**LOK JAGRUTI UNIVERSITY (LJU)**

**INSTITUTE OF ENGINEERING & TECHNOLOGY**

**Department of Computer Science and Engineering (712)**

**Bachelor of Engineering (B.E.) – Semester - VI**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Code:** | **017124691** |  | **Teaching Scheme** | | | | |
| **Course Name:** | **Object Oriented Programming using Java** |  | **Lecture (L)** | **Tutorial (T)** | **Practical (P)** | **Credit** | **Total Hours** |
| **Category of Course:** | PEC |  | 4 | **0** | **4** | **6** | 4 |
| **Prerequisite Course:** | -- |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Syllabus** | | | | |
| **Unit No.** | | **Topic** | **Prerequisite Topic** | **Successive Topic** | **Teaching Hours** |
| **01** | | **Basics of Java** | | | **5**  **(12%)** |
| * 1. Features of Java, Byte Code and Java Virtual Machine, JDK, JRE, Basic Structure of Program, Identifiers, Literals | --- | --- |
| * 1. Data Types- Primitive | Non-Primitive, Typecasting, Operators, Evaluation of Expression, Precedence and Associativity | --- | --- |
| * 1. Command line argument | --- | --- |
| * 1. Scanner class and its methods: (nextInt(), nextDouble(), nextFloat(), nextBoolean(), next(),nextLine()). |  |  |
| * 1. if, If-else, Nested if, if else…if ladder, Switch Statement |  |  |
| * 1. while loop, for loop, do-while, Break and Continue Statements |  |  |  |
|  |  | | | | |
| **02** | | **Arrays and Strings** | | | **5**  **(12%)** |
| 2.1 One Dimensional Arrays, Two Dimensional Arrays, for each loop | --- | --- |
| 2.2 String class and its methods (charAt(), length(), concat(),  equals(), equalsIgnoreCase(), compareTo(), compareToIgnoreCase(), toUpperCase(), toLowerCase(), split(),  replace(),toString(),startsWith(),endsWith(),indexOf(),toCharArr  ay(),trim()) | --- | --- |
| 2.3 StringBuffer class and its method (append(), insert(), replace(),  delete(), reverse()) | --- | --- |
| 2.4 Difference between String and StringBuffer class | --- | --- |
|  |  | | | | |
| **03** | | **Class, Object and Method** | | | **3**  **(12%)** |
| 3.1 General form of class, Declaring object, Scope & Life time of Variable, Static variable, Instance variable | Data Types (Unit-1) | --- |
| 3.2 Method declaration and calling, method overloading | --- | --- |
| 3.3 Passing Arrays, array of objects to method | One Dimensional Arrays (Unit-2) | --- |
| 3.4 Call by Value and Call by Reference | --- |  |  |
| 3.5 Returning object | --- |  |  |
|  |  | | | | |
| **04** | | **Constructor** | | | **5**  **(8%)** |
| 4.1 Default, Parameterized and Copy Constructor | Class and Method (Unit-3) | --- |
| * 1. Constructor Overloading, this keyword | --- | --- |
| 4.3 static keyword: method, class, variable, block | --- | --- |
|  |  | | | | |
| **05** | | **Inheritance and Package** | | | **5**  **(10%)** |
| 5.1 Introduction of inheritance and its types | --- | --- |
| 5.2 Inheriting Data members and Methods | Class and Method (Unit-3) | --- |
| 5.3 Method overriding, super and final keyword | --- | --- |
| 5.4 Use of Package, Import statement, Access Modifiers | --- | --- |
|  |  | | | | |
| **06** | | **Runtime Polymorphism and Exception handling** | | | **5**  **(14%)** |
| 6.1 Abstract class, Abstract Method | Inheriting Data members (Unit-5) | --- |
| 6.2 Interface, extending Interface | --- | --- |
| 6.3 Dynamic method dispatch, Object casting and instance of operator |  | --- |
| 6.4 Types of Errors, checked and unchecked Exception | --- | --- |
| 6.5 Exception handling mechanism (try, catch, throw, throws and finally) | --- | --- |
| 6.6 Built in Exception, Custom Exception |  |  |  |
|  |  | | | | |
| **07** | | **Multithreaded Programming** | | | **5**  **(12%)** |
| 7.1 Thread life cycle | --- | --- |
| 7.2 Thread class and Runnable interface | --- | --- |
| 7.3 Thread methods: start(), run(), getName(), setName(), sleep(), join(), isAlive(), wait(), notify(), currentThread() | --- | --- |
| 7.4 Synchronized Methods and Synchronized Blocks, | --- | --- |
| 7.5 Producer – Consumer Problem solution using wait() & notify() |  |  |  |
|  |  | | | | |
| **08** | | **Collection-I** | | | **4**  **(7%)** |
| 8.1 Collection framework Hierarchy | --- | --- |
| 8.2 Classes and interfaces in collections, Methods of **Collection interface**: add(), addAlll(), clear(), contains(), isEmpty(), iterator(), remove(), removeAll(), toArray(), |  |  |
| 8.3 **List Interface: ArrayList**: add(int index, E element), add(E e), clear(), ensureCapacity(int requiredCapacity), get(), set(), isEmpty(), lastIndexOf(Object o), remove(int index), sort(),size(), for each loop to print all elements indexOf(), lastIndexOf() Add Multiple element with Arrays.asList() in Constructor. | --- | --- |
|  |  | | | | |
| **09** | | **Collection-II** | | | **3**  **(7%)** |
| 9.1 **Collections class :** frequency(), reverse(), max(), min(), sort(), Comparator.comparing() to sort class objects by properties |  |  |
| 9.2 **Queue Interface:** **Priority Queue** class : add(object), offer(object), remove(), poll(), element(), peek(): Print Queue using iterator |  |  |
| 9.3 **Dequeue Interface:** **ArrayDeque** class: ArrayDeque creation, add(Element e), addAll​(Collection<? extends E> c), addFirst(Element e), addLast(Element e), clear(), getFirst(), getLast(), isEmpty(), offerFirst(Element e), offerLast(Element e), peek(), remove(), removeFirst(), removeLast(), size() | --- | --- |
| 9.4 **Set Interface:** **HashSet**: add(), clear(), remove(), isEmpty(), size(), removeAll(), addAll(), equals(), print using iterator  Map Interface: **HashMap**: create HashMap, size(), isEmpty(), remove(), put(), putAll(), getKey(), getValue(),print using iterator or foreach **HashTable:** put(), remove(), containsKey(), clear(),getKey(), getValue(),print using iterator or foreach | --- | --- |
|  |  | | | | |
| **10** | | **IO Programming** | | | **1**  **(6%)** |
| 10.1 Introduction to Stream, Byte Stream, Character stream | --- | --- |
| 10.2 File Class and its method, constructor of File Class, methods like : canExecute(), canRead(), createNewFile(), equals(), exists(), getAbsolutePath(), getName(), getParent(), getParentFile(), getPath(), isDirectory(), isFile(), length(), listFiles(), mkdir(), list(). | **---** | --- |
| 10.3 File Input Stream, File Output Stream |  |  |  |
| 10.4 Readers and Writers class, FileReader, FileWriter |  |  |  |
| 10.5 Buffered Reader, InputStreamReader, |  |  |  |
| 10.6 RandomAccessFile with constructor and methods: close(), readInt(), readUTF(), seek(), writeDouble(), writeFloat(), write(), read(), length(), getFilePointer() |  |  |  |

