

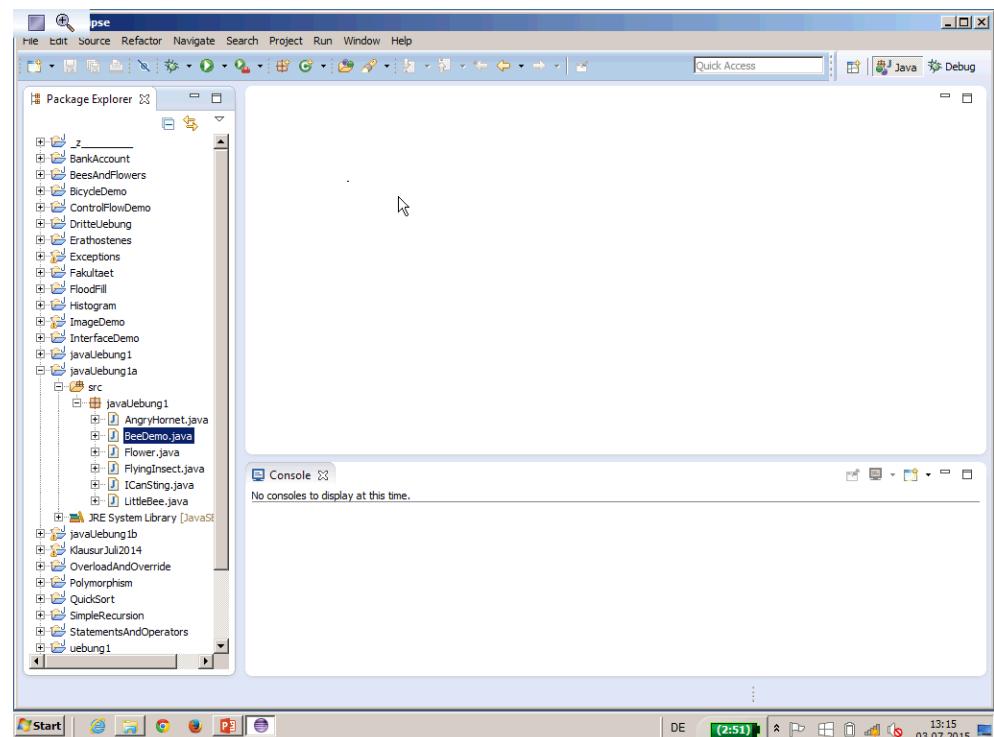
Script generated by TTT

Title: groh: profile1 (03.07.2015)

Date: Fri Jul 03 13:15:20 CEST 2015

Duration: 101:01 min

Pages: 148



The screenshot shows a Microsoft PowerPoint slide titled "Access Modifiers, Packages". The slide content includes:

- A bulleted list: "● Access Modifier für Methoden und Attribute:"
- A table comparing access modifiers across different scopes:

| | Class | Package | Subclasses | World |
|-------------|-------|---------|------------|-------|
| public | ✓ | ✓ | ✓ | ✓ |
| protected | ✓ | ✓ | ✓ | |
| no modifier | ✓ | ✓ | | |
| private | ✓ | | | |
- A bulleted list: "● Packages:
 - Kapseln (hierarchisch organisiert) Mengen von Klassen und Interfaces
 - Deklaration: package *nameOfPackage*;
 - Beispiele: `java.math`, `java.lang`, `java.net`, `de.tum.in`

The slide has a footer indicating "FOLIE 129 VON 176 ENGLISCH USA" and "DE (2:37) 03.07.2015".

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Modifiers: static und final

- **static:**
 - Methode oder Attribut gehört zur Klasse und nicht zum Objekt
(Attribut: existiert nur einmal, ist für alle Objekte dasselbe)
„Klassenmethode, Klassenattribut“ <-> „Instanzenmethode, Instanzenattribut“
- **final:**
 - für Attribute: können nicht mehr geändert werden (**Konstanten**)
 - für Methoden: können nicht overridden oder hidden werden (kommt gleich)
 - für Klassen: Es können **keine Unterklassen** davon abgeleitet werden.

```
final class MyClass {
    static int sameForAllInstances = 3;
    final int constantMayBeDifferentForEachInstance;
    static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

    static void methodOne() { /* ... */ }
    final void methodTwo() { /* ... */ }
    static final void methodThree() { /* ... */ }
    void methodFour() { /* ... */ }
}
```

Java.pptx - PowerPoint

DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHEINPRÄSENTATION ÜBERPRÜFEN ANSICHT Anmelde

Einfügen
Neue Folie F K U \$ abc AV Aa A Formen Anordnen Schnellformatvorlagen Zeichnung Bearbeiten

Zwischenablage Folienschriftart Absatz Zeichnung Bearbeiten

118 

119 

120 

121 

122 

Overloading

Overloading: In einer Klasse mehrere Methoden mit **gleichem Namen**, aber **verschiedener Parameterliste**:

```
class OverloadingDemoClass {
    public int doSomething() {
        return 1 + 1;
    }

    public int doSomething(int param) {
        return param + 2;
    }
}

OverloadingDemoClass odc = new OverloadingDemoClass();
int result1 = odc.doSomething();
int result2 = odc.doSomething(33);
```

Sinn: Flexibilität (speziellere und weniger spezielle Varianten der Methode anbieten, Abstraktion (-> APIs):

DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHEIRMÄRINTATION ÜBERPRÜFEN ANSICHT

Einfügen Folien Schriftart Absatz Formen Anordnen Schnellformatvorlagen Zeichnung Bearbeiten

Suchen S/AC Ersetzen Markieren

118

119

120

121

122

Overloading

Overloading: In einer Klasse mehrere Methoden mit gleichem Namen, aber verschiedener Parameterliste:

```
class OverloadingDemoClass {
    public int doSomething() {
        return 1 + 1;
    }

    public int doSomething(int param) {
        return param + 2;
    }
}

OverloadingDemoClass odc = new OverloadingDemoClass();
int result1 = odc.doSomething();
int result2 = odc.doSomething(33);
```

Sinn: Flexibilität (speziellere und weniger spezielle Varianten der Methode anbieten, Abstraktion (→ APIs):

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

- Title Bar:** aUebung1a/src/javaUebung1/Flower.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, and others.
- Quick Access Bar:** Contains icons for Java and Debug.
- Package Explorer:** Shows the project structure with packages like BankAccount, BeesAndFlowers, BicycleDemo, ControlFlowDemo, DritteUebung, Eratosthenes, Exceptions, Fakultaet, Floodfill, Histogram, ImageDemo, InterfaceDemo, javaUebung1, and javaUebung1a. Under javaUebung1a/src, there are files: AngryHornet.java, BeeDemo.java, Flower.java, FlyingInsect.java, IcarSting.java, and LittleBee.java.
- Editor:** Displays the code for Flower.java:

```
1 package javaUebung1;
2
3 public class Flower {
4
5     public double amountOfPollen = 100.0d;
6
7     public double getAmountOfPollen(){
8         System.out.println("lalalalalalla = " + amountOfPollen);
9         return amountOfPollen;
10    }
11
12    public double harvestPollen(double howMuch){
13        double returnedAmountOfPollen;
14        if(howMuch > amountOfPollen){
15            returnedAmountOfPollen = amountOfPollen;
16            amountOfPollen = 0.0d;
17        }
18        else {
19            returnedAmountOfPollen = howMuch;
20            amountOfPollen = amountOfPollen - howMuch;
21        }
22        return returnedAmountOfPollen;
23    }
24
25 }
```
- Console:** Shows the message: No consoles to display at this time.
- Bottom Status Bar:** Writable, Smart Insert, 5:11, 13:23, 03.07.2015, and various system icons.

aUebung1/src/javaUebung1/Flower.java - Eclipse

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

Quick Access Java Debug

Package Explorer

```
1 package javaUebung1;
2
3 public class Flower {
4
5     public double amountOfPollen = 100.0d;
6
7     public double getAmountOfPollen(){
8         System.out.println("lalalalalalla = " + amountOfPollen);
9         return amountOfPollen;
10    }
11
12    public double harvestPollen(double howMuch){
13        double returnedAmountOfPollen;
14        if(howMuch > amountOfPollen){
15            returnedAmountOfPollen = amountOfPollen;
16            amountOfPollen = 0.0d;
17        }
18        else {
19            returnedAmountOfPollen = howMuch;
20            amountOfPollen = amountOfPollen - howMuch;
21        }
22        return returnedAmountOfPollen;
23    }
24
25 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 5:11

Start DE (2:38) 13:23 03.07.2015

aUebung1/src/javaUebung1/Flower.java - Eclipse

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

Quick Access Java Debug

Package Explorer

```
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen = 100.0d;
6
7     public double getAmountOfPollen(){
8         return amountOfPollen;
9     }
10
11    public double harvestPollen(double howMuch){
12        double returnedAmountOfPollen;
13        if(howMuch > amountOfPollen){
14            returnedAmountOfPollen = amountOfPollen;
15            amountOfPollen = 0.0d;
16        }
17        else {
18            returnedAmountOfPollen = howMuch;
19            amountOfPollen = amountOfPollen - howMuch;
20        }
21        return returnedAmountOfPollen;
22    }
23
24 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 8:1

Start DE (2:38) 13:24 03.07.2015

aUebung1/src/javaUebung1/Flower.java - Eclipse

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

Quick Access Java Debug

Package Explorer

```
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen = 100.0d;
6
7     public double getAmountOfPollen(){
8         return amountOfPollen;
9     }
10
11    public double harvestPollen(double howMuch){
12        double returnedAmountOfPollen;
13        if(howMuch > amountOfPollen){
14            returnedAmountOfPollen = amountOfPollen;
15            amountOfPollen = 0.0d;
16        }
17        else {
18            returnedAmountOfPollen = howMuch;
19            amountOfPollen = amountOfPollen - howMuch;
20        }
21        return returnedAmountOfPollen;
22    }
23
24 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 8:1

Start DE (2:38) 13:24 03.07.2015

aUebung1/src/javaUebung1/Flower.java - Eclipse

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

Quick Access Java Debug

Package Explorer

```
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen = 100.0d;
6
7     public double getAmountOfPollen(){
8         return amountOfPollen;
9     }
10
11    public double harvestPollen(double howMuch){
12        double returnedAmountOfPollen;
13        if(howMuch > amountOfPollen){
14            returnedAmountOfPollen = amountOfPollen;
15            amountOfPollen = 0.0d;
16        }
17        else {
18            returnedAmountOfPollen = howMuch;
19            amountOfPollen = amountOfPollen - howMuch;
20        }
21        return returnedAmountOfPollen;
22    }
23
24 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 5:34

Start DE (2:41) 13:25 03.07.2015

aUebung1/src/javaUebung1/Flower.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
*Flower.java
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen;
6
7     public double getAmountOfPollen(){
8         return amountOfPollen;
9     }
10
11    public double harvestPollen(double howMuch){
12        double returnedAmountOfPollen;
13        if(howMuch > amountOfPollen){
14            returnedAmountOfPollen = amountOfPollen;
15            amountOfPollen = 0.0d;
16        } else {
17            returnedAmountOfPollen = howMuch;
18            amountOfPollen = amountOfPollen - howMuch;
19        }
20        return returnedAmountOfPollen;
21    }
22
23    public double getAmountOfPollen(){
24        return amountOfPollen;
25    }
}
Console
No consoles to display at this time.
Writable Smart Insert 5:34 13:25 03.07.2015
```

aUebung1/src/javaUebung1/Flower.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
*Flower.java
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen;
6
7     public Flower(){
8         amountOfPollen = 100.0d;
9     }
10
11    public double getAmountOfPollen(){
12        return amountOfPollen;
13    }
14
15    public double harvestPollen(double howMuch){
16        double returnedAmountOfPollen;
17        if(howMuch > amountOfPollen){
18            returnedAmountOfPollen = amountOfPollen;
19            amountOfPollen = 0.0d;
20        } else {
21            returnedAmountOfPollen = howMuch;
22            amountOfPollen = amountOfPollen - howMuch;
23        }
24    }
25
}
Console
No consoles to display at this time.
Writable Smart Insert 8:12 13:26 03.07.2015
```

aUebung1/src/javaUebung1/Flower.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
*Flower.java
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen;
6     private int numberOfPetals;
7
8     public Flower(){
9         amountOfPollen = 100.0d;
10    }
11
12    public double getAmountOfPollen(){
13        return amountOfPollen;
14    }
15
16    public double harvestPollen(double howMuch){
17        double returnedAmountOfPollen;
18        if(howMuch > amountOfPollen){
19            returnedAmountOfPollen = amountOfPollen;
20            amountOfPollen = 0.0d;
21        } else {
22            returnedAmountOfPollen = howMuch;
23            amountOfPollen = amountOfPollen - howMuch;
24        }
25    }
}
Console
No consoles to display at this time.
Writable Smart Insert 8:12 13:26 03.07.2015
```

aUebung1/src/javaUebung1/Flower.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
*Flower.java
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen;
6     private int numberOfPetals;
7
8     public Flower(){
9         amountOfPollen = 100.0d;
10        numberOfPetals = 5;
11        //...
12    }
13
14    public double getAmountOfPollen(){
15        return amountOfPollen;
16    }
17
18    public double harvestPollen(double howMuch){
19        double returnedAmountOfPollen;
20        if(howMuch > amountOfPollen){
21            returnedAmountOfPollen = amountOfPollen;
22            amountOfPollen = 0.0d;
23        } else {
24            returnedAmountOfPollen = howMuch;
25            amountOfPollen = amountOfPollen - howMuch;
26        }
27    }
}
Console
No consoles to display at this time.
Writable Smart Insert 10:23 13:27 03.07.2015
```

aUebung1/src/javaUebung1/Flower.java - Eclipse

```
13
14     public Flower(double amount){
15         amountOfPollen = amount;
16         numberOFpetals = 5;
17     }
18
19     public double getAmountOfPollen(){
20         return amountOfPollen;
21     }
22
23     public double harvestPollen(double howMuch){
24         double returnedAmountOfPollen;
25         if(howMuch > amountOfPollen){
26             returnedAmountOfPollen = amountOfPollen;
27             amountOfPollen = 0.0d;
28         }
29         else{
30             returnedAmountOfPollen = howMuch;
31             amountOfPollen = amountOfPollen - howMuch;
32         }
33         return returnedAmountOfPollen;
34     }
35
36 }
```

Console

```
No consoles to display at this time.
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
13
14     public static void main(String[] args) {
15         Flower wurscht;
16         wurscht = new Flower();
17         Flower weisswurscht = new Flower();
18         LittleBee maja = new LittleBee();
19         LittleBee willi = new LittleBee();
20         maja.collectPollen(wurscht);
21         System.out.println(maja.collectedPollen);
22         wurscht.getAmountOfPollen();
23         willi.snooze();
24     }
25
26 }
```

Console

```
No consoles to display at this time.
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly; I
7         wurscht = new Flower();
8         Flower weisswurscht = new Flower();
9         LittleBee maja = new LittleBee();
10        LittleBee willi = new LittleBee();
11        maja.collectPollen(wurscht);
12        maja.collectPollen(weisswurscht);
13        System.out.println(maja.collectedPollen);
14        wurscht.getAmountOfPollen();
15        willi.snooze();
16    }
17
18 }
19
20 }
```

Find/Replace

Find: ivSlow
Replace with:
Direction: Foward
Scope: All
Options:
 Case sensitive Wrap search
 Whole word Incremental
 Regular expressions

Console

```
No consoles to display at this time.
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly; I
7         lilly = new Flower();
8         Flower weisslilly = new Flower();
9         LittleBee maja = new LittleBee();
10        LittleBee willi = new LittleBee(); I
11        maja.collectPollen(lilly);
12        maja.collectPollen(weisslilly);
13        System.out.println(maja.collectedPollen);
14        lilly.getAmountOfPollen();
15        willi.snooze();
16    }
17
18 }
19
20 }
```

Console

```
No consoles to display at this time.
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower();
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.collectedPollen());
13        System.out.println(lilly.amountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16    }
17
18 }
19
20 }
```

Console

No consoles to display at this time.

