

Gujarat Technological University

Master of Computer Applications

Subject Name: Programming skills – IV (Java)

Subject Code:630006

(W.E.F. June 2012)

Suggested practical for Java:

1. Observe the interactions involved in the process of booking a railway ticket. Identify the various objects involved and the interactions between the objects in order to solve a problem of booking a railway ticket.
2. Write a simple java application to print a pyramid with 5 lines. The first line has one character, 2nd line has two characters and so on. The character to be used in the pyramid is taken as a command line argument.
3. Write a Java application which takes several command line arguments, which are supposed to be names of students and prints output as given below:
(Suppose we enter 3 names then output should be as follows)..

Number of arguments = 3

1.: First Student Name is = Tom

2.: Second Student Name is = Dick

3.: Third Student Name is = Harry

Hint: An array may be used for converting from numeric values from 1 to 20 into String.

4. Write a class, with main method, which declares floating point variables and observe the output of dividing the floating point values by a 0, also observe the effect of assigning a high integer value (8 digits and above) to a float and casting it back to int and printing.
5. Write a class called Statistics, which has a static method called average, which takes a one-dimensional array for double type, as parameter, and prints the average for the values in the array. Now write a class with the main method, which creates a two-dimensional array for the four weeks of a month, containing minimum temperatures for the days of the week(an array of 4 by 7), and uses the average method of the Statistics class to compute and print the average temperatures for the four weeks.
6. Define a class called Product, each product has a name, a product code and manufacturer name. define variables, methods and constructors, for the Product class. Write a class called TestProduct, with the main method to test the methods and constructors of the Product class.
7. Define a class called CartesianPoint, which has two instance variables, x and y. Provide the methods getX() and getY() to return the values of the x and y values respectively, a method called move() which would take two integers as parameters and change the values of x and y respectively, a method called display() which would display the current values of x and y. Now overload the method move() to work with single parameter, which would set both x and y to the

same values, . Provide constructors with two parameters and overload to work with one parameter as well. Now define a class called TestCartesianPoint, with the main method to test the various methods in the CartesianPoint class.

8. Define a class called Triangle, which has constructor with three parameters, which are of type CartesianPoint, defined in the exercise 7. Provide methods to find the area and the perimeter of the Triangle, a method display() to display the three CartesianPoints separated by ':' character, a method move() to move the first CartesianPoint to the specified x, y location, the move should take care of relatively moving the other points as well, a method called rotate, which takes two arguments, one is the CartesianPoint and other is the angle in clockwise direction. Overload the move method to work with CartesianPoint as a parameter. Now define a class called TestTriangle to test the various methods defined in the Triangle class. Similarly also define a class called Rectangle which has four CartesianPoint.
9. Override the toString, equals and the hashCode methods of the classes Triangle and Rectangle defined in exercises 7 and 8 above, in appropriate manner, and also redefine the display methods to use the toString method.
10. Define an abstract class called Polygon. Provide a constructor which takes an array of CartesianPoint as parameter. Also provide method called perimeter, which calculates and returns the perimeter of the Polygon. Declare abstract method area for this class. Also define a method called move, which takes two parameters x and y to specify the destination for the first point of the Polygon, and overload to make it work for CartesianPoint as a parameter. Now update the classes Triangle and Rectangle in the exercise 8 above, to be a subclass of the Polygon class. Write appropriate class with main method to test the polymorphism in the area method.
11. Make the class CartesianPoint, belong to a package called edu.gtu.geometry, the classes Polygon, Triangle and Rectangle belong to the package edu.gtu.geometry.shapes and the classes TestCartesianPoint, TestTriangle, TestRectangle and TestPolygon belong to the package edu.gtu.test. Use appropriate access specifiers for the classes and the members of the classes defined in the earlier exercises. Now onwards all the classes must be defined in a package.
12. Update the classes Triangle and Rectangle, to throw an exception if the CartesianPoint instances passed as parameter does not specify an appropriate Triangle or Rectangle. eg. In case of Triangle, if the three points are in a straight line, or in case of Rectangle, if the lines when connected cross each other.
13. Define a class called PolygonManager, which manages a number of Polygon instances. Provide methods to add, remove and list the Polygon instances managed by it. Test the methods of PolygonManager by writing appropriate class with main method.
14. Define a class called StatisticalData which manages a number of readings of type int. Provide a method to setData available in the form of an array, a method add to individually add data of type int, a method to reset the entire data, and methods to return the following statistics:
 1. mean
 2. median
 3. mode