|  |  |
| --- | --- |
| **Major Components/ Equipment** | |
| **Sr. No.** | **Component/Equipment** |
| 1 | Computer |
| 2 | JDK, JRE, VS CODE |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Proposed Theory + Practical Evaluation Scheme by Academicians  (% Weightage Category Wise and it’s Marks Distribution)** | | | | | |
| **L:** |  | **T:** | **0** | **P:** |  |
| **Note: In Theory Group, Total 4 Test (T1+T2+T3+T4) will be conducted for each subject.  Each Test will be of 25 Marks. Each Test Syllabus Weightage: Range should be 20% - 30%** | | | | | |
| **Group (Theory or Practical)** | **Group (Theory or Practical) Credit** | **Total Subject Credit** | **Category** | **% Weightage** | **Marks Weightage** |
| Theory |  | **5** | MCQ | 6% | 8 |
| Theory | Theory Descriptive (Mainly Algorithms) | 32% | 40 |
| Theory | Formulas and Derivation (Mainly Complexity & Analysis) | 16% | 20 |
| Theory | Numerical | 26% | 32 |
| **Expected Theory %** | **50%** | **Calculated Theory %** | **80%** | **100** |
| Practical |  | Individual Project | 4% | 20 |
| Practical | Group Project | 4% | 20 |
| Practical | Internal Practical Evaluation (IPE) | 8% | 40 |
| Practical | Viva | 4% | 20 |
| Practical | Seminar | 0% | 0 |
| **Expected Practical %** | **5**  **0%** |  | **Calculated Practical %** | **20%** | **100** |
| **Overall %** | **100%** |  |  | **100%** | **200** |