Writable Smart Insert 9:33

aUebung1/src/javaUebung1/Flower.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java *BeeDemo.java
1 package javaUebung1;
2
3 public class Flower {
4
5     private double amountOfPollen;
6     private int numberofPetals;
7
8     public Flower(){
9         amountOfPollen = 100.0d;
10        numberofPetals = 5;
11        //...
12    }
13
14    public Flower(double amount){
15        amountOfPollen = amount;
16        numberofPetals = 5;
17    }
18
19    public double getAmountOfPollen(){
20        return amountOfPollen;
21    }
22
23    public double harvestPollen(double howMuch){
24        double returnedAmountOfPollen;
25        if(howMuch > amountOfPollen){
26            returnedAmountOfPollen = amountOfPollen;
27            amountOfPollen = 0.0d;
28        } else {
29            returnedAmountOfPollen = howMuch;
30        }
31        return returnedAmountOfPollen;
32    }
33
34 }
```

Console

No consoles to display at this time.

Writable Smart Insert 9:33

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.collectedPollen());
13        System.out.println(lilly.amountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16    }
17
18 }
19
20 }
```

Console

No consoles to display at this time.

Writable Smart Insert 7:38

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java BeeDemo.java LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSing{
4
5     public double collectedPollen = 0.0;
6
7     void collectPollen(Flower f){
8         double amount = f.harvestPollen(10.0);
9         collectedPollen = collectedPollen + amount;
10        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
11    }
12
13    void snooze(){
14        System.out.print("schnarch!");
15    }
16
17    public void sting(){
18        System.out.println("pieks!");
19    }
20
21 }
```

Console

No consoles to display at this time.

Writable Smart Insert 1:1

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.collectedPollen());
13        lilly.getAmountOfPollen();
14        willi.snooze();
15    }
16
17 }
18
19 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 13 : 53

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSing{
4
5     private double collectedPollen = 0.0;
6
7     public double getCollectedPollen(){
8         return collectedPollen;
9     }
10
11     void collectPollen(Flower f){
12         double amount = f.harvestPollen(10.0);
13         collectedPollen = collectedPollen + amount;
14         System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
15     }
16
17     void snooze(){
18         System.out.print("schnarch!");
19     }
20
21     public void sting(){
22         System.out.println("pieks!");
23     }
24 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 10 : 5

aUebung1/src/javaUebung1/Flower.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java
1 package javaUebung1;
2
3 public class Flower {
4
5     double amountOfPollen = 0;
6
7     public double getAmountOfPollen(){
8         return amountOfPollen;
9     }
10
11     public double harvestPollen(double howMuch){
12         double returnedAmountOfPollen;
13         if(howMuch > amountOfPollen){
14             returnedAmountOfPollen = amountOfPollen;
15             amountOfPollen = 0.0d;
16         }
17         else {
18             returnedAmountOfPollen = howMuch;
19             amountOfPollen = amountOfPollen - howMuch;
20         }
21         return returnedAmountOfPollen;
22     }
23
24     public double harvestPollen(){
25         double result = harvestPollen(50.0d);
26         return result;
27     }
28
29 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 19 : 38

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSing{
4
5     private double collectedPollen = 0.0;
6
7     public double getCollectedPollen(){
8         return collectedPollen;
9     }
10
11     void collectPollen(Flower f){
12         double amount = f.harvestPollen();
13         collectedPollen = collectedPollen + amount;
14         System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
15     }
16
17     void snooze(){
18         System.out.print("schnarch!");
19     }
20
21     public void sting(){
22         System.out.println("pieks!");
23     }
24 }
```

Console

```
No consoles to display at this time.
```

Writable Smart Insert 12 : 40

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSing{
4
5     private double collectedPollen = 0.0;
6
7     public double getCollectedPollen(){
8         return collectedPollen;
9     }
10
11    void collectPollen(Flower f){
12        double amount = f.harvestPollen();
13        collectedPollen = collectedPollen + amount;
14        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
15    }
16
17    void collectPollen(Flower f, double howMuch){
18        double amount = f.harvestPollen(howMuch);
19        collectedPollen = collectedPollen + amount;
20        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
21    }
22
23    void snooze(){
24        System.out.print("schnarch!");
25    }

```

Console

No consoles to display at this time.

Writable Smart Insert 18 : 48

aUebung1/src/javaUebung1/FlyingInsect.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 class FlyingInsect {
4
5     int weight;
6
7     void flySlow(){
8         System.out.println("oooooooooooooo");
9     }
10
11 }

```

Console

No consoles to display at this time.

javaUebung1.FlyingInsect.java - javaUebung1/src

Start DE 13:40 03.07.2015

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSing{
4
5     private double collectedPollen = 0.0;
6
7     public double getCollectedPollen(){
8         return collectedPollen;
9     }
10
11    void collectPollen(Flower f){
12        double amount = f.harvestPollen();
13        collectedPollen = collectedPollen + amount;
14        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
15    }
16
17    void collectPollen(Flower f, double howMuch){
18        double amount = f.harvestPollen(howMuch);
19        collectedPollen = collectedPollen + amount;
20        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
21    }
22
23    void snooze(){
24        System.out.print("schnarch!");
25    }

```

Console

No consoles to display at this time.

Writable Smart Insert 3 : 44

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSing{
4
5     private double collectedPollen = 0.0;
6
7     public double getCollectedPollen(){
8         return collectedPollen;
9     }
10
11    void collectPollen(Flower f){
12        double amount = f.harvestPollen();
13        collectedPollen = collectedPollen + amount;
14        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
15    }
16
17    void collectPollen(Flower f, double howMuch){
18        double amount = f.harvestPollen(howMuch);
19        collectedPollen = collectedPollen + amount;
20        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
21    }
22
23    void snooze(){
24        System.out.print("schnarch!");
25    }
26
27    public void sting(){
28        System.out.println("pieks!");
29    }
30
31 }

```

Console

No consoles to display at this time.

javaUebung1.LittleBee.java - javaUebung1/src

Start DE 13:41 03.07.2015

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java
8     return collectedPollen;
9 }
10
11 void collectPollen(Flower f){
12     double amount = f.harvestPollen();
13     collectedPollen = collectedPollen + amount;
14     System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
15 }
16
17 void collectPollen(Flower f, double howMuch){
18     double amount = f.harvestPollen(howMuch);
19     collectedPollen = collectedPollen + amount;
20     System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
21 }
22
23
24 void snooze(){
25     System.out.println("schnarch!");
26 }
27
28 public void sting(){
29     System.out.println("pieks!");
30 }
31 }
32 
```

Console

```
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
oooooooooooooo
```

aUebung1/src/javaUebung1/LittleBee.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java
12     double amount = f.harvestPollen();
13     collectedPollen = collectedPollen + amount;
14     System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
15 }
16
17 void collectPollen(Flower f, double howMuch){
18     double amount = f.harvestPollen(howMuch);
19     collectedPollen = collectedPollen + amount;
20     System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
21 }
22
23
24 void snooze(){
25     System.out.println("schnarch!");
26 }
27
28 public void sting(){
29     System.out.println("pieks!");
30 }
31
32 public void flySlow(){
33 }
34 
```

Console

```
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
oooooooooooooo
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanon = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanon);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16        maja.flySlow();
17
18    }
19
20 }
21 
```

Console

```
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
oooooooooooooo
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanon = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanon);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16        maja.flySlow();
17
18    }
19
20 }
21 
```

Console

```
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
brummsididumsel!
```

aUebung1/src/javaUebung1/FlyingInsect.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer Flower.java *BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16        maja.flySlow();
17        FlyingInsect flyingInsect;
18        flyingInsect
19
20    }
21
22 }
23
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:43:39)
Ei, ich hab so schoen 50.0 pollen eingesammelt "grins"
100.0
250.0
schnarch!
brumselfidumsel!
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer Flower.java *BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16        maja.flySlow();
17        FlyingInsect flyingInsect;
18        flyingInsect = maja; | I
19
20    }
21
22 }
23
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:43:39)
Ei, ich hab so schoen 50.0 pollen eingesammelt "grins"
100.0
250.0
schnarch!
brumselfidumsel!
```

aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer Flower.java *BeeDemo.java LittleBee.java FlyingInsect.java
1 package javaUebung1;
2
3 class FlyingInsect {
4
5     int weight;
6
7     void flySlow(){
8         System.out.println("oooooooooooo");
9     }
10
11 }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:43:39)
Ei, ich hab so schoen 50.0 pollen eingesammelt "grins"
100.0
250.0
schnarch!
brumselfidumsel!
```

Java - aUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access
Package Explorer Flower.java *BeeDemo.java LittleBee.java FlyingInsect.java AngryHornet.java
1 package javaUebung1;
2
3 public class AngryHornet extends FlyingInsect implements ICanSing{
4
5     boolean isVeryAngry;
6
7     public void sting(){
8         System.out.println("MEGAPIEKS!");
9     }
10
11 }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:45:40)
100.0
250.0
schnarch!
brumselfidumsel!
```

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java *AngryHornet.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16        maja.flySlow();
17        FlyingInsect flyingInsect;
18        flyingInsect = maja;
19        flyingInsect.flySlow();
20        AngryHornet horst = new AngryHornet();
21        flyingInsect = horst;
22        flyingInsect.flySlow();
23    }
24
25 }
```

Console > terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:45:40)

```
100.0
250.0
schnarch!
brumseldidumsel!
brumseldidumsel!
```

Java - javaUebung1/src/javaUebung1/AngryHornet.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java AngryHornet.java
1 package javaUebung1;
2
3 public class AngryHornet extends FlyingInsect implements ICanSting{
4
5     boolean isVeryAngry;
6
7     public void sting(){
8         System.out.println("MEGAPIEKS!");
9     }
10
11    public void flySlow(){
12        System.out.println("MEGABRUMMSEL!");
13    }
14
15 }
```

Console > terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:48:06)

```
250.0
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGABRUMMSEL!
```

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java AngryHornet.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16        maja.flySlow();
17        FlyingInsect flyingInsect;
18        flyingInsect = maja;
19        flyingInsect.flySlow();
20        AngryHornet horst = new AngryHornet();
21        flyingInsect = horst;
22        flyingInsect.flySlow();
23    }
24
25 }
```

Console > terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:48:06)

```
250.0
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGABRUMMSEL!
```

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
Flower.java BeeDemo.java LittleBee.java FlyingInsect.java AngryHornet.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getAmountOfPollen();
15        willi.snooze();
16        maja.flySlow();
17        FlyingInsect flyingInsect;
18        flyingInsect = maja;
19        flyingInsect.flySlow();
20        AngryHornet horst = new AngryHornet();
21        horst.flyFast();
22        flyingInsect = horst;
23        flyingInsect.flySlow();
24    }
25 }
```

Console > terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:49:15)

```
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERBRUMMSEL!
MEGABRUMMSEL!
```

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```

1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getSnooze();
15        maja.flySlow();
16        FlyingInsect flyingInsect;
17        flyingInsect = maja;
18        flyingInsect.flySlow();
19        AngryHornet horst = new AngryHornet();
20        horst.flyFast();
21        flyingInsect = horst;
22        flyingInsect.flySlow();
23    }
24
25 }

```

Console

```

<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:49:15)
schnarch!
brumselidumself!
brumselidumself!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!

```

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```

1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getSnooze();
15        willi.snooze();
16        maja.flySlow();
17        FlyingInsect flyingInsect;
18        flyingInsect = maja;
19        flyingInsect.flySlow();
20        AngryHornet horst = new AngryHornet();
21        horst.flyFast();
22        flyingInsect = horst;
23        flyingInsect.flySlow();
24        flyingInsect.flyFast();
25    }
26
27 }

```

Console

```

<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:49:15)
schnarch!
brumselidumself!
brumselidumself!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!

```

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```

1 package javaUebung1;
2
3 public class BeeDemo {
4
5     public static void main(String[] args) {
6         Flower lilly = new Flower(300.0d);
7         Flower elanor = new Flower();
8         LittleBee maja = new LittleBee();
9         LittleBee willi = new LittleBee();
10        maja.collectPollen(lilly);
11        maja.collectPollen(elanor);
12        System.out.println(maja.getCollectedPollen());
13        System.out.println(lilly.getAmountOfPollen());
14        lilly.getSnooze();
15        willi.snooze();
16        FlyingInsect flyingInsect;
17        flyingInsect = maja;
18        flyingInsect.flySlow();
19        AngryHornet horst = new AngryHornet();
20        horst.flyFast();
21        flyingInsect = horst;
22        flyingInsect.flySlow();
23        horst.flyFast();
24    }
25
26 }

```

Console

```

<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumselidumself!
brumselidumself!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!