Use this class to calculate the given statistics for the data regarding the total of marks obtained by all the students in your class in the previous exams. The data may be accepted one by one for each student from the keyboard.

15. Update the class `StatisticalData`, and define a method called `loadFromCSV`, which takes as parameter an `InputStream`, where numeric data is available in an ASCII format, in a comma separated form. Overload this method to take a `File` instance as parameter. Test the new methods using appropriate data.
16. A college maintains the information about the marks of the students of a class in a text file with fixed record length. Each line in the file contains data of one student. The first 25 characters have the name of the student, next 12 characters have marks in the four subjects, each subject has 3 characters. Create a class called `StudentMarks`, which has `studentName`, and marks for four subjects. Provide appropriate getter methods and constructors, for this class. Write an application class to load the file into an array of `StudentMarks`. Use the `StatisticalData` class to compute the statistics mean, median, mode for each of the subjects in the class.
17. In the above exercise, use multithreading, to compute the statistics, after loading the `StudentMarks` from the file, for marks information available for different classes available from files placed in a directory. Create atleast five files in a directory with fixed record length to test your code.
18. Dasher is an information-efficient text-entry interface, driven by natural continuous pointing gestures. Dasher is a competitive text-entry system wherever a full-size keyboard cannot be used. Dasher is a zooming interface. You point where you want to go, and the display zooms in wherever you point. The world into which you are zooming is painted with letters, so that any point you zoom in on corresponds to a piece of text. The more you zoom in, the longer the piece of text you have written. You choose what you write by choosing where to zoom. To make the interface efficient, we use the predictions of a language model to determine how much of the world is devoted to each piece of text. Probable pieces of text are given more space, so they are quick and easy to select. Improbable pieces of text (for example, text with spelling mistakes) are given less space, so they are harder to write. The language model learns all the time: if you use a novel word once, it is easier to write next time.

Write a class in Java which models the probabilities for the alphabets in a language as the starting alphabet and then the probabilities for the subsequent alphabets which may follow the alphabet and so on leading to the end of word. The probabilities may be built by reading a text file containing lots of texts in the given language. (Hint: create a class to represent a node, which has an alphabet and its probability and each node has capability of maintaining a list of subsequent nodes). (*Advanced question*)

19. Create an applet which has a `TextField` to accept a URL string, and displays the document of the URL string in a new browser window.
20. Consider two types of residency, a Flat or a Villa. All types of residences have an area (square yards) and a rate (per square yard). The property price of a residency is by default calculated as $\text{area} * \text{rate}$. In case of Flat, the price get incremented by the maintenance charges, and in case of Villa the price is incremented by furniture charges. Now define the following classes in a

common package called “**residence**”.

1. An abstract class called **Residency**, with appropriate methods and constructors.
 2. Two sub-classes called **Flat** and **Villa**, which inherit from the **Residence** class and override the appropriate methods, from **Residency** class.
 3. Also override appropriate methods from the Object class.
21. Consider the above exercise and define the following:
1. Appropriate class(es) implementing the Comparator interface to make comparison of Residence based on the area, rate or price. These class(es) may be defined in the “**residence.util**” package.
 2. A class called **PropertyList** as part of the “**residence**” package, which maintains various Residence objects in an appropriate data structure. has methods to add residence, remove residence, get residence list in price range, get residence list in area range.
22. Write a program that takes names of a text files as command line argument and searches the files for occurrence of palindromes. The output should print all the occurrences of palindromes in the file, with their filename and line numbers. The end of line in the file will be marked by the newline character, '\n'. (A palindrome is a word which has the same spelling when read from left to right or right to left). Use multithreading to process files in parallel.
23. Create an applet named “**UnitConversion**”, which allows user to select a particular conversion from following options.(Use list)
1. Decimal to HexaDecimal [Use Integer.toHexString()].
 2. Decimal to Octal [Use Integer.toOctalString()].
 3. Feet to Centimeter.(1 feet = 30.48cm)
 4. Inches to Feet (1 feet = 12 inches)