|  |  |
| --- | --- |
| **Course Outcome** | |
| 1 | Understand the concepts of basic Java programming, packages and interface |
| 2 | Apply multi-threading and file handling using Java. |
| 3 | From and Manipulate collections of data. (such as Lists, set and map) |
| 4 | Understand the concepts of basic Java programming, multi-threading and collections in java. |
| **Suggested Reference Books** | |
| 1 | Java: The Complete Reference, Tenth Edition (Complete Reference Series), Herbert Schildt – McGrawHill |
| 2 | Java Server Programming Java EE 7 (J2EE 1.7), Black Book Kindle Edition, Kogent Learning Solutions Inc – Dreamtech |
| 3 | Core Java Volume I--Fundamentals, 11th edition, Cay Horstman – Pearson |
| 4 | Core Java - An Integrated Approach Includes All Version Upto Java 8, Dr. R. Nageswara Rao – Dreamtech |
| 5 | Programming with Java by, E Balagurusamy – McGrawHill |

|  |  |
| --- | --- |
| **List of Open Source Software/Learning website** | |
| 1 | <https://www.javatpoint.com/java-tutorial> |
| 2 | <https://www.tutorialspoint.com/java/index.htm> |
| 3 | https://www.geeksforgeeks.org/java/ |
| 4 | https://www.oracle.com/java/technologies/downloads/#jdk17-windows & https://notepad-plus-plus.org/downloads/ |
| 5 | https://www.programiz.com/java-programming/ |

