Programming

in

Java

Exercise pack

EXERCISE PACK

PROGRAMMING IN JAVA

Contents

OBJECT ORIENTED PROGRAMMING

Exercise 1: OBJECT RELATIONSHIPS

1. Identify a system (either manual or computerised) with which you are familiar
2. Try to identify where at least one example of each of the three types of relationships (is –a, has – a and uses – a) may exist within this system

LANGUAGE BASICS

Exercise 2: EDITING AND COMPILING

- Use a text editor to create a text file named Ex2.java
- Code a public class named Ex2 containing a single method called **main**, which returns void and takes an array of strings as an argument. At present, the method should contain no actions.
- Save the file and compile it with **javac**. If any compile errors are encountered, fix them and recompile it.
- Run the program using the JVM **java.** The program should simply return to the command prompt.

Exercise 3: DATA DEFINITIONS

- Edit the file Ex2.java to include the following data definitions:
  - A public string constant usable throughout the class containing the name of your company
  - A character variable named Grade, usable throughout the class
  - Integers named x and y to be used only within main. Both should be initialised to 1.
  - A double named Wage, used only within main.
  - An array of double values to be used within main named Deductions. This should be created with a dimension of 10.
- Save, compile and run Ex2. It should return to the command prompt with no errors.

Exercise 4: ASSIGNMENTS AND OPERATORS

- Write, compile and test a program named Ex4.java which does the following:
  - Expects two numbers to be typed in as arguments on the command line.
  - Uses the method **Double.parseDouble** to convert these two numbers to double format. **parseDouble** expects a string value as an argument, and returns a double value.
  - Prints:
    - The sum of the numbers
    - The difference between the numbers
    - The first number divided by the second number
    - The first number multiplied by the second number
    - The square of each number. (Use n * n rather than the Math.pow method to calculate this.)

Exercise 5: PROGRAM CONTROL

- Write, compile and test a program name Ex5.java which will:
  - Expect a whole number as an argument on the command line.
  - Convert this number to an integer

- Where n is the number entered, print the ‘times table’ for n (i.e. n multiplied by each number from 1 to 12) in the following form:
  1 * n is x
  2 * n is x etc.

LIBRARY CLASSES

Exercise 6: THE CHARACTER CLASS

- Write, compile and test a program named Ex6.java which will take a single character argument on the command line If the character is lowercase, convert it to upper case and print the resulting character. If it is not, print a message stating that it is not a lowercase letter.

Exercise 7: INTEGER, DOUBLE AND MATH

- Write, compile and test a program named Ex7.java which calculates the area of a figure as follows:
  - The first command line parameter should be **R, T,** or **C**, indicating that the figure will be a rectangle, triangle or square respectively.
  - If the first parameter is **R**, the second and third parameters should be the length and width of the rectangle. Calculate and print the area as *length* multiplied by *width*.
  - If the first parameter is **T**, the second, third and fourth parameters should be **a**, **b** and **C**, where **a** and **b** are the lengths of two sides of a triangle, and **C** is the angle between them in degrees. Firstly, convert **C** will need to be converted to radians, since the Math functions all use radians. This is done using the formula **Angle in radians =C * 2π / 360**. The area is then calculated using the formula **½ab (sinC).** Print the area.
  - If the first parameter is a **C**, then the second parameter is the radius **r** of the circle. Calculate the area using the formula πr^2, and print it.

Exercise 8: STRINGS

- Write, compile and test a program named Ex8.java which will accept two strings as arguments on the command line, and output them concatenated so that the string with the highest alphabetical value is output first.

CLASSES, INTERFACES AND OBJECTS

Exercise 9: CREATING AND USING AN OBJECT

- Write a program named Ex9.java which will:
  - Take an employee name and number as arguments from the command line.
  - Create a GregorianCalendar object to hold today's date the date hired. Note: if no parameters are supplied when creating a new GregorianCalendar, its values default to today's date.
  - Create a second GregorianCalendar object to hold the date of next increment, also initialised to today's date.
  - Add one year to the date of next increment
  - Display the employee name, number, date hired and date of next increment on the screen.

Exercise 10: CLASSES WHICH DEFINE OBJECTS

- Create a source file named Ex10.java
- Define a class named StockPrice, which contains:
  - Instance variables for Stock Code, CostPrice and SellingPrice
  - Two constructors, one of which will take all three instance variables as parameters, and one of which will take only the StockCode and CostPrice, and calculate the SellingPrice at a mark up of 30% on CostPrice.
  - Accessor methods for each of the three instance variables.
  - Methods to allow cost price and selling price respectively to be set to a supplied value
  - A method which will increase both cost and selling price by a given percentage
  - A method which will calculate gross profit per sale as SellingPrice – CostPrice
- Define a public class named Ex10 which tests the StockPrice class by:
  - Creating an object from it with suitable values using the first constructor
  - Sets cost and selling prices to a different value, then retrieves and displays the new values
  - Increases the cost and selling prices by 5%, again retrieving and displaying the new values
  - Retrieves and displays the gross profit per sale
  - Creates a second object using the second constructor with suitable values
  - Retrieves and displays the values stored in the second object

Exercise 11: INHERITANCE

- Modify the program Ex10 as follows to include a class FurniturePrice which is a subclass of StockPrice. The new class should:
  - Include an additional instance variable DealerPrice
  - Include two constructors. The first will contain all four instance variables, and should call the first constructor of the superclass, as well as setting the value of DealerPrice. The second will call the

    second constructor of the superclass, and will calculate DealerPrice as CostPrice marked up by 15%
    - Contain a method to access the dealer price
    - Contain a method to set the dealer price to a given value
- Modify the class Ex10 to test this new class.

