S8354: Java from the very beginning Part III- Exercises

Ex. 1 Polynomial

In this exercise you will write and call a Java method to evaluate a degree 2 polynomial function. A degree 2 polynomial is an equation of the form

$$ax^2 + bx + c$$

1) Open the **Polynomial.java** file which is in the **Polynomial** folder.

2) The first step is to complete the body of the "for" loop in the *main* method. Locate the following comment inside main:

```
     // TO DO
     // Call the polynomial method with argument x, and save the result in
     // polynomialAtX
```

Add a line of source code which calls a method "polynomial", passing in x as the method argument, and storing the return value in polynomialAtX. You will define the polynomial method in step 3.

3) Define the polynomial method. Find the comment:

```
      // TO DO
      // Define a method called polynomial which returns the value of the
polynomial at
      // its argument, x.
      // Make use of the identity (ax + b)x + c = ax^2 + bx + c
      // Declare the method as static.
```

and declare the polynomial method immediately beneath it. We are going to make polynomial a *static*, or *class*, method since we haven't learnt about objects yet! To do this you need to add the static modifier to your method declaration:

```
      public static int polynomial( int x ) {

      }
```

4) Inside the body of the polynomial function, calculate the value of $ax^2 + bx + c$, and return the result as the return value of the method. The constants a,b and c have already been defined for you at the top of the source code.

5) Save and run the program. What is the value of the polynomial function when x = 5?

Ex. 2 JavaDoc

In this exercise you will use Eclipse to create documentation for a Java class. Eclipse relies on an external tool (that is provided by the JDK) called **javadoc**. We will need to configure Eclipse to use this tool.

1) Open the **Circle.java** file which is in the **JavaDoc** folder. Look at the code and comments.

2) Since the JRE provided by Eclipse by default does not contain the javadoc tool, it is necessary to download and install a Java Developer Kit (JDK), which is freely available from the IBM Web site. However, IBM's JDK has already been installed for this lab, so we can move onto the next step. The instructor will tell you where this has been installed.

3) To configure the Eclipse environment, from the menu bar, select **Window** -> **Preferences**. Expand the **Java** folder and select **Javadoc**. Enter the location of the javadoc in the **Javadoc command** text box. Click **OK** to apply the changes.

Now the Eclipse environment has been configured to use the javadoc tool to create documentation, we can specify exactly what documentation should be created.

4) We are only interested in generating documentation for the classes in the **JavaDoc** folder. Select the **JavaDoc** folder in the **Package Explorer** view. From the menu bar, select **File** -> **Export**.
This will display an "Export" dialog box. Select **Javadoc** from the export destination list and click **Next**.
We can decide which members will be documented. The javadoc tool will document members arranged by the following visibilities:

Private (all classes and all members)
Package (package, protected and public classes and members)
Protected (protected and public classes and members)
Public (only public classes and members)

For the purposes of this lab, select **Private**.
Take a note of the destination. This is where Eclipse will generate the documentation. The documentation could be created anywhere on the filesystem. However, to include it within the **JavaDoc** folder, replace the trailing **doc** with **JavaDoc\doc**.
Click **Finish** to complete the process and, if necessary, confirm that you would like the javadoc location updating.

The results of this operation will be seen appearing in the console view.

5) The newly-created documentation will appear in the **Package Explorer** view. This documentation is more easily viewed in the **Resource Perspective**.
Click the **Resource Perspective** icon .
Expand **JavaDoc** -> **doc** to reveal the documentation.

Look through these files to get an idea of what has been generated.

- packages.html lists all of the packages in the documented code. We didn’t declare any packages so our list is empty.

- Circle.html is the documentation for the circle class. Compare the generated documentation to the source file and see how the two compare.

- index.html displays a variety of methods for displaying the class and member information.

Java from the very beginning Part III- Example Solutions

Ex. 1 Polynomial

```
/**
 * A Java application to calculate the value of a degree 2 polynomial
 */
class Polynomial {

 // A polynomial of degree 2 in x is given by
 // a x^2 + b x + c,
 //

 // Polynomial constants a,b and c: these can be accessed by any method of
the
 // class

 private static final int a = 10;
 private static final int b = 5;
 private static final int c = -7;

 /**
  * Main method - prints a table of values for the polynomial
  */
 public static void main(String[] args) {

   int polynomialAtX;

   for(int x = 0; x <= 10; x++) {

     // TO DO
     // Call the polynomial method with argument x, and save the result in
     // polynomialAtX
     polynomialAtX = polynomial(x);

     System.out.println(x + "\t\t" + polynomialAtX);
   }

 } // end of main method


 // TO DO
 // Define a method called polynomial which returns the value of the
polynomial at
 // its argument, x.
 // Make use of the identity (ax + b)x + c = ax^2 + bx + c
 // Declare the method as static for now.

 /**
  * Calculate the polynomial function at x
  */
 public static int polynomial(int x) {
   int returnValue;
   returnValue = (((a*x) + b) * x) + c;
   return returnValue;
 }


} // end of class
```

The output of the program is:

```
0               -7
1               8
2               43
3               98
```

4	173
5	268
6	383
7	518
8	673
9	848
10	1043