Once user selects particular conversion then show the converted value with proper formatted message. (Like if user selects Inches to Feet option and input value is 60 then message should be “60 inches is equal to 5 Feet”.)

If user enters any wrong kind of input then prompt proper errorDialog and clear the entered value.

24. Write a program that can create an index for a document. The document should be read from an input file, and the index data should be written to an output file. The names of the input file and output file should be specified as command line arguments when the program is run. Also take the number of lines per page and number of characters per line as command line arguments. (Assume there is no hyphenation and words are wrapped). The end of line in the file may also be marked by the newline character, '\n'. (Try to use multiple threads to speed up the indexing process, you can take the number of threads to use as a command-line argument, default value can be four threads.)

Because it is so common, don't include the word "the" in your index. Also, do not include words that have length less than 3.

25. Write a program that can search all duplicate files in a disk. The duplicate file is one which has the same file name, file size and last modified date. The contents of the files need not be

compared. ie. you need to locate files occurring in different directories, which are duplicates. The output should display the last modified date, file size and the full path of the duplicate files grouped together.

26. A software reseller maintains inventory for various software products. The reseller receives boxes containing packets with the software licence and media. Each packet has a unique serial number. The serial numbers within a Box are in a sequence, ie. the box contains the from serial number and to serial number. The reseller usually sells these products in terms of packet. Write classes called Product, Box, Packet and ProductInventory, to track the total inventory, in terms of Box and Packets.
27. A DNA chain is double stranded, where each strand is a long sequence of four kind of bases, Adenine (A), Guanine (G), Cytosine (C) and Thymine (T). One end of the strands is known as 5'(5 prime) and the other end is known as 3'(3 prime). The reading of the strand can be done from 5' to 3' or 3' to 5'. The bonding between the two strands is always done as A on one strand is bonded with T on the opposite strand, and C on one strand is bonded with G on the opposite strand. So a DNASquence is stored as sequence of bases, either from 5' to 3' or 3' to 5'. Define an enum for the Bases, and another enum for the strand direction, and then define a class called DNASquence, which stores the sequence of bases as a String and the strand (5' to 3', or 3' to 5'). Define a method which return the String of bases from the DNASquence, by specifying the strand for which the sequence is required. ie. The DNASquence class may have the following methods.
 1. public String getSequence()
 2. public Strand getStrand()
 3. public String getSequence(Strand s)Here Strand is an enum with two values FIVE_PRIME_TO_THREE_PRIME and THREE_PRIME_TO_FIVE_PRIME.
28. Write an application to convert digits into words, use appropriate structures available from java.util package.
29. An educational institution wishes to maintain a database of its employees. The database is divided into a number of classes whose hierarchical relationships are as follows:
 - Staff (code, name) is the base class, which in turn has teacher (subject, publication), typist (speed) and officer (grade) as its child classes. The typist again has regular () and casual (daily Wages) as its child classes. Note that the information given in brackets specifies the minimum information required for each class. Specify all classes and define functions to create the database and retrieve information as and when needed.
30. Define an abstract base class **Shape** that includes *protected* data members for the (x, y) position of a shape, a *public abstract* method to compute the area of the shape, and a *public abstract* method paint() to draw the shape on a Graphics object, which is passed as parameter. Derive subclasses for Point, Line, Circle and Rectangle. You can represent – a line as two points, a circle as a centerPoint and a radius, and

a Rectangle as two points on diagonally opposite corners. Now define a class called DrawingBoard, which extends the Canvas class and maintains instances of the Shape objects in a List, also oerrides the paint method to draw the Shape instances maintained in the List,