```

Java.pptx - PowerPoint

DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHIRMPRÄSENTATION ÜBERPRÜFEN ANSICHT Anmelden

Overriding

Overriding: In einer **Unterklasse** Methode mit **gleichem Namen**, und **gleicher Parameterliste** wie in **Oberklasse**:

```

class Bicycle {
    int speed;
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("superclass instance-method");
    }
}

class MountainBike extends Bicycle {
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("subclass instance-method");
    }
}

MountainBike mb = new MountainBike();
mb.speedUp(10); // mb.speed == 20

```

Ausgabe: **subclass instance-method**

Sinn: Unterklasse bietet speziellere Version der Methode an (Aspekt von Polymorphie)

FOLIE 121 VON 176 ENGLISCH USA NOTIZEN KOMMENTARE DE (2:17) 13:50 03.07.2015

Overriding

Overriding: In einer Unterklass Methode mit **gleichem Namen**, und **gleicher Parameterliste** wie in Oberklasse:

```

class Bicycle {
    int speed;
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("superclass instance-method");
    }
}

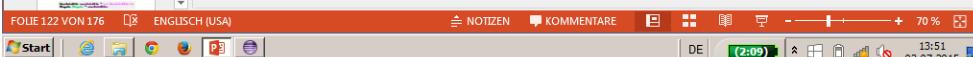
class MountainBike extends Bicycle {
    public void speedUp(int increment) {
        super(2 * increment); // call overridden method of superclass
        System.out.println("subclass instance-method");
    }
}

MountainBike mb = new MountainBike();
mb.speedUp(10); // mb.speed == 20

```

Ausgabe: **superclass instance-method**
subclass instance-method

Sinn: Unterklasse bietet speziellere Version der Methode an (Aspekt von Polymorphie)



Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java Debug

Package Explorer

Flower.java BeeDemo.java LittleBee.java FlyingInsect.java AngryHornet.java

```

6 Flower lilly = new Flower("300.ed");
7 Flower elanor = new Flower();
8 LittleBee maja = new LittleBee();
9 LittleBee willi = new LittleBee();
10 maja.collectPollen(lilly);
11 maja.collectPollen(elanor);
12 System.out.println(maja.getCollectedPollen());
13 System.out.println(lilly.getAmountOfPollen());
14 lilly.getAmountOfPollen();
15 willi.snooze();
16 maja.flySlow();
17 FlyingInsect flyingInsect;
18 FlyingInsect = maja;
19 flyingInsect.flySlow();
20 AngryHornet horst = new AngryHornet();
21 horst.flyFast();
22 flyingInsect = horst;
23 flyingInsect.flySlow();
24 horst.flyFast();

```

Console

```

<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumselidumself!
brumselidumself!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMSEL!
MEGASUPERSUPERBRUMMSEL!

```

Writable Smart Insert 24:14

DE (2:09) 13:52 03.07.2015

Overriding

Overriding: In einer Unterklass Methode mit **gleichem Namen**, und **gleicher Parameterliste** wie in Oberklasse:

```

class Bicycle {
    int speed;
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("superclass instance-method");
    }
}

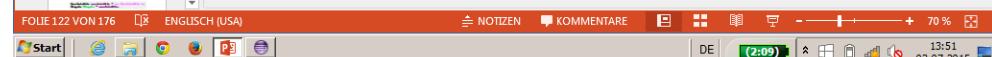
class MountainBike extends Bicycle {
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("subclass instance-method");
    }
}

MountainBike mb = new MountainBike();
mb.speedUp(10); // mb.speed == 20

```

Ausgabe: **subclass instance-method**

Sinn: Unterklasse bietet speziellere Version der Methode an (Aspekt von Polymorphie)



Polymorphie, Interfaces <--> Polymorphie

- Sinn und Zweck von Polymorphie:** Allgemeinere(s) Superklassen-Verhalten und -Zustände sind **garantiert**, d.h. können stets benutzt werden (→ gutes Software-Design).
- „Ein MountainBike ist ein Bicycle“*
- Ähnlich:** **Interfaces** „garantieren“ ebenfalls einen Satz von Methoden → diese garantierten Methoden können von **Objekten** aller derjenigen **Klassen**, die das Interface **implementieren**, auch genutzt werden.

DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHEINPRÄSENTATION ÜBERPRÜFEN ANSICHT Anmelden

Einfügen Neue Folie Formen Anordnen Schnellformatvorlagen Suchen Ersetzen Markieren

Zwischenablage Foliens Schriftart Absatz Zeichnung Bearbeiten

125 126 127 128 129

FOLIE 126 VON 176 ENGLISCH (USA) NOTIZEN KOMMENTARE DE (2:09) 13:53 03.07.2015

Java.pptx - PowerPoint

DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHRIMPRASENTATION ÜBERPRÜFEN ANSICHT FORMAT Anmel...

Einfügen
Neue Folie
Zwischenablage Folien

Schriftart: Courier New - 16 A A A F K U S abc AV Aa A Absatz

Formen Anordnen Schnellformatvorlagen Zeichnung Bearbeiten

Suchen Ersetzen Markieren Bearbeiten

126

127

128

129

130

Interfaces <---> Polymorphie : Beispiel

```

interface SubOrderApocrita {
    public void sting();
}

class LittleBee implements SubOrderApocrita {
    public void sting() {
        System.out.println("*pieks*");
    }
}

class AngryHornet implements SubOrderApocrita {
    public void sting() {
        System.out.println("MEGAPIEKS");
    }
}

LittleBee majia = new LittleBee();
AngryHornet horst = new AngryHornet();
SubOrderApocrita someStinger;
someStinger = majia;
someStinger.sting(); // *pieks*
someStinger = horst;
someStinger.sting(); // MEGAPIEKS

```

129

FOLIE 129 VON 176 DEUTSCH (DEUTSCHLAND) NOTIZEN KOMMENTARE 13:54

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

- Project Explorer View (left):** Shows the project structure under "javaUebung1". The "src" folder contains packages like "javaUebung1" and "javaUebung1a", which further contain classes such as "AngryHornet.java", "BeeDemo.java", "Flower.java", "FlyingInsect.java", "ICanSting.java", and "LittleBee.java".
- Editor View (center):** Displays the file "ICanSting.java" containing the following code:

```
1 package javaUebung1;
2
3 public interface ICanSting {
4     public void sting();
5 }
6
7 }
```
- Console View (bottom):** Shows the output of a Java application named "BeeDemo". The console output is:

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumselidumse1!
brumselidumse1!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMSEL!
MEGASUPERSUPERBRUMMSEL!
```
- Standard Windows Taskbar (bottom):** Shows icons for Start, Internet Explorer, File Explorer, and others.

Java - javaUebung1/src/javaUebung1/ICanString.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access Java Debug
Package Explorer
* BeeDemo.java LittleBee.java FlyingInsec... AngryHornet... ICanString.java
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
Flower lilly = new Flower("300.0d");
Flower elanor = new Flower();
LittleBee maja = new LittleBee();
LittleBee willi = new LittleBee();
maja.collectPollen(lilly);
maja.collectPollen(elanor);
System.out.println(maja.getCollectedPollen());
System.out.println(lilly.getAmountOfPollen());
lilly.getAmountOfPollen();
willi.snooze();
maja.flySlow();
FlyingInsect flyingInsect;
flyingInsect = maja;
flyingInsect.flySlow();
AngryHornet horst = new AngryHornet();
horst.flyFast();
flyingInsect = horst;
flyingInsect.flySlow();
horst.flyFast();
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseidumsel!
brumseidumsel!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
```

Writable Smart Insert 1 : 1

Start DE (2:08) 13:56 03.07.2015

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access Java Debug
Package Explorer
* BeeDemo.java LittleBee.java FlyingInsec... AngryHornet... ICanString.java
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
package javaUebung1;
public interface ICanString {
    public void sting();
}
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseidumsel!
brumseidumsel!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
```

Writable Smart Insert 26 : 26

Start DE (2:06) 13:56 03.07.2015

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access Java Debug
Package Explorer
* BeeDemo.java LittleBee.java FlyingInsec... AngryHornet... ICanString.java
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
Flower lilly = new Flower("300.0d");
Flower elanor = new Flower();
LittleBee maja = new LittleBee();
LittleBee willi = new LittleBee();
maja.collectPollen(lilly);
maja.collectPollen(elanor);
System.out.println(maja.getCollectedPollen());
System.out.println(lilly.getAmountOfPollen());
lilly.getAmountOfPollen();
willi.snooze();
maja.flySlow();
FlyingInsect flyingInsect;
flyingInsect = maja;
flyingInsect.flySlow();
AngryHornet horst = new AngryHornet();
horst.flyFast();
flyingInsect = horst;
flyingInsect.flySlow();
horst.flyFast();
ICanString someStinger = new ICanString();
someStinger.sting();
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseidumsel!
brumseidumsel!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
```

