Innopolis University Software Architecture Spring 2016

Java Basics in Eclipse

Néstor Cataño **Homework – Week 01** Due: 25/Jan/2016

Installing Eclipse

You need first to install Eclipse Luna for editing, running your programs, and answering the questions below. Eclipse can be downloaded from `http://www.eclipse.org/downloads/`. I recommend to use Eclipse IDE for Java EE developers.

Source Code for Questions

I provide the Java source code needed for answering the questions below. The source code is given as an Eclipse Java project under the form of a zip file. The zip file is available in Moodle (it's called `hw-01-java.zip`). From Eclipse, you must import that file by clicking File, Import, General, Existing Projects into Workspace, Next, Select archive file, and then Browse into the zip file. You will need to compile and/or run the code to get a better insight on the questions and so to be able to justify your answers.

Installing Java in Eclipse

Instructions for installing Java in Eclipse depend on the platform (Windows, Linux, Mac OS X) you're using. Check the Internet for instructions or ask your Teaching Assistant for more precise directions.

Remember: You must justify all your answers!

Question Guidelines

Q1. Java interfaces and Java abstract classes are similar concepts with some subtle differences. You must grasp these differences from the questions at hand. Eclipse can help you discover which of the options are correct and which ones are not. Create a Java class called Shape and copy-paste the code for parts (a) to (e). Eclipse Java compiler will warn any compilation errors. Interpret those errors and justify your answers. To create a new class, e.g. class Shape, you should right-click lab_01 and then select New, Class.

Q2. Copy the code of each of the questions into the class Flower. Eclipse Java compiler will warn compilation errors. Interpret those errors and justify your answers.

Q3. Right-click class Minor, and then select Run As, Java application. You will then observe the results of the execution of program Minor in the Console window of Eclipse. You should interpret the results and justify your answers.

Q4. Edit the code of class TestDog by pasting the code of each of the questions. You will learn how type-casting works in Java.

Q5. Edit the code of class Chrome. This point exercises on Java types.

Questions

Q1. (**5 points**) Given

```
public abstract interface Shape {
 public void setColor(String s);
}
```

Which is a correct class? (choose all that apply)

(a)
```
public abstract class Square implements Shape {
 public abstract void setColor(String s);
}
```

(b)
```
public abstract class Square implements Shape {
 public abstract void setColor(String s) {}
}
```

(c)
```
public abstract class Square extends Shape {
 public void setColor(String s) {}
}
```

(d)
```
public abstract class Square implements Shape {
 public void setColor(Integer i) {}
}
```

(e)
```
public abstract class Square implements Shape {
 public void setColor(String s) {}
 public void setColor(Integer i) {}
}
```

Q2. (**5 points**) Given

```
public class Plant {
 public String getName() { return "plant"; };
 public Plant getType() { return this; }
}

public class Flower extends Plant {
 // insert code here
}

public class Tulip extends Flower { }
```

Which statement(s), inserted at line 7, will compile? (Choose all that apply)

(a) `public Flower getType() { return this; }`

(b) `public String getType() { return "this"; }`

(c) `public Plant getType() { return this; }`

(d) `public Tulip getType() { return new Tulip(); }`

(e) `public String getType() { return this; }`

Q3. (**5 points**) Given

```
public class Uber {
 public static int y = 2;
 public Uber(int x) { this(); y = y * 2; }
 public Uber() { y++; }
}

public class Minor extends Uber {
 public Minor() { super(y); y = y + 3; }
 public static void main(String[] args) {
  new Minor();
  System.out.println(y);
 }
}
```

What is the result?

(a) 6

(b) 7

(c) 8

(d) 9

(e) Compilation fails

(f) An exception is thrown

Q4. (**4 points**) Given

```
public class Dog { }

public class PitBull extends Dog { }

public class TestDog {
 public static void main(String[] args) {
  PitBull p1 = new PitBull();
  Dog dog1 = new Dog();
  Dog dog2 = p1;
  // insert code here
 }
}
```

Which, inserted at line 10, will compile? (Choose all that apply)

(a) `PitBull p2 = (PitBull) dog1;`

(b) `PitBull p2 = (PitBull) dog2;`

(c) `PitBull p2 = dog2;`

(d) None of the above statements will compile

Q5. (**5 points**) Given the following

```
public class X { public void do1() { } }
public class Y extends X { public void do2() { } }

public class Chrome {
 public static void main(String args[]) {
  X x1 = new X();
  X x2 = new Y();
  Y y1 = new Y();
  //insert code here
 }
}
```

Which, inserted at line 9, will compile? (choose all that apply)

(a) `x2.do2();`

(b) `((Y)x2).do2();`

(c) `(Y)x2.do2();`

(d) None of the statements will compile