Optional:

If you have spare time at the end of this exercise, create methods for FurniturePrice as follows:

- Override the percentage price increase in the superclass with a suitable method that takes into account dealer price
- Provide an additional method which will calculate the gross profit per sale when selling to dealers.

THE JAVA DEVELOPMENT KIT

Exercise 12: USING JAVA UTILITIES

- Use the **jar** utility to download a copy of the java source libraries under your own directory. After this has completed, check to see that they have been downloaded into a directory named **src** under the current directory.
- Create a directory named **doc** underneath your current directory
- Use the **javadoc** utility to create documentation on the class BigDecimal defined in src\java\lang, placing the documentation in the directory **doc** which you created in the last step.
- Use a browser such as Internet Explorer to view this documentation

EXCEPTION HANDLING

Exercise 13: EXCEPTION HANDLING

- Modify the program Ex7 as follows:
    - Each statement that accesses the command line arguments to extract a number must be enclosed in try …. catch blocks to check for error type **NumberFormatException.**
    - If the exception occurs, the program should display a suitable error message stating which parameter caused the error, print a trace, and exit the program using the method **System.exit(0)**

THE GRAPHIC USER INTERFACE CLASSES

Exercise 14: Using JFrames

- Write a program named Ex14 which:
- Creates a JFrame type window of size 400 by 400, with title set to "Exercise 14", and icon set to \ **jdk1.2\demo\jfc\swingset\images\smallcow.gif**
  - **NB – If not using version 1.2 of Java, ask the lecturer to supply a different pathname to a suitable icon**
- **Displays a panel containing the text "This is a window!"**
- **Shuts down the program when the user presses the close button**

Exercise 15: Using Panels

- Write a program named Ex15 which will;
  - Display a Jframe type window which will close when the user presses the close button. The size and title are up to you.
  - Create a panel containing your name. Experiment with different font settings.
  - Create a second panel containing a filled 3D rectangle. The size and colour are up to you.
  - If you have extra time, experiment with different shapes.

Exercise 16: Command Buttons

- Modify the program written in exercise 15 to include three command buttons labelled Serif, Dialog and Symbol.
- Add an action listener for these buttons which will modify the font of the text panel to the font selected.

Exercise 17: Using JOptionPane

- Modify the program written as Ex4 so that instead of taking the two numbers from the command line, it uses Input Dialog boxes to obtain the numbers.

Exercise 18: Using Text Fields

- Use the program written as Ex10 as a basis for a new program Ex18
- The class Ex10 should be replaced by a class Ex18. Whereas Ex10 used actual values to create an instance of Price, Ex18 should:
  - Create a suitable JFrame type window
  - Include text boxes for all input fields
  - Allow the user to enter new price details, and confirm using a command button that he is ready to update. When he presses the button, a new instance of **price** must be created.

- Include a command button to activate a price increase. Input the percentage increase using an input dialog box. Call the appropriate methods of the price class, and display the updated prices in the appropriate text boxes.

Exercise 19: OPTIONS

Write a new program which will allow customer details to be entered. This should include:

- Text fields for account number, surname, first name and telephone number
- A text area for address
- Two groups of radio buttons; one for Male/Female, and one for Single/Married/Divorced/Widowed
- Checkboxes for Referees Supplied and Credit Rating Checked
- A command button which the user can press when all details have been entered correctly

Once all details have been entered, create a new object Customer using these details

THE INPUT/OUTPUT CLASSES

Exercise 20: Keyboard Entry

- Modify the program written as Ex7 to take its input from the keyboard rather than the command line.

Exercise 21: Data Files

- Modify the program written in Exercise 19 to write the details entered by the user away to a file. The file should be written in text format.

DATABASES

Exercise 22: Queries

- Make sure you have a copy of the Product database supplied with this course. If not, ask the lecturer to supply one. The Product database contains a single table Stock, with the following columns:
    - Stock
    - Description
    - CostPrice
    - SellingPrice
- Set up a DSN for this database through the control panel
- Write a program which will:
    - Retrieve all rows from the Stock table
    - For each row in the Result Set, display the StockCode, Description and SellingPrice.
    - Close the database connection

Exercise 23: Inserts

- Write a program which will allow the user to enter stock details via the keyboard, and insert rows in the database.