Writable Smart Insert 25 : 18

Start DE (2:08) 13:57 03.07.2015

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse

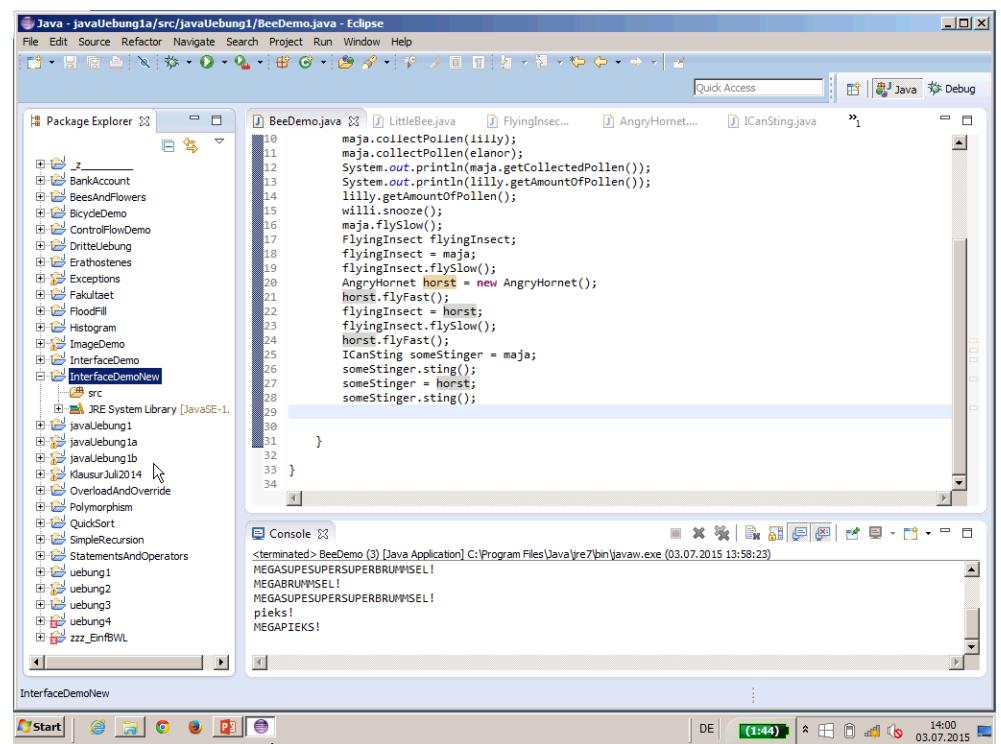
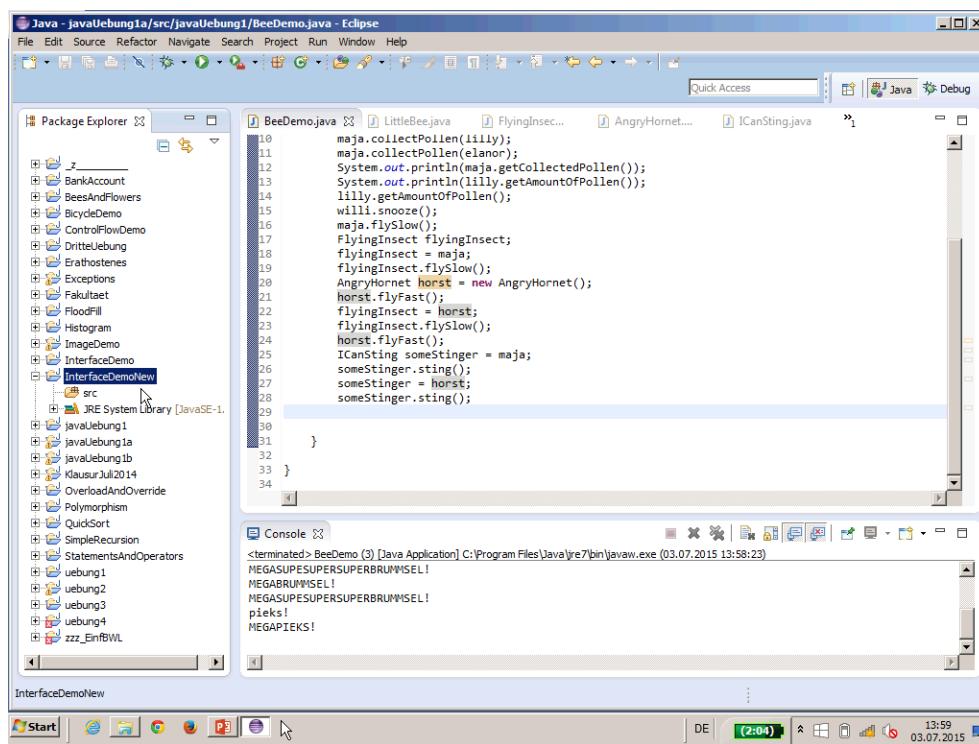
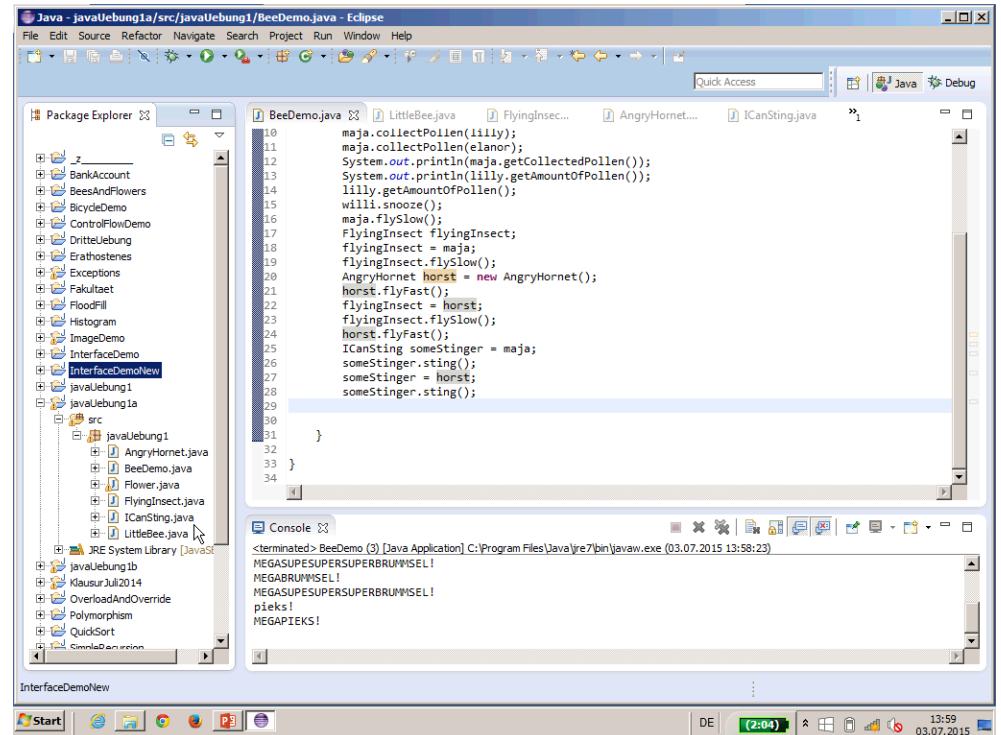
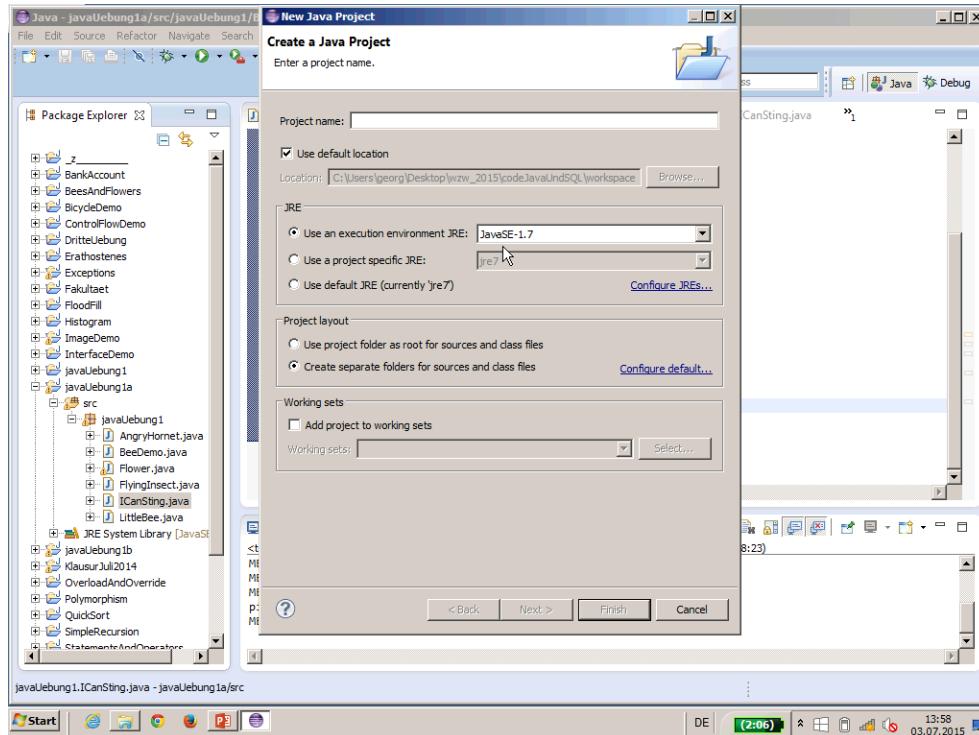
```
File Edit Source Refactor Navigate Project Run Window Help
Quick Access Java Debug
Package Explorer
* BeeDemo.java LittleBee.java FlyingInsec... AngryHornet... ICanString.java
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
maja.collectPollen(lilly);
maja.collectPollen(elanor);
System.out.println(maja.getCollectedPollen());
System.out.println(lilly.getAmountOfPollen());
lilly.getAmountOfPollen();
willi.snooze();
maja.flySlow();
FlyingInsect flyingInsect;
flyingInsect = maja;
flyingInsect.flySlow();
AngryHornet horst = new AngryHornet();
horst.flyFast();
flyingInsect = horst;
flyingInsect.flySlow();
horst.flyFast();
ICanString someStinger = maja;
someStinger.sting();
someStinger = horst;
someStinger.sting();
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseidumsel!
brumseidumsel!
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
```

Writable Smart Insert 26 : 29

Start DE (2:08) 13:57 03.07.2015



Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java InterfaceDem...
src
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7
8     }
9
10 }
11
```

Console <terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java *InterfaceDem...
src
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6
7     }
8
9
10 }
11
12 }
```

Console <terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java *InterfaceDem...
src
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6
7     }
8
9
10 }
11
12 }
```

Console <terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

Wikipedia - Bubblesort

Nicht verbunden
Es sind Verbindungen verfügbar.

Drahtlosnetzwerkverbindung
FMI
eduroam
lrz
beamerjhs
Anderes Netzwerk

Debug

Magyar
Հայերեն
Íslenska
Italiano
日本語
Қазақша
한국어
Kurdi
Lëtzebuergesch
Lietuviai
ଓଡ଼ିଆ
Nederlands
Norsk bokmål
Polski
Português
Русский
Simple English
Slovenčina
Slovenščina
Српски / srpski
Svenska
ไทย
Tagalog
Türkçe
Українська
Tiếng Việt

Je nachdem, ob auf- oder absteigend sortiert wird, steigen die größeren oder kleineren Elemente wie Blasen im Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation $>$ (größer als) verwendet. Wie bei jedem vergleichbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

```
bubbleSort(Array A)
    for (n=A.size; n>1; n=n-1){
        for (i=0; i<n-1; i=i+1){
            if (A[i] > A[i+1]){
                A.swap(i, i+1)
            } // ende if
        } // ende innere for-Schleife
    } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubble an der rechten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

W Bubblesort - Wikipedia

<https://de.wikipedia.org/wiki/Bubblesort>

```
bubbleSort2(Array A)
n = A.size
do{
    swapped = false
    for (i=0; i<n-1; ++i){
        if (A[i] > A[i+1]){
            A.swap(i, i+1)
            swapped = true
        } // ende if
    } // ende for
    n = n-1
} while (swapped == true)
```

Die äußere Schleife durchläuft die zu sortierenden Daten, bis keine Vertauschungen mehr notwendig sind.

Eine weitere Optimierung besteht in der Ausnutzung der Eigenschaft, dass am Ende einer äußeren Iteration alle Elemente rechts von der letzten Tauschposition schon an ihrer endgültigen Position stehen:

```
bubbleSort3(Array A)
n = A.size
do{
    newn = 1
```

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

W Bubblesort - Wikipedia

<https://de.wikipedia.org/wiki/Bubblesort>

Kurdî Léitzeburgesch Lietuvia ମେଲାଙ୍ଗୁଡ଼ି Nederlands Norsk bokmål Polski Português Русский Simple English Slovenčina Slovenščina Српски / srpski Svenska ไทย Tagalog Türkçe Українська Tiếng Việt 中文

Links bearbeiten

```
bubbleSort(Array A)
for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
        if (A[i] > A[i+1]){
            A.swap(i, i+1)
        } // ende if
    } // ende innere for-Schleife
} // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubble an der rechten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium verletzen).

Allerdings nutzt diese einfachste Variante nicht die Eigenschaft aus, dass nach einer Iteration, in der keine Vertauschungen stattfanden auch in den restlichen Iterationen keine Vertauschungen mehr stattfinden. Der folgende Pseudocode berücksichtigt dies:

```
bubbleSort2(Array A)
```

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

W Bubblesort - Wikipedia

<https://de.wikipedia.org/wiki/Bubblesort>

Links auf diese Seite Änderungen an verlinkten Seiten Spezialseiten Permanenter Link Seiteninformationen Wikidata-Datenobjekt Artikel zitieren

In anderen Sprachen العربية Azərbaycanca Български Català Čeština Dansk Ελληνικά English Español Estonian فارسی Suomi Français עברית Magyar Հայերեն Íslenska

4.3 Durchschnittlicher Fall 5 Abgrenzung 5.1 Hasen und Schildkröten 6 Einzelnachweise 7 Literatur 8 Weblinks

Prinzip [Bearbeiten]

In der Bubble-Phase wird die Eingabe-Liste von links nach rechts durchlaufen. Dabei wird in jedem Schritt das aktuelle Element mit dem rechten Nachbarn verglichen. Falls die beiden Elemente das Sortierkriterium verletzen, werden sie getauscht. Am Ende der Phase steht bei auf- bzw. absteigender Sortierung das größte bzw. kleinste Element der Eingabe am Ende der Liste.

Die Bubble-Phase wird solange wiederholt, bis die Eingabeliste vollständig sortiert ist. Dabei muss das letzte Element des vorherigen Durchlaufs nicht mehr betrachtet werden, da die restliche zu sortierende Eingabe keine größeren bzw. kleineren Elemente mehr enthält.

Je nachdem, ob auf- oder absteigend sortiert wird, steigen die größeren oder kleineren Elemente wie Blasen im Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

Bubblesort an einer Zahlenliste

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

W Bubblesort - Wikipedia

<https://de.wikipedia.org/wiki/Bubblesort>

Magyar Հայերեն Íslenska Italiano 日本語 Kazaksha 한국어 Kurdî Léitzeburgesch Lietuvia ମେଲାଙ୍ଗୁଡ଼ି Nederlands Norsk bokmål Polski Português Русский Simple English Slovenčina Slovenščina Српски / srpski Svenska ไทย Tagalog Türkçe Українська Tiếng Việt 中文

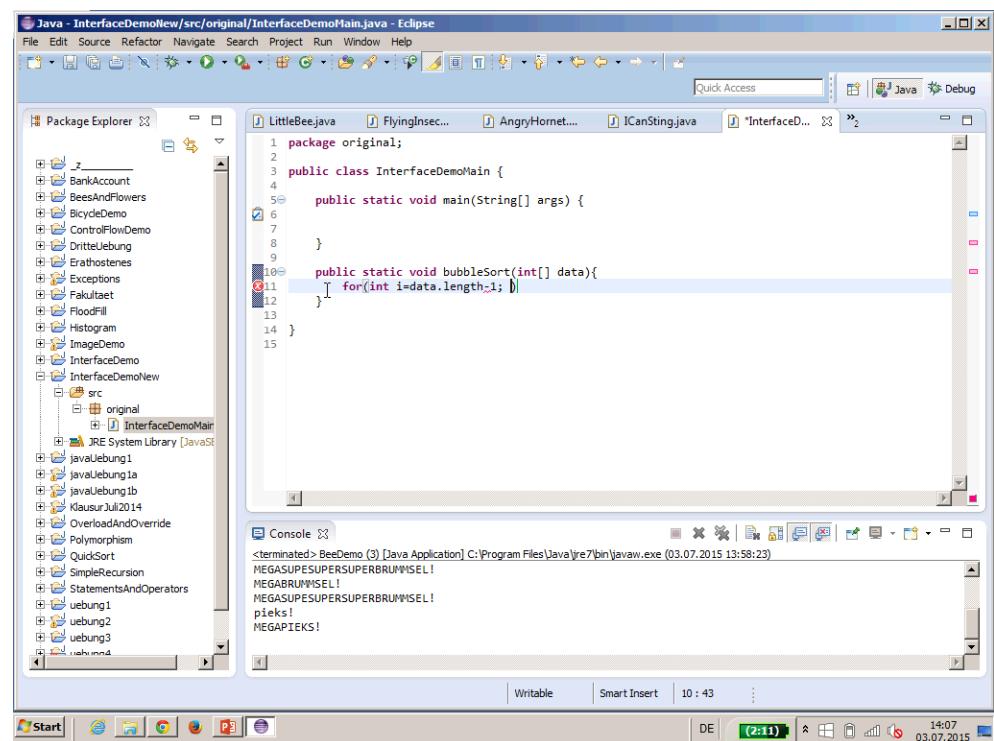
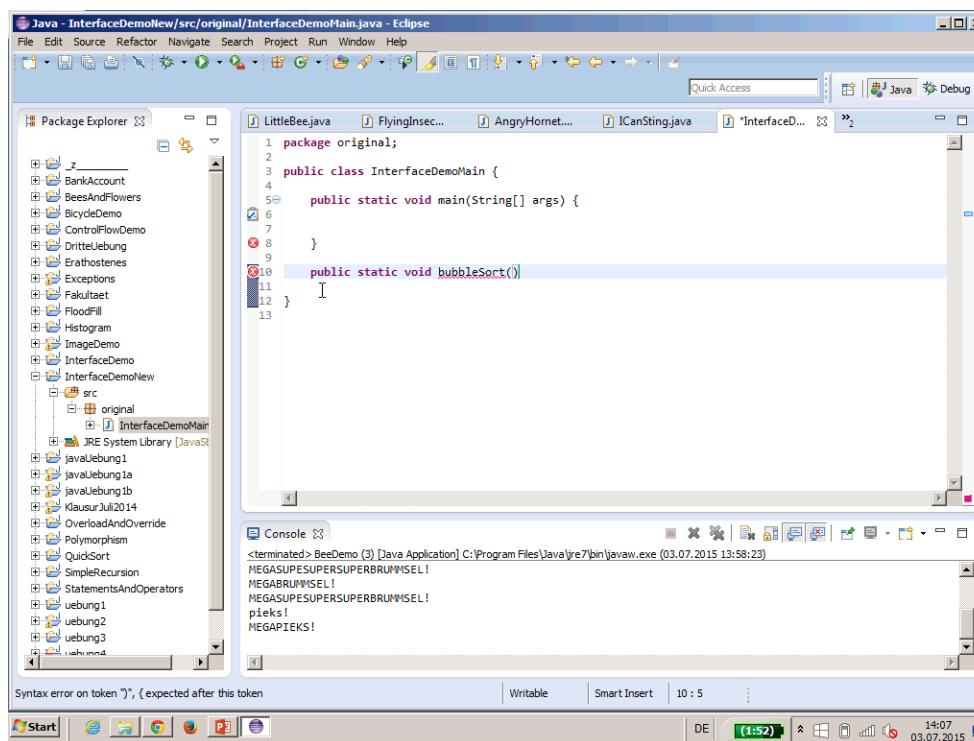
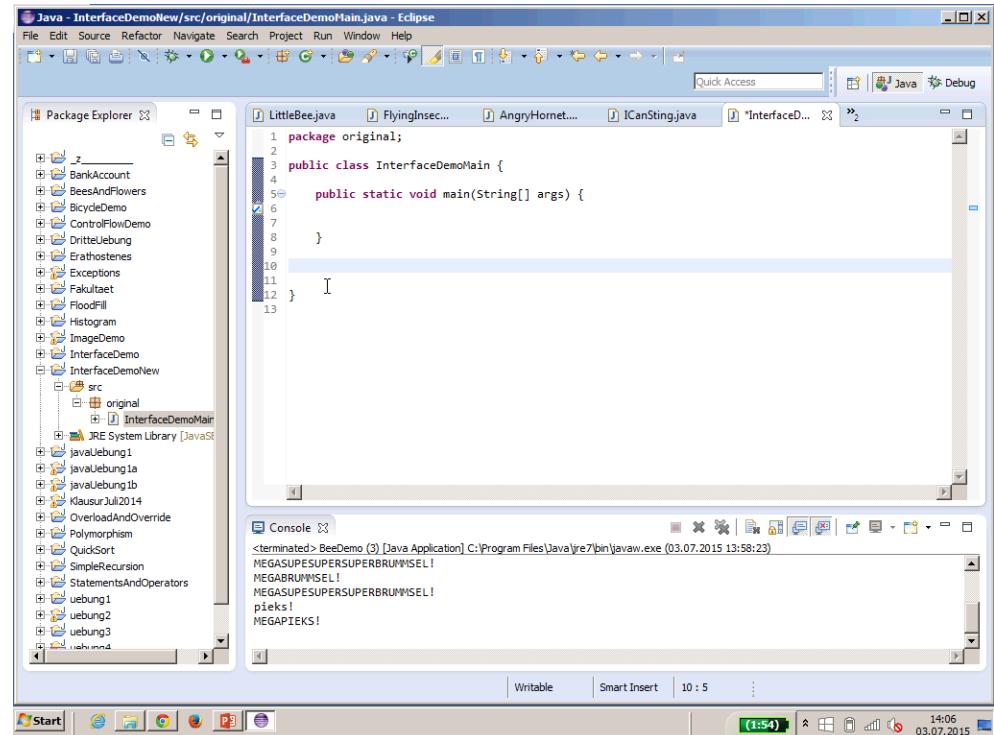
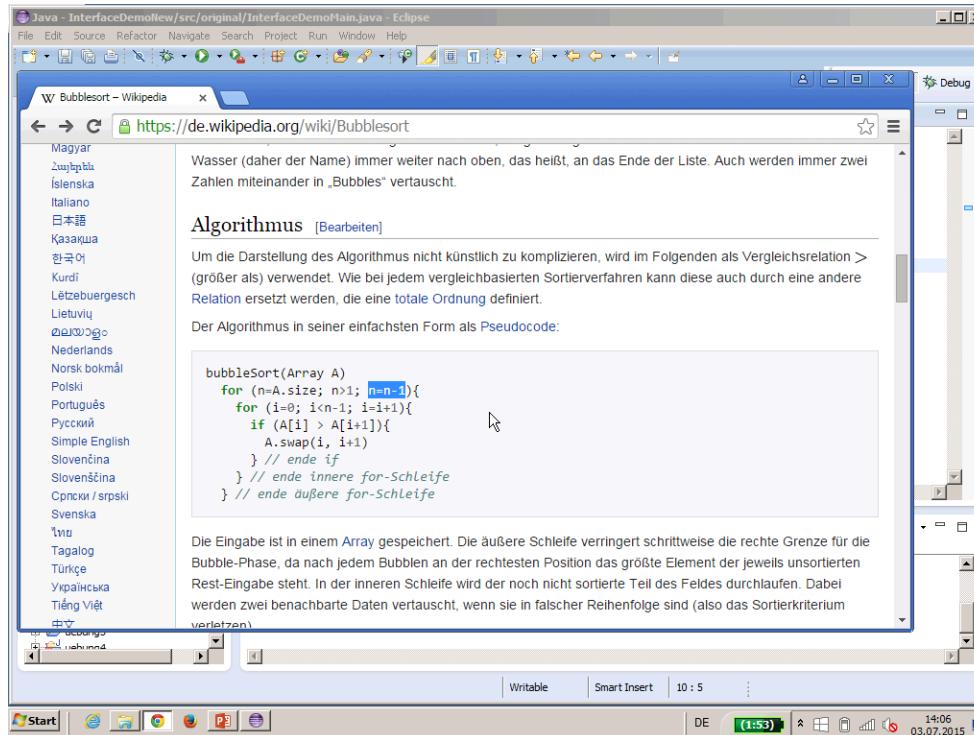
Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation $>$ (größer als) verwendet. Wie bei jedem vergleichbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

```
bubbleSort(Array A)
for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
        if (A[i] > A[i+1]){
            A.swap(i, i+1)
        } // ende if
    } // ende innere for-Schleife
} // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubble an der rechten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium verletzen).



Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

W Bubblesort – Wikipedia

https://de.wikipedia.org/wiki/Bubblesort

Magyar
Հայերեն
Íslenska
Italiano
日本語
Қазақша
한국어
Kurdî¹
Lëtzebuergesch
Lietuvių
ଓঠামুণ্ডো
Nederlands
Norsk bokmål
Polski
Português
Русский
Simple English
Slovenčina
Slovenščina
Српски / srpski
Svenska
ไทย
Tagalog
Türkçe
Українська
Tiếng Việt
中文

Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation $>$ (größer als) verwendet. Wie bei jedem vergleichsbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

```
bubbleSort(Array A)
    for (n=A.size; n>1; n=n-1){
        for (i=0; i<n-1; i=i+1){
            if (A[i] > A[i+1]){
                A.swap(i, i+1)
            } // ende if
        } // ende innere for-Schleife
    } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubble an der rechten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sorterkriterium verletzen).

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java Debug

Package Explorer

LittleBee.java FlyingInsec... AngryHornet... ICanSting.java *InterfaceD...

```
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6
7
8     }
9
10    public static void bubbleSort(int[] data){
11        for(int i=data.length-1; i>0; i--){
12
13        }
14    }
15
16 }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

W Bubblesort – Wikipedia

https://de.wikipedia.org/wiki/Bubblesort

Magyar
Հայերեն
Íslenska
Italiano
日本語
Қазақша
한국어²
Kurdî¹
Lëtzebuergesch
Lietuvių
ଓঠামুণ্ডো
Nederlands
Norsk bokmål
Polski
Português
Русский
Simple English
Slovenčina
Slovenščina
Српски / srpski
Svenska
ไทย
Tagalog
Türkçe
Українська
Tiếng Việt
中文

Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation $>$ (größer als) verwendet. Wie bei jedem vergleichsbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

```
bubbleSort(Array A)
    for (n=A.size; n>1; n=n-1){
        for (i=0; i<n-1; i=i+1){
            if (A[i] > A[i+1]){
                A.swap(i, i+1)
            } // ende if
        } // ende innere for-Schleife
    } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubble an der rechten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sorterkriterium verletzen).

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java Debug

Package Explorer

LittleBee.java FlyingInsec... AngryHornet... ICanSting.java *InterfaceD...

```
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6
7
8     }
9
10    public static void bubbleSort(int[] data){
11        for(int i=data.length-1; i>0; i--){
12            for(int j=0; j<i; j++){
13
14        }
15    }
16 }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

Java.pptx - PowerPoint

DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCREENPRÄSENTATION UBERPRÜFEN ANSICHT Anmelden

W Bubblesort – Wikipedia https://de.wikipedia.org/wiki/Bubblesort

Seiteninformationen Literatur Weblinks

In anderen Sprachen: العربية Azərbaycanca Български Català Čeština Dansk Ελληνικά English Español Estonian فارسی Suomi Français עברית Magyar સংস্কৃত Isielska Italiano 日本語 Қазақша 한국어 Kurdi

Prinzip [Bearbeiten]

In der Bubble-Phase wird die Eingabe-Liste von links nach rechts durchlaufen. Dabei wird in jedem Schritt das aktuelle Element mit dem rechten Nachbarn verglichen. Falls die beiden Elemente das Sortierkriterium verletzen, werden sie getauscht. Am Ende der Phase steht bei auf- bzw. absteigender Sortierung das größte bzw. kleinste Element der Eingabe am Ende der Liste.

Die Bubble-Phase wird solange wiederholt, bis die Eingabeliste vollständig sortiert ist. Dabei muss das letzte Element des vorherigen Durchlaufs nicht mehr betrachtet werden, da die restliche zu sortierende Eingabe keine größeren bzw. kleineren Elemente mehr enthält.

Je nachdem, ob auf- oder absteigend sortiert wird, steigen die größeren oder kleineren Elemente wie Blasen im Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation $>$ (größer als) verwendet. Wie bei jedem vergleichbasierten Sortierverfahren kann diese auch durch eine andere

FOLIE 116 VON 176 ENGLISCH (USA) NOTIZEN KOMMENTARE DE (2:16) 14:10 03.07.2015

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java Debug

Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java *InterfaceD...

```
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6
7
8
9
10    public static void bubbleSort(int[] data){
11        for(int i=data.length-1; i>0; i--){
12            for(int j=0; j<i; j++){
13                if(data[j] > data[j+1])
14                    //swap elements
15
16            }
17        }
18    }
19
20
21
22
23 }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

Writable Smart Insert 13 : 38

Start DE (2:16) 14:10 03.07.2015

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java Debug

Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java *InterfaceD...

```
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6
7
8
9
10    public static void bubbleSort(int[] data){
11        for(int i=data.length-1; i>0; i--){
12            for(int j=0; j<i; j++){
13                if(data[j] > data[j+1])
14                    //swap elements
15                    data[j] = data[j+1];
16                    data[j+1] = data[j];
17            }
18        }
19
20
21
22
23 }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

Writable Smart Insert 15 : 29

Start DE (2:11) 14:11 03.07.2015

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

- Top Bar:** Java - InterfaceDemoNew/src/original/InterfaceDemoMain.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, etc.
- Quick Access:** A search bar at the top right.
- Java Perspective:** Indicated by the Java and Debug buttons in the top right.
- Package Explorer:** Shows the project structure with packages like BankAccount, BeeAndFlowers, BicycleDemo, ControlFlowDemo, DritteÜbung, Erathostenes, Exceptions, Fakultät, FloodFill, Histogram, ImageDemo, InterfaceDemo, and InterfaceDemoNew.
- Editor:** The main window displays the Java code for `InterfaceDemoMain`. The code implements a `bubbleSort` method that sorts an array of integers using a bubble sort algorithm. The code is highlighted with syntax coloring.
- Console:** Shows the terminal output of the application. It includes the command `BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)`, followed by several lines of text:

```
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```
- Bottom Status Bar:** Shows Writable, Smart Insert, and the current time (14:12).
- Taskbar:** Shows icons for Start, File Explorer, Task View, Internet Explorer, and others.

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

- Title Bar:** Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
- Menu Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help
- Toolbar:** Standard Eclipse toolbar icons.
- Quick Access:** A search bar at the top right.
- Java Perspective:** Indicated by the Java icon in the title bar.
- Package Explorer:** Shows the project structure with packages like BankAccount, BeesAndFlowers, BicycleDemo, ControlFlowDemo, DritteUebung, Erathostenes, Exceptions, Fakultaet, Floodfill, Histogram, ImageDemo, InterfaceDemo, and InterfaceDemoNew.
- Editor:** The main editor window displays the Java code for `InterfaceDemoMain`. The cursor is positioned on line 23, which contains the method signature `public static void printarray()`.
- Console:** The bottom console window shows the output of a previous run:

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMSEL!
MEGABRUMSEL!
MEGASUPERSUPERBRUMSEL!
pieks!
MEGAPIEKS!
```
- Syntax Error:** A message at the bottom left states "Syntax error on token ')', { expected after this token".
- Bottom Status Bar:** Writable, Smart Insert, 23 : 7, and a timestamp of 14:13.
- Taskbar:** Shows icons for Start, File Explorer, Task View, Internet Explorer, and File Explorer.

