16    for(int i=0; i<a.length; i++){
17        result = result + (a[i] - expectation) * (a[i] - expectation);
18    }
19    result = oneOverNMinusOne * result;
20    return result;
21 }
22
23
24 public double expectation(double[] a){
25     double result = 0.0d;
26     double oneOverN = 0.0d;
27     if(a.length >= 1)
28         oneOverN = 1.0d / a.length;
29 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 14:33:12)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
```

Writable SmartInsert 25: 15

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
4
5    public static void main(String[] args) {
6        // TODO Auto-generated method stub
7        int result = 0;
8        Demo demo = new Demo();
9        result = demo.minimum(5, 10*2);
10       System.out.println("das ergebnis ist: " + String.valueOf(result));
11       //long result2 = demo.factury(-3);
12       //double result2 = demo.powerHitWhile(2.0d, 3);
13       double result2 = demo.exp(2.0d);
14       System.out.println("das ergebnis ist: " + String.valueOf(result2));
15       double[] theArray = new double[3];
16       theArray[0] = 1.0d;
17       theArray[1] = 2.0d;
18       theArray[2] = 3.0d;
19       double result3 = demo.expectation(theArray);
20       System.out.println("das ergebnis ist: " + String.valueOf(result3));
21   }
22
23
24 public double variance(double []a){
25     double result = 0.0d;
26     double oneOverNMinusOne = 0.0d;
27     if(a.length >2)
28         oneOverNMinusOne = 1.0f / ((float)(a.length) - 1);
29     else
30         System.out.println("array is not long enough");
31     double expectation = expectation(a);
32 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 14:33:12)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
```

Writable SmartInsert 20: 1

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer (left):** Shows a package named "fasel" containing several Java files like "studivNew", "zue1Wzw", "zue2Wzw", "zue3Wzw", and "zue4Wzw".
- Editor (center-left):** Displays the content of the file "Demo.java". The code implements a class "Demo" with methods for calculating minimum, power, expectation, and variance of an array of doubles.
- Console (bottom):** Shows the terminal output of the Java application. It prints three lines of text: "das ergebnis ist: 5", "das ergebnis ist: 7.3887125220458545", and "das ergebnis ist: 2.0".
- Bottom Status Bar:** Shows the status "Writable", "Smart Insert", and the current time "21:39".
- Bottom Taskbar:** Includes icons for Start, File, Edit, Source, Refactor, Search, Project, Run, Window, Help, Quick Access, Java, and Debug.

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse
- Toolbar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Quick Access:** Buttons for Java and Debug.
- Left Sidebar:** Package Explorer showing projects: fasel, studiuNew, zue1Wzw, zue2Wzw, zue3Wzw, and zue4Wzw.
- Central Area:** Demo.java code editor with the following content:

```
4
5  public static void main(String[] args) {
6      // TODO Auto-generated method stub
7      int result = 0;
8      Demo demo = new Demo();
9      result = demo.minimum(5, 10*2);
10     System.out.println("das ergebnis ist: " + String.valueOf(result));
11     //long result2 = demo.factor(-3);
12     //double result2 = demo.powerNthWhile(2.0d, 3);
13     double result2 = demo.exp(2.0d);
14     System.out.println("das ergebnis ist: " + String.valueOf(result2));
15     double[] theArray = new double[3];
16     theArray[0] = 1.0d;
17     theArray[1] = 2.0d;
18     theArray[2] = 3.0d;
19     double result3 = demo.expectation(theArray);
20     System.out.println("das ergebnis ist: " + String.valueOf(result3));
21     double result4 = demo.variance(theArray);
22     System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 }
24
25 public double variance(double a[]){
26     double result = 0.0d;
27     double oneOverMMinusOne = 0.0d;
28     if(a.length >=2)
29         oneOverMMinusOne = 1.0d / ((double)(a.length) - 1);
30     else
31 }
```

- Bottom Area:** Console tab showing the output of the Java application:

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse
- Menu Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help
- Toolbar:** Includes icons for New, Open, Save, Cut, Copy, Paste, Find, Replace, and others.
- Quick Access:** A button labeled "Quick Access" with a magnifying glass icon.
- Java View:** Shows the package structure:
  - fasel
  - studivzNew
  - zue1Wzw
  - zue2Wzw
  - zue3Wzw
  - zue4Wzw
- Code Editor:** The file `Demo.java` is open, displaying Java code. A tooltip is visible over the `expectation` variable in the `variance` method.

```
11     //long result2 = demo.faculty(-3);
12     //double result2 = demo.powerNWithWhile(2.0d, 3);
13     double result2 = demo.exp(2.0d);
14     System.out.println("das ergebnis ist: " + String.valueOf(result2));
15     double[] theArray = new double[3];
16     theArray[0] = 1.0d;
17     theArray[1] = 2.0d;
18     theArray[2] = 3.0d;
19     double result3 = demo.expectation(theArray);
20     System.out.println("das ergebnis ist: " + String.valueOf(result3));
21     double result4 = demo.variance(theArray);
22     System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 }
24
25 public double variance(double a[]){
26     double result = 0.0d;
27     double oneOverNMinusOne = 0.0d;
28     if(a.length >=2)
29         oneOverNMinusOne = 1.0d / ((double)(a.length) - 1);
30     else
31         System.out.println("array is not long enough");
32     double expectation = expectation(a);
33     for(int i=0; i<a.length; i++){
34         result = result + (a[i] - expectation) * (a[i] - expectation);
35     }
36     result = oneOverNMinusOne * result;
37     return result;
38 }
```
- Console:** Shows the output of the application:

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```
- Status Bar:** Shows Writable, SmartInsert, 34 : 50, and battery level at 99%.
- System Tray:** Shows icons for Start, Task View, Internet Explorer, and others.

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse
- Menu Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help
- Toolbar:** Includes icons for New, Open, Save, Cut, Copy, Paste, Find, Select All, etc.
- Project Explorer:** Shows packages like Packa..., fasel, studienvNew, zue1Wzw, zue2Wzw, zue3Wzw, and zue4Wzw.
- Code Editor:** Displays the content of Demo.java. The code calculates various statistical measures (faculty, power, exp, variance) and prints them to the console.
- Console:** Shows the terminal output of the application's execution. The output is:

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_32\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```
- Status Bar:** Shows Writable, Smart Insert, 34 : 50, DE, 99%, and system icons.

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNthWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```

Writable SmartInsert 23 : 46

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNthWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```

Writable SmartInsert 23 : 17

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNthWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```

Writable SmartInsert 23 : 46

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Packa... Demo.java

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNthWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 }
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```

Writable SmartInsert 25 : 6

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNWithWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 System.out.println("das ergebnis ist: " + String.valueOf(result4));
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:04:29)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
```

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNWithWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 System.out.println("das ergebnis ist: " + String.valueOf(matrix[2][2]));
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:17:02)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
das ergebnis ist: 0.0
```

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNWithWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 System.out.println("das ergebnis ist: " + String.valueOf(matrix[2][2]));
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:17:02)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
das ergebnis ist: 0.0
```

Java - zue4Wzw/src/zue4Wzw/Demo.java - Eclipse

```
11 //long result2 = demo.faculty(-3);
12 //double result2 = demo.powerNWithWhile(2.0d, 3);
13 double result2 = demo.exp(2.0d);
14 System.out.println("das ergebnis ist: " + String.valueOf(result2));
15 double[] theArray = new double[3];
16 theArray[0] = 1.0d;
17 theArray[1] = 2.0d;
18 theArray[2] = 3.0d;
19 double result3 = demo.expectation(theArray);
20 System.out.println("das ergebnis ist: " + String.valueOf(result3));
21 double result4 = demo.variance(theArray);
22 System.out.println("das ergebnis ist: " + String.valueOf(result4));
23 double[][] matrix = new double[3][3];
24 System.out.println("das ergebnis ist: " + String.valueOf(matrix[2][2]));
```

Console

```
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (03.06.2016, 15:17:02)
das ergebnis ist: 5
das ergebnis ist: 7.3887125220458545
das ergebnis ist: 2.0
das ergebnis ist: 1.0
das ergebnis ist: 0.0
```

The screenshot shows a Microsoft PowerPoint slide titled "Aufgabe". The slide contains the following Java code:

```

public class Demo {
    public float variance(float a[]){
        float result = 0.0f;
        float oneOverNMinusOne = 0.0f;
        if(a.length >=2) {
            oneOverNMinusOne = 1.0f / ((float)a.length - 1);
        } else {
            System.out.println("array is not long enough");
        }
        float expectation = expectation(a);
        for(int i=0; i<a.length; i++){
            result = result + (a[i] - expectation) * (a[i] - expectation);
        }
        result = oneOverNMinusOne * result;
        return result;
    }

    public float expectation(float a[]){
        float result = 0.0f;
        float oneOverN = 0.0f;
        if(a.length >=1) {
            oneOverN = 1.0f / ((float)a.length);
        } else {
            System.out.println("array is empty");
        }
        for(int i=0; i<a.length; i++){
            result = result + a[i];
        }
        result = oneOverN * result;
        return result;
    }
}

```

Below the code, there is a note: "Klicken Sie, um Notizen hinzuzufügen". The status bar at the bottom shows "FOLIE 106 VON 196 ENGLISCH (USA)" and the date "03.06.2016".

## Aufgabe

Gegeben sei folgender Java Code:

```

public static void demo(){
    int multiplier = 10;
    int[][] array = new int[10][10];
    for(int i=0; i<10; i++){
        for(int j=0; j<10; j++){
            array[i][j] = j * multiplier;
        }
    }
    multiplier = multiplier * 10;
}
System.out.println(array[2][2]);
//  

boolean a = true;
boolean b = false;
boolean c = a || b; //logical or
if(a && b) //logical and
    c = c && true;
else
    c = c && false;
System.out.println(c);
}

```

Welche Ausgabe produziert demo?  
BEGRÜNDEN Sie jeweils ganz kurz!

108

## Aufgabe

Gegeben sei folgender Java Code:

```

public static void demo(){
    int multiplier = 10;
    int[][] array = new int[10][10];
    for(int i=0; i<10; i++){
        for(int j=0; j<10; j++){
            array[i][j] = j * multiplier;
        }
    }
    multiplier = multiplier * 10;
}
System.out.println(array[2][2]);
//  

boolean a = true;
boolean b = false;
boolean c = a || b; //logical or
if(a && b) //logical and
    c = c && true;
else
    c = c && false;
System.out.println(c);
}

```

Welche Ausgabe produziert demo?  
BEGRÜNDEN Sie jeweils ganz kurz!

## Aufgabe

Gegeben sei folgender Java Code:

```

public static void demo(){
    int multiplier = 10;
    int[][] array = new int[10][10];
    for(int i=0; i<10; i++){
        for(int j=0; j<10; j++){
            array[i][j] = j * multiplier;
        }
    }
    multiplier = multiplier * 10;
}
System.out.println(array[2][2]);
//  

boolean a = true;
boolean b = false;
boolean c = a || b; //logical or
if(a && b) //logical and
    c = c && true;
else
    c = c && false;
System.out.println(c);
}

```

Welche Ausgabe produziert demo?  
BEGRÜNDEN Sie jeweils ganz kurz!

108

## Aufgabe

Gegeben sei folgender Java Code:

```
public static void demo(){
    int multiplier = 10;
    int[][] array = new int[10][10];
    for(int i=0; i<10; i++){
        for(int j=0; j<10; j++){
            array[i][j] = j * multiplier;
        }
        multiplier = multiplier * 10;
    }
    System.out.println(array[2][2]);
    //
    boolean a = true;
    boolean b = false;
    boolean c = a || b; //logical or
    if(a && b) //logical and
        c = c && true;
    else
        c = c && false;
    System.out.println(c);
}
```

Welche Ausgabe produziert demo?

BEGRÜNDEN Sie jeweils ganz kurz!

## Aufgabe

Gegeben sei folgender Java Code:

```
public static void demo(){
    int multiplier = 10;
    int[][] array = new int[10][10];
    for(int i=0; i<10; i++){
        for(int j=0; j<10; j++){
            array[i][j] = j * multiplier;
        }
        multiplier = multiplier * 10;
    }
    System.out.println(array[2][2]);
    //
    boolean a = true;
    boolean b = false;
    boolean c = a || b; //logical or
    if(a && b) //logical and
        c = c && true;
    else
        c = c && false;
    System.out.println(c);
}
```

Welche Ausgabe produziert demo?

BEGRÜNDEN Sie jeweils ganz kurz!

108

108

## Aufgabe

Gegeben seien zwei Java Methoden, die die Funktion  $f(x) = x^n$  berechnen:

```
public static double powerWithWhile(double x, int n)
    double result = 1.0;
    int i = 1;
    while (i <= n) {
        result = [ ];
        i++;
    }
    return result;
}

public static double powerWithFor(double x, int n)
    double result = 1.0;
    for ([ ]) {
        result = result * x;
    }
    return result;
}
```

Ergänzen Sie die fehlenden Elemente sinnvoll!

112