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

- File Bar:** Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse. Includes options: File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Toolbars:** Standard toolbar with icons for New, Open, Save, Cut, Copy, Paste, Find, etc.
- Quick Access:** A search bar at the top right.
- Views:**
 - Package Explorer:** Shows the project structure with packages like BankAccount, BeesAndFlowers, BicycleDemo, ControlFlowDemo, DritteJebung, Erathostes, Exceptions, Fakultat, FloodFill, Histogram, ImageDemo, InterfaceDemo, and InterfaceDemoNew. The InterfaceDemoNew package is expanded, showing its src folder containing the InterfaceDemoMain.java file.
 - Editor:** Displays the Java code for InterfaceDemoMain.java. The code includes a main method, a bubbleSort static method that sorts an array of integers using nested loops, and a printarray static method that prints the elements of an array.
 - Console:** Shows the terminal output of the application's execution. The output reads:

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```
 - Bottom Status Bar:** Shows Writable, Smart Insert, and the current time (14:14 03.07.2015).
 - Taskbar:** Shows the taskbar with icons for Start, Internet Explorer, Google, and others.

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

- Title Bar:** Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
- Menu Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help
- Toolbar:** Standard Eclipse toolbar icons.
- Quick Access:** A search bar at the top right.
- Package Explorer:** Shows the project structure with packages like BeesAndFlowers, BicycleDemo, ControlFlowDemo, DritteJebung, Eratosthenes, Exceptions, Fakultaet, Floodfill, Histogram, ImageDemo, InterfaceDemo, and InterfaceDemoNew.
- Editor:** The main editor window displays the Java code for `InterfaceDemoMain.java`. The code includes a `bubbleSort` method that sorts an array of integers using nested loops and swaps. It also contains a `printarray` method that prints each element of the array to the console.
- Console:** The bottom-right window shows the terminal output of the application. The output reads:

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGAUSPESUMERBRUMSEL!
MEGABRUMSEL!
```
- Bottom Bar:** Shows the operating system's taskbar with icons for Start, Task View, Internet Explorer, Google Chrome, File Explorer, and File History. The status bar at the bottom right shows the date and time: 14:14 03.07.2015.

The screenshot shows the Eclipse Java IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, etc. The left sidebar is the Package Explorer showing a project structure with packages like original, BankAccount, BeeAndFlowers, BicycleDemo, ControlFlowDemo, DritteLebung, Erathostenes, Exceptions, Fakultat, FloodFill, Histogram, ImageDemo, InterfaceDemo, InterfaceDemoNew, and javaUebung1. The main editor window displays the code for InterfaceDemoMain.java:

```
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         int[] foo = {1, 9, 3, 5, 7, 2, -3};
7         |
8     }
9
10    public static void bubbleSort(int[] data){
11        for(int i=data.length-1; i>0; i--){
12            for(int j=0; j<i; j++){
13                if(data[j] > data[j+1]){
14                    //swap elements.
15                    int backup = data[j];
16                    data[j] = data[j+1];
17                    data[j+1] = backup;
18                }
19            }
20        }
21    }
22 }
23
24    public static void printarray(int[] data){
25        for(int i=0; i<data.length; i++){

```

The bottom right corner shows the status bar with the time 14:15 and date 03.07.2015. The bottom left corner shows the Windows taskbar with icons for Start, Internet Explorer, File Explorer, and others.

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

- Title Bar:** Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
- Menu Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help
- Toolbar:** Standard Eclipse toolbar icons.
- Quick Access:** A search bar at the top right.
- Java Perspective:** The main workspace area.
- Package Explorer:** Shows the project structure with packages like BankAccount, BeesAndFlowers, BicycleDemo, ControlFlowDemo, DritteJebung, Eratosthenes, Exceptions, Fakultaet, Floodfill, Histogram, ImageDemo, InterfaceDemo, InterfaceDemoNew, and src.
- Code Editor:** Displays the Java code for InterfaceDemoMain.java. The code includes a main method and a bubbleSort static method. The line `int[] foo = {1, 9, 3, 5, 7, 2, -3};` is highlighted.
- Console:** Shows the output of the application. It prints:

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```
- Bottom Status Bar:** Shows Writable, SmartInsert, and the current time (11:45).
- Task List:** Shows two tasks: (2,11).
- System Tray:** Shows standard Windows system tray icons.

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java *InterfaceDem...
src
LittleBee.java
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         int[] foo = {1, 9, 3, 5, 7, 2, -3};
7         print
8             bubbleSort(foo);
9     }
10
11     public static void bubbleSort(int[] data){
12         for(int i=data.length-1; i>0; i--){
13             for(int j=0; j<i; j++){
14                 if(data[j] > data[j+1]){
15                     //swap elements
16                     int backup = data[j];
17                     data[j] = data[j+1];
18                     data[j+1] = backup;
19                 }
20             }
21         }
22     }
23
24     public static void printArray(int[] data){
25
26         <terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
27         MEGAPUSERSUPERBRUNNSEL!
28         MEGABRUMNSEL!
29         MEGAPUSERSUPERBRUNNSEL!
30         pieks!
31         MEGAPIEKS!
```

Console

```
MEGAPUSERSUPERBRUNNSEL!
MEGABRUMNSEL!
MEGAPUSERSUPERBRUNNSEL!
pieks!
MEGAPIEKS!
```

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java InterfaceDem...
src
LittleBee.java
1 package original;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         int[] foo = {1, 9, 3, 5, 7, 2, -3};
7         printArray(foo);
8         bubbleSort(foo);
9     }
10
11     public static void bubbleSort(int[] data){
12         for(int i=data.length-1; i>0; i--){
13             for(int j=0; j<i; j++){
14                 if(data[j] > data[j+1]){
15                     //swap elements
16                     int backup = data[j];
17                     data[j] = data[j+1];
18                     data[j+1] = backup;
19                 }
20             }
21         }
22     }
23
24     public static void printArray(int[] data){
25
26         <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:37)
27         1
28         9
29         3
30         5
31         7
32         2
33         -3
```

Console

```
1
9
3
5
7
2
-3
```

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java InterfaceDem...
src
LittleBee.java
11
12     public static void bubbleSort(int[] data){
13         for(int i=data.length-1; i>0; i--){
14             for(int j=0; j<i; j++){
15                 if(data[j] > data[j+1]){
16                     //swap elements
17                     int backup = data[j];
18                     data[j] = data[j+1];
19                     data[j+1] = backup;
20                 }
21             }
22         }
23     }
24
25     public static void printArray(int[] data){
26         for(int i=0; i<data.length; i++){
27             System.out.println(data[i]);
28         }
29     }
```

Console

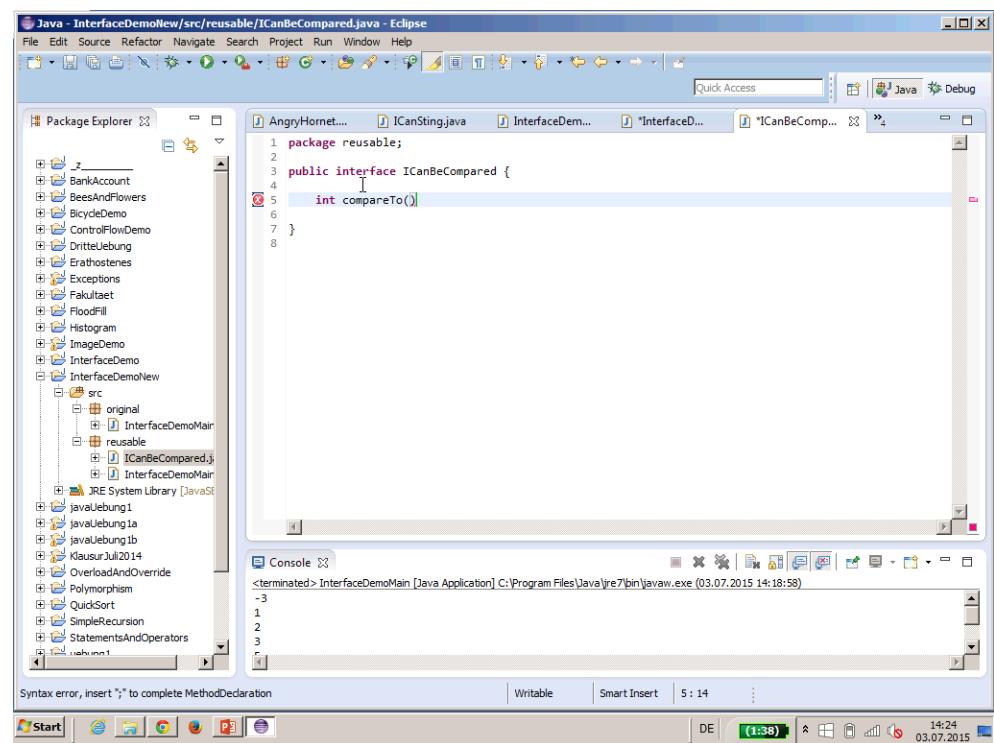
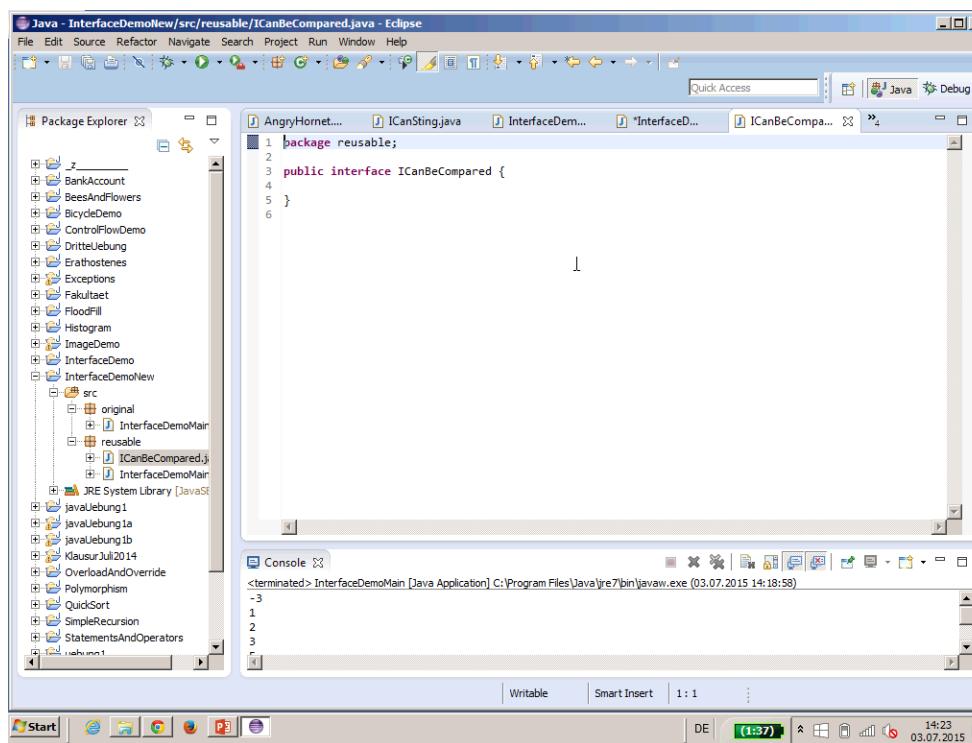
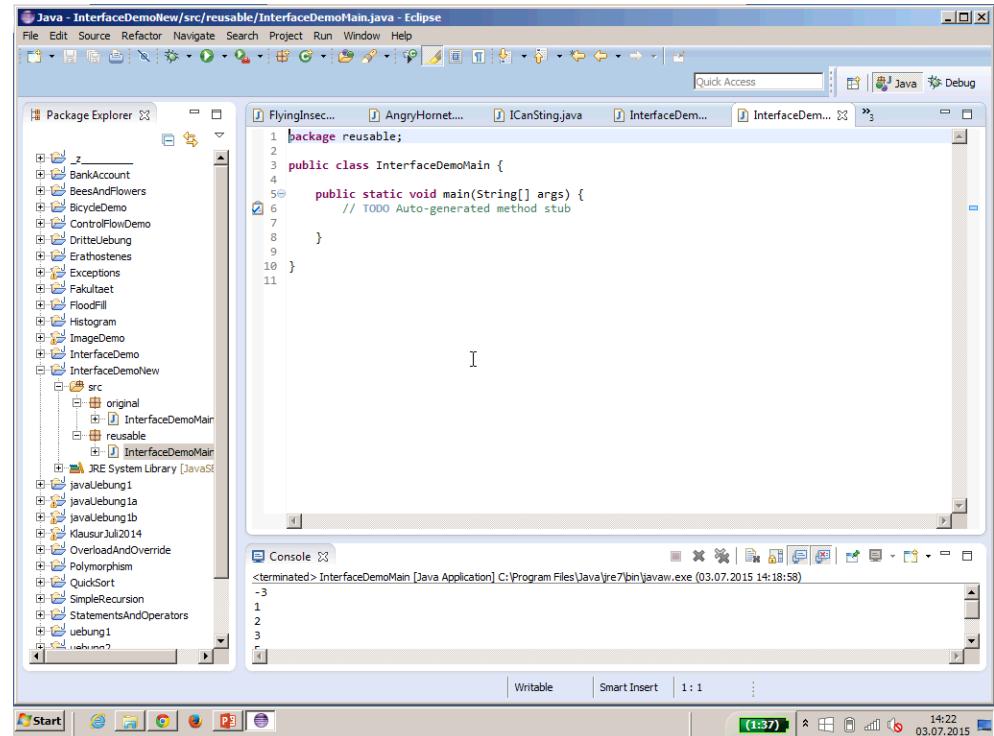
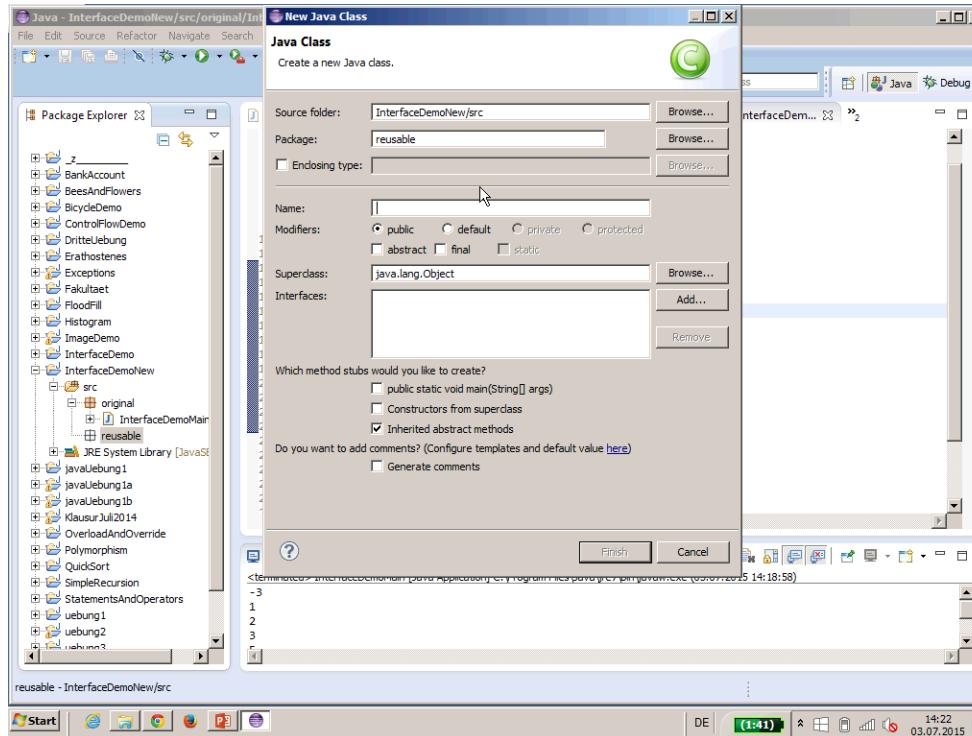
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
4
5
6
7
8
9
```

Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer LittleBee.java FlyingInsec... AngryHornet... ICanString.java InterfaceDem...
src
LittleBee.java
3
4     public class InterfaceDemoMain {
5
6         public static void main(String[] args) {
7             int[] foo = {1, 9, 3, 5, 7, 2, -3};
8             bubbleSort(foo);
9             printArray(foo);
10
11         }
12
13         public static void bubbleSort(int[] data){
14             for(int i=data.length-1; i>0; i--){
15                 for(int j=0; j<i; j++){
16                     if(data[j] > data[j+1]){
17                         //swap elements
18                         int backup = data[j];
19                         data[j] = data[j+1];
20                         data[j+1] = backup;
21                     }
22                 }
23             }
24
25         public static void printArray(int[] data){
26             for(int i=0; i<data.length; i++){
27                 System.out.println(data[i]);
28             }
29         }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
4
5
6
7
8
9
```



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
1 package reusable;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         int[] foo = {1, 9, 3, 5, 7, 2, -3};
7         bubbleSort(foo);
8         printArray(foo);
9     }
10
11     public static void bubbleSort(int[] data){
12         for(int i=data.length-1; i>0; i--){
13             for(int j=0; j<i; j++){
14                 if(data[j] > data[j+1]){
15                     //swap elements
16                     int backup = data[j];
17                     data[j] = data[j+1];
18                     data[j+1] = backup;
19                 }
20             }
21         }
22     }
23
24     public static void printArray(int[] data){
25         for(int i=0; i<data.length; i++){
26             System.out.println(data[i]);
27         }
28     }
29 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 1 : 18

Start

Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
1 package reusable;
2
3 public interface ICanBeCompared {
4
5     int compareTo(ICanBeCompared other);
6 }
7 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 6 : 5

Start

Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
1 package reusable;
2
3 public interface ICanBeCompared {
4
5     int compareTo(ICanBeCompared other);
6 }
7 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 5 : 40

Start

Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
1 package reusable;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         int[] foo = {1, 9, 3, 5, 7, 2, -3};
7         bubbleSort(foo);
8         printArray(foo);
9     }
10
11     public static void bubbleSort(ICanBeCompared[] data){
12         for(int i=data.length-1; i>0; i--){
13             for(int j=0; j<i; j++){
14                 if(data[j] > data[j+1]){
15                     //swap elements
16                     int backup = data[j];
17                     data[j] = data[j+1];
18                     data[j+1] = backup;
19                 }
20             }
21         }
22     }
23
24     public static void printArray(int[] data){
25         for(int i=0; i<data.length; i++){
26             System.out.println(data[i]);
27         }
28     }
29 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 15 : 30

Start

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
public static void main(String[] args) {
    int[] foo = {1, 9, 3, 5, 7, 2, -3};
    bubbleSort(foo);
    printArray(foo);
}

public static void bubbleSort(ICanBeCompared[] data){
    for(int i=data.length-1; i>0; i--){
        for(int j=0; j<i; j++){
            if(data[j].compareTo(data[j+1]) == 1){
                //swap elements
                ICanBeCompared backup = data[j];
                data[j] = data[j+1];
                data[j+1] = backup;
            }
        }
    }
}

public static void printArray(ICanBeCompared[] data){
    for(int i=0; i<data.length; i++){
        System.out.println(data[i]);
    }
}
```

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```
public static void main(String[] args) {
    int[] foo = {1, 9, 3, 5, 7, 2, -3};
    bubbleSort(foo);
    printArray(foo);
}

public static void bubbleSort(ICanBeCompared[] data){
    for(int i=data.length-1; i>0; i--){
        for(int j=0; j<i; j++){
            if(data[j].compareTo(data[j+1]) == 1){
                //swap elements
                ICanBeCompared backup = data[j];
                data[j] = data[j+1];
                data[j+1] = backup;
            }
        }
    }
}

public static void printArray(ICanBeCompared[] data){
    for(int i=0; i<data.length; i++){
        System.out.println(data[i]);
    }
}
```

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
package reusable;

public class InterfaceDemoMain {
    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);
        printArray(foo);
    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(ICanBeCompared[] data){
        for(int i=0; i<data.length; i++){
            data[i].print();
        }
    }
}
```

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
package reusable;

public class Vehicle {
}
```

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... *Vehicle.java
1 package reusable;
2
3 public class Vehicle {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11     public s
12
13 }
14
15 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 11: 12

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... *Vehicle.java
1 package reusable;
2
3 public class Vehicle implements ICanBeCompared {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11     public void printItself(){
12         System.out.println(size);
13     }
14
15     public int compareTo(ICanBeCompared someThingThatCanBeCompared)
16     {
17     }
18
19 }
20
21 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 3: 47

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... *Vehicle.java
1 package reusable;
2
3 public class Vehicle implements ICanBeCompared {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11     public void printItself(){
12         System.out.println(size);
13     }
14
15     public int compareTo(ICanBeCompared someThingThatCanBeCompared)
16     {
17     }
18
19 }
20
21 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 15: 41

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... *Vehicle.java
1 package reusable;
2
3 public class Vehicle implements ICanBeCompared {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11     public void printItself(){
12         System.out.println(size);
13     }
14
15     public int compareTo(ICanBeCompared someThingThatCanBeCompared)
16     {
17     }
18
19 }
20
21 }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 16: 9

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java
1 package reusable;
2
3 public class Vehicle implements ICanBeCompared {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11    public void printItself(){
12        System.out.println(size);
13    }
14
15    public int compareTo(ICanBeCompared somethingThatCanBeCompared){
16        if(somethingThatCanBeCompared == null)
17            return -1;
18        else if(this.size < somethingThatCanBeCompared.size)
19            return -1;
20        else if(this.size > somethingThatCanBeCompared.size)
21            return 1;
22        else
23            return 0;
24    }
25
26 }
27
```

Syntax error on token ")", Statement expected after this token

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java
1 package reusable;
2
3 public class Vehicle implements ICanBeCompared {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11    public void printItself(){
12        System.out.println(size);
13    }
14
15    public int compareTo(ICanBeCompared somethingThatCanBeCompared){
16        if(somethingThatCanBeCompared instanceof Vehicle){
17            Vehicle vehicle = (Vehicle) somethingThatCanBeCompared;
18            if(this.size < vehicle.size)
19                return -1;
20            else if(this.size == vehicle.size)
21                return 0;
22            else
23                return 1;
24        }
25        else
26            return -1;
27    }
28
29 }
30
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java
1 package reusable;
2
3 public class Vehicle implements ICanBeCompared {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11    public void printItself(){
12        System.out.println(size);
13    }
14
15    public int compareTo(ICanBeCompared somethingThatCanBeCompared){
16        if(somethingThatCanBeCompared instanceof Vehicle){
17            Vehicle vehicle = (Vehicle) somethingThatCanBeCompared;
18            if(this.size < vehicle.size)
19                return -1;
20            else if(this.size == vehicle.size)
21                return 0;
22            else
23                return 1;
24        }
25        else
26            return -1;
27    }
28
29 }
30
```

Syntax error on token ")", Statement expected after this token

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java
1 package reusable;
2
3 public class Vehicle implements ICanBeCompared {
4
5     private int size;
6
7     public Vehicle(int someSize){
8         size = someSize;
9     }
10
11    public void printItself(){
12        System.out.println(size);
13    }
14
15    public int compareTo(ICanBeCompared somethingThatCanBeCompared){
16        if(somethingThatCanBeCompared instanceof Vehicle){
17            Vehicle vehicle = (Vehicle) somethingThatCanBeCompared;
18            if(this.size < vehicle.size)
19                return -1;
20            else if(this.size == vehicle.size)
21                return 0;
22            else
23                return 1;
24        }
25        else
26            return -1;
27    }
28
29 }
30
```

Multiple markers at this line

- This method must return a result of type int
- implements reusable.ICanBeCompared.compareTo() somethingThatCanBeCompared;

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
ICanString.java InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java
src
  original
    ICanBeCompared.j...
  reusable
    ICanBeCompared.j...
    InterfaceDemoMain...
    Vehicle.java
JRE System Library [JavaSt...
javaJebung1
javaJebung1a
javaJebung1b
KlausurJul2014
OverloadAndOverride
Polymorphism
QuidSort
SimpleRecursion
StatementsAndOperat...
Console
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
229
230
231
232
233
234
235
236
237
238
239
239
240
241
242
243
244
245
245
246
247
248
249
249
250
251
252
253
254
254
255
256
257
257
258
259
259
260
261
262
262
263
264
264
265
265
266
266
267
267
268
268
269
269
270
270
271
271
272
272
273
273
274
274
275
275
276
276
277
277
278
278
279
279
280
280
281
281
282
282
283
283
284
284
285
285
286
286
287
287
288
288
289
289
290
290
291
291
292
292
293
293
294
294
295
295
296
296
297
297
298
298
299
299
300
300
301
301
302
302
303
303
304
304
305
305
306
306
307
307
308
308
309
309
310
310
311
311
312
312
313
313
314
314
315
315
316
316
317
317
318
318
319
319
320
320
321
321
322
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408

```

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
private int size;  
public Vehicle(int someSize){  
    size = someSize;  
}  
  
public void printItself(){  
    System.out.println(size);  
}  
  
public int compareTo(ICanBeCompared somethingThatCanBeCompared){  
    if(somethingThatCanBeCompared instanceof Vehicle){  
        Vehicle vehicle = (Vehicle) somethingThatCanBeCompared;  
        if(this.size < vehicle.size){  
            return -1;  
        } else if (this.size == vehicle.size){  
            return 0;  
        } else {  
            return 1;  
        }  
    }  
    return 0; // to be fixed  
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)  
-3  
1  
2  
3
```

Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```
private int matriculationNumber;  
public Student(int someMatriculationNumber){  
    matriculationNumber = someMatriculationNumber;  
}  
  
public void printItself(){  
    System.out.println(matriculationNumber);  
}  
  
public int compareTo(ICanBeCompared somethingThatCanBeCompared){  
    if(somethingThatCanBeCompared instanceof Student){  
        Student student = (Student) somethingThatCanBeCompared;  
        if(this.matriculationNumber < student.matriculationNumber){  
            return -1;  
        } else if (this.matriculationNumber == student.matriculationNumber){  
            return 0;  
        } else {  
            return 1;  
        }  
    }  
    return 0; // to be fixed  
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)  
-3  
1  
2  
3
```

Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```
private int matriculationNumber;  
public Student(int someMatriculationNumber){  
    matriculationNumber = someMatriculationNumber;  
}  
  
public void printItself(){  
    System.out.println(matriculationNumber);  
}  
  
public int compareTo(ICanBeCompared somethingThatCanBeCompared){  
    if(somethingThatCanBeCompared instanceof Student){  
        Student student = (Student) somethingThatCanBeCompared;  
        if(this.matriculationNumber < student.matriculationNumber){  
            return -1;  
        } else if (this.matriculationNumber == student.matriculationNumber){  
            return 0;  
        } else {  
            return 1;  
        }  
    }  
    return 0; // to be fixed  
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)  
-3  
1  
2  
3
```

Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```
private int matriculationNumber;  
private int bodyHeight;  
public Student(int someMatriculationNumber, int someHeight){  
    matriculationNumber = someMatriculationNumber;  
    bodyHeight = someHeight;  
}  
  
public void printItself(){  
    System.out.println(matriculationNumber);  
}  
  
public int compareTo(ICanBeCompared somethingThatCanBeCompared){  
    if(somethingThatCanBeCompared instanceof Student){  
        Student student = (Student) somethingThatCanBeCompared;  
        if(this.matriculationNumber < student.matriculationNumber){  
            return -1;  
        } else if (this.matriculationNumber == student.matriculationNumber){  
            return 0;  
        } else {  
            return 1;  
        }  
    }  
    return 0; // to be fixed  
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)  
-3  
1  
2  
3
```

Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java Student.java
public class Student implements ICanBeCompared {
    private int matriculationNumber;
    private int bodyHeight;

    public Student(int someMatriculationNumber, int someHeight) {
        matriculationNumber = someMatriculationNumber;
        bodyHeight = someHeight;
    }

    public void printItself() {
        System.out.println(matriculationNumber);
    }

    public int compareTo(ICanBeCompared somethingThatCanBeCompared) {
        if(somethingThatCanBeCompared instanceof Student) {
            Student student = (Student) somethingThatCanBeCompared;
            if(this.matriculationNumber < student.matriculationNumber) {
                return -1;
            } else if (this.matriculationNumber == student.matriculationNumber) {
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 20 : 43

Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java Student.java
ICLASS Student implements ICanBeCompared {
    private int matriculationNumber;
    private int bodyHeight;

    public Student(int someMatriculationNumber, int someHeight) {
        matriculationNumber = someMatriculationNumber;
        bodyHeight = someHeight;
    }

    public void printItself() {
        System.out.println(matriculationNumber);
    }

    public int compareTo(ICanBeCompared somethingThatCanBeCompared) {
        if(somethingThatCanBeCompared instanceof Student) {
            Student student = (Student) somethingThatCanBeCompared;
            if((this.matriculationNumber + this.bodyHeight) < (student.matriculationNumber + student.bodyHeight)) {
                return -1;
            } else if ((this.matriculationNumber + this.bodyHeight) == (student.matriculationNumber + student.bodyHeight)) {
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 20 : 43

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java Student.java
package reusable;

public class InterfaceDemoMain {
    public static void main(String[] args) {
        Vehicle[] vehicle = new Vehicle[50];
        bubbleSort(foo);
        printArray(foo);

        public static void bubbleSort(ICanBeCompared[] data){
            for(int i=data.length-1; i>0; i--){
                for(int j=0; j<i; j++){
                    if(data[j].compareTo(data[j+1]) == 1){
                        //swap elements
                        ICanBeCompared backup = data[j];
                        data[j] = data[j+1];
                        data[j+1] = backup;
                    }
                }
            }
        }

        public static void printArray(ICanBeCompared[] data){
            for(int i=0; i<data.length; i++)
        }
    }
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 7 : 9

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer
InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java Student.java
package reusable;

public class InterfaceDemoMain {
    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i=0; i<vehicles.length; i++){
        }
        bubbleSort(foo);
        printArray(foo);
    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 7 : 40

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemoMain *InterfaceDemoMain ICanBeCompa... Vehicle.java Student.java
1 package reusable;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         Vehicle[] vehicles = new Vehicle[50];
7         for(int i = 0; i<vehicles.length; i++){
8             vehicles[i] = new Vehicle();
9         }
10        bubbleSort(foo);
11        printArray(foo);
12    }
13
14
15    public static void bubbleSort(ICanBeCompared[] data){
16        for(int i=data.length-1; i>0; i--){
17            for(int j=0; j< i; j++){
18                if(data[j].compareTo(data[j+1]) == 1){
19                    //swap elements
20                    ICanBeCompared backup = data[j];
21                    data[j] = data[j+1];
22                    data[j+1] = backup;
23                }
24            }
25        }
26    }
27}
```

The constructor Vehicle() is undefined

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Start | Java | Help | Window | Favorites | Tools | View | Status Bar | DE | 14:50 | 03.07.2015

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemoMain *InterfaceDemoMain ICanBeCompa... Vehicle.java Student.java
1 package reusable;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         Vehicle[] vehicles = new Vehicle[50];
7         for(int i = 0; i<vehicles.length; i++){
8             vehicles[i] = new Vehicle(50 * Math.random());
9         }
10        bubbleSort(foo);
11        printArray(foo);
12    }
13
14
15    public static void bubbleSort(ICanBeCompared[] data){
16        for(int i=data.length-1; i>0; i--){
17            for(int j=0; j< i; j++){
18                if(data[j].compareTo(data[j+1]) == 1){
19                    //swap elements
20                    ICanBeCompared backup = data[j];
21                    data[j] = data[j+1];
22                    data[j+1] = backup;
23                }
24            }
25        }
26    }
27}
```

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Start | Java | Help | Window | Favorites | Tools | View | Status Bar | DE | 14:50 | 03.07.2015

Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemoMain *InterfaceDemoMain ICanBeCompa... Vehicle.java Student.java
1 package reusable;
2
3 public class Vehicle{
4     private int size;
5
6     public Vehicle(int someSize){
7         size = someSize;
8     }
9
10    public void printItself(){
11        System.out.println(size);
12    }
13
14
15    public int compareTo(ICanBeCompared somethingThatCanBeCompared){
16        if(somethingThatCanBeCompared instanceof Vehicle){
17            Vehicle vehicle = (Vehicle) somethingThatCanBeCompared;
18            if(this.size < vehicle.size){
19                return -1;
20            } else if (this.size == vehicle.size){
21                return 0;
22            } else {
23                return 1;
24            }
25        }
26        return 0; // to be fixed
27    }
28
29
30 }
```

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Start | Java | Help | Window | Favorites | Tools | View | Status Bar | DE | 14:50 | 03.07.2015

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemoMain *InterfaceDemoMain ICanBeCompa... Vehicle.java Student.java
1 package reusable;
2
3 public class InterfaceDemoMain {
4
5     public static void main(String[] args) {
6         Vehicle[] vehicles = new Vehicle[50];
7         for(int i = 0; i<vehicles.length; i++){
8             vehicles[i] = new Vehicle((int)(50 * Math.random()));
9         }
10        bubbleSort(vehicles);
11        printArray(foo);
12    }
13
14
15    public static void bubbleSort(ICanBeCompared[] data){
16        for(int i=data.length-1; i>0; i--){
17            for(int j=0; j< i; j++){
18                if(data[j].compareTo(data[j+1]) == 1){
19                    //swap elements
20                    ICanBeCompared backup = data[j];
21                    data[j] = data[j+1];
22                    data[j+1] = backup;
23                }
24            }
25        }
26    }
27}
```

Console <terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)

```
-3
1
2
3
```

Start | Java | Help | Window | Favorites | Tools | View | Status Bar | DE | 14:50 | 03.07.2015

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

- Top Bar:** Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse. Includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Toolbar:** Standard Eclipse toolbar with icons for New, Open, Save, Cut, Copy, Paste, Find, etc.
- Quick Access:** A search bar at the top right.
- Java Perspective:** The main workspace area.
- Left Sidebar:** Package Explorer showing the project structure. The `src` folder contains `original`, `reusable`, `ICanBeCompared.j`, `InterfaceDemoMain`, `Student.java`, and `Vehicle.java`. Other projects like `javaUebung1` and `javaUebung2` are also listed.
- Code Editor:** Displays the `InterfaceDemoMain` class with its main method and a bubble sort implementation for the `ICanBeCompared` interface.

```
1 public class InterfaceDemoMain {  
2  
3     public static void main(String[] args) {  
4         Vehicle[] vehicles = new Vehicle[50];  
5         for(int i = 0; i<vehicles.length; i++){  
6             vehicles[i] = new Vehicle((int)(50 * Math.random()));  
7         }  
8         printArray(vehicles);  
9         bubbleSort(vehicles);  
10        System.out.println("-----");  
11        printArray(vehicles);  
12    }  
13  
14    public static void bubbleSort(ICanBeCompared[] data){  
15        for(int i=data.length-1; i>0; i--){  
16            for(int j=0; j; j++){  
17                if(data[j].compareTo(data[j+1]) == 1){  
18                    //swap elements  
19                    ICanBeCompared backup = data[j];  
20                    data[j] = data[j+1];  
21                    data[j+1] = backup;  
22                }  
23            }  
24        }  
25    }  
26 }  
27
```

- Console View:** Shows the output of the application.

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)  
-3  
1  
2  
3
```
- Bottom Status Bar:** Writable, Smart Insert, 10 : 13, and other system status indicators.
- System Tray:** Shows the date (03.07.2015), time (14:52), battery level, signal strength, and other system icons.

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

- Top Bar:** Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse. Includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help menus.
- Toolbar:** Standard Eclipse toolbar with icons for file operations, search, and project navigation.
- Quick Access:** A search bar at the top right.
- Java/Debug Buttons:** Buttons for switching between Java and Debug perspectives.
- Package Explorer:** Shows the project structure with packages like BankAccount, BeesAndFlowers, Bicycle, ControlFlowDemo, DritteJebung, Eratosthenes, Exceptions, Fakultaet, Floodfill, Histogram, ImageDemo, InterfaceDemo, InterfaceDemoNew, and a src folder containing original, reusable, ICanBeCompared, InterfaceDemoMain, Student, and Vehicle classes.
- Console:** Shows the output of the InterfaceDemoMain application, which includes the main method code and the output of the bubbleSort method.
- Code Editor:** The InterfaceDemoMain.java file is open, displaying Java code for generating vehicles and performing bubble sort on them.

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

- Project Explorer (left):** Shows the project structure with the **InterfaceDemoNew** project expanded. Inside, there's a **src** folder containing **original**, **InterfaceDemoMain**, **ICanBeCompa...**, **InterfaceDemoMain**, **Student.java**, and **Vehicle.java**.
- Editor (center):** Displays the **InterfaceDemoMain** class. The code implements the **ICanBeCompared** interface and contains a static main method that sorts an array of vehicles using bubble sort.
- Console (bottom-left):** Shows the output of the application running as a Java application.
- Bottom Status Bar:** Shows the status bar with "Writable" and "Smart Insert" buttons, and the current time as 14:52 03.07.2015.

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

- Title Bar:** Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse
- Menu Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help
- Toolbar:** Standard Eclipse toolbar icons.
- Quick Access:** A search bar at the top right.
- Java Perspective:** The main workspace area.
- Package Explorer:** Shows the project structure with packages like BankAccount, BeesAndFlowers, BicycleDemo, ControlFlowDemo, DritteJubung, Erathostenes, Exceptions, Fakultaet, Floodfill, Histogram, ImageDemo, InterfaceDemo, and InterfaceDemoNew. Under InterfaceDemoNew, there is a 'src' folder containing 'original' and 'reusable' packages, each with their respective Java files.
- Code Editor:** Displays the content of InterfaceDemoMain.java. The code includes imports for java.util, a main method that initializes an array of Vehicle objects, prints them, sorts them using bubbleSort, and prints them again; and a static bubbleSort method for Comparable objects.
- Console:** Shows the output of the executed code, indicating the application has terminated successfully.
- Bottom Status Bar:** Shows the status 'Writable', 'Smart Insert', the current time '17 : 39', and a battery icon.
- System Tray:** Shows standard Windows system tray icons.

Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemo... InterfaceD... ICanBeCompa... Vehicle.java Student.java
7
8 public Student(int someMatriculationNumber, int someHeight){
9     matriculationNumber = someMatriculationNumber;
10    bodyHeight = someHeight;
11 }
12
13 public void printItself(){
14     System.out.println(matriculationNumber);
15 }
16
17 public int compareTo(ICanBeCompared somethingThatCanBeCompared){
18     if(somethingThatCanBeCompared instanceof Student){
19         Student student = (Student) somethingThatCanBeCompared;
20         if((this.matriculationNumber + this.bodyHeight) < (student.matriculationNumber + student.bodyHeight))
21             return -1;
22         else if (this.matriculationNumber == student.matriculationNumber){
23             return 0;
24         } else {
25             return 1;
26         }
27     }
28     return 0; // to be fixed
29 }
```

Console

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
29
34
```

Writable Smart Insert 20 : 113

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemo... InterfaceD... ICanBeCompa... Vehicle.java Student.java
5
6 public static void main(String[] args) {
7     Vehicle[] vehicles = new Vehicle[50];
8     for(int i = 0; i<vehicles.length; i++){
9         vehicles[i] = new Vehicle((int)(50 * Math.random()));
10    }
11    printArray(vehicles);
12    bubbleSort(vehicles);
13    System.out.println("-----");
14    printArray(vehicles);
15
16    Student[] students = {
17        new Student(1234, 184),
18        new Student(1235, 160),
19        new Student(1236, 190)
20    };
21    printArray(students);
22    bubbleSort(students);
23    System.out.println("-----");
24    printArray(students);
25
26 }
```

Console

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
29
34
```

Writable Smart Insert 25 : 1

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemo... InterfaceD... ICanBeCompa... Vehicle.java Student.java
5
6 public static void main(String[] args) {
7     Vehicle[] vehicles = new Vehicle[50];
8     for(int i = 0; i<vehicles.length; i++){
9         vehicles[i] = new Vehicle((int)(50 * Math.random()));
10    }
11    printArray(vehicles);
12    bubbleSort(vehicles);
13    System.out.println("-----");
14    printArray(vehicles);
15
16    Student[] students = {
17        new Student(1234, 184),
18        new Student(1235, 160),
19        new Student(1236, 190)
20    };
21    printArray(students);
22    bubbleSort(students);
23    System.out.println("-----");
24    printArray(students);
25
26 }
```

Console

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
29
34
```

Writable Smart Insert 14 : 1

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java Debug
Package Explorer InterfaceDemo... InterfaceD... ICanBeCompa... Vehicle.java Student.java
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
```

Console

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:55:39)
1418
1395
1426
-----
1395
1418
1426
```

Writable Smart Insert 14 : 1

Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java Debug

Package Explorer InterfaceDem... InterfaceDem... ICanBeCompa... Vehicle.java Student.java

```
23     printArray(students);
24     //
25
26
27 }
28
29 public static void bubbleSort(ICanBeCompared[] data){
30     for(int i = data.length-1; i > 0; i--){
31         for(int j = 0; j < i; j++){
32             if(data[j].compareTo(data[j+1]) == 1){
33                 //swap elements
34                 ICanBeCompared backup = data[j];
35                 data[j] = data[j+1];
36                 data[j+1] = backup;
37             }
38         }
39     }
40 }
```

Console

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:55:39)
1418
1395
1426
-----
1395
1418
1426
```

Writable Smart Insert 14 : 1

Start DE 14:55 03.07.2015

The screenshot shows the Eclipse IDE interface. The central part is the Java editor displaying a class named 'InterfaceDemoMain'. The code implements a 'bubbleSort' method that sorts an array of objects implementing the 'ICanBeCompared' interface. The 'compareTo' method is used to compare elements. The 'printArray' method is also present. Below the editor is a terminal window showing the execution of the application, which prints three integer values: 1418, 1395, 1426, followed by a separator line and then the same values again. At the bottom of the screen is the Windows taskbar, which includes icons for Start, Task View, File Explorer, Edge browser, and others, along with the system clock showing 14:55 and the date 03.07.2015.