|  |  |  |
| --- | --- | --- |
| **Sr No.** | **Practical Title** | **Link to Theory Syllabus** |
| 1 | WAP to calculate the Area of a Circle, Area and Perimeter of rectangle, Area of Triangle. | Unit-1 |
| 2 | WAP that reads two nos. from key board and gives their addition, subtraction, multiplication, division and  modulo. | Unit-1 |
| 3 | WAP to enter three subject marks, and calculate total, percentage of student and display the same in proper  format. | Unit-1 |
| 4 | WAP to convert days into months and days. | Unit-1 |
| 5 | WAP that converts Fahrenheit temperature to centigrade and vis-a-versa Centigrade =(5\* (fahr.-32))/9 | Unit-1 |
| 6 | WAP to swap two numbers with and without using temporary variable | Unit-1 |
| 7 | WAP to determine a given number is ‘odd’ or ‘even’ and print the following message “Number is ODD” or  “Number is Even” (i) Without using else option. (ii) With else option | Unit-1 |
| 8 | WAP to accept three numbers from user and Print Maximum number | Unit-1 |
| 9 | WAP to print grade of a student using following rules: Percentage >70 means Grade A  Percentage 60-70 means Grade B Percentage 50-60 means Grade C Percentage <50 means Grade F | Unit-1 |
| 10 | WAP to perform addition, multiplication, subtraction and division with Switch statement. | Unit-1 |
| 11 | WAP that reads number from 1 to 7 and accordingly it should print MONDAY to SUNDAY. | Unit-1 |
| 12 | WAP to enter a character and check whether it is a vowel or consonant using switch statement | Unit1 |
| 13 | WAP to check whether entered character is vowel or not. | Unit-1 |
| 14 | WAP to display “Hello” five times. | Unit-1 |
| 15 | WAP to display 1-10 numbers, 20-30 numbers | Unit-1 |
| 16 | WAP to display multiplication table. | Unit-1 |
| 17 | Assume that you want to make the sum of 1 to 100. Give the necessary code to perform the same using (1)  For loop (2) While loop (3) Do-while loop | Unit-1 |
| 18 | WAP for finding sum of 1 to k. The number k should be read from the keyboard using Command line  argument. | Unit-1 |
| 19 | WAP to print multiple of N from given range of unsigned integers. For example, if N=5 and range is [17, 45]  it prints 20, 25, 30, 35, 40, 45. Take input using Scanner class | Unit-1 |
| 20 | WAP to find sum of all integers greater than 100 & less than 200 and are divisible by 5. | Unit-1 |
| 21 | WAP to count ODD and EVEN numbers from given 10 numbers | Unit-1 |
| 22 | WAP to find the sum of first N odd numbers. | Unit-1 |
| 23 | WAP to find 1+3/5+5/7+7/9+… series. Print addition of first N part. | Unit-1 |
| 24 | WAP to find 1+1/2+1/3+1/4+………+1/N series. | Unit-1 |
| 25 | WAP to find Factorial of a number. | Unit-1 |
| 26 | WAP to generate Fibonacci series of numbers | Unit-1 |
| 27 | WAP to reverse a number. | Unit-1 |
| 28 | WAP to calculate sum of digits (Ex: 123 => so sum of digit = 1+2+3=6) | Unit-1 |
| 29 | WAP to find out Armstrong Numbers. Example: - 153 is an Armstrong Number. | Unit-1 |
| 30 | WAP to check whether the given number is Prime or not. OR Write a Java program to find and print prime  numbers between the numbers 1 to n, where the number n should be read from the keyboard. | Unit-1 |
| 31 | WAP to check whether a number is a perfect number or not. (e.g. 123 is a perfect no i.e.1+2+3=1\*2\*3) | Unit-1 |
| 32 | WAP to find out sum of first and last digit of a given number | Unit-1 |
| 33 | WAP to print following pattern using loop statement for n row.  \*  \* \*  \* \* \*  \* \* \* \*  \* \* \* \* \*  1  1 3  1 3 5  1 3 5 7  1  0 1  1 0 1  0 1 0 1  1  2 3  4 5 6  7 8 9 10  \*  # #  \* \* \*  # # # #  \* \* \* \* \*  # # # # # #  \* \* \* \* \* \* \*  1  2 2  3 3 3  4 4 4 4  5 5 5 5 5  \* \* \* \* \*  \* \* \* \*  \* \* \*  \* \*  \*  54321  4321  321  21  1  1 2 3 4 5  2 3 4 5  3 4 5  4 5  5 | Unit-1 |
| 34 | WAP to Print following pattern using loop statement for n row  \*  \* \*  \* \* \*  \* \* \* \*  1  A B  1 2 3  A B C D  1 2 3 4 5  1  A B  2 3 4  C D E F  5 6 7 8 9  1  2 2  3 3 3  4 4 4 4  1  1 2 3  1 2 3 4 5  1 2 3 4 5 6 7  1 2 3 4 5 6 7 8 9 | Unit-1 |
| 35 | WAP to read 10 numbers from user and find Sum, Maximum, Minimum and Average of them | Unit-1 |
| 36 | WAP which declares array of 10 integers, enter data and sum all the elements which are even. Also find  maximum number from them. | Unit-1 |
| 37 | WAP to accept array of N integers and find Largest odd number as well as largest even number and display  them. | Unit-1 |
| 38 | WAP read in an array of integers and print its elements in reverse order. | Unit-1 |
| 39 | WAP to add two 3 x 3 Matrix or 2x2 matrix | Unit-2 |
| 40 | WAP to find maximum element from 3\*3 Matrices | Unit-2 |
| 41 | WAP to find the minimum value from the array of 3 x 3. | Unit-2 |
| 42 | WAP to count number of positive, negative and zero elements from 3 x 3 matrix. | Unit-2 |
| 43 | WAP to read two matrix from the user and store the multiplication of two matrix in the resultant matrix. i.e.  C=A \* B | Unit-2 |
| 44 | WAP to display transpose of given 3\*3 matrix. | Unit-2 |
| 45 | WAP to check given string is palindrome or not. | Unit-2 |
| 46 | WAP to reverse the input string. | Unit-2 |
| 47 | WAP to concatenate two strings without using built in function. | Unit-2 |
| 48 | WAP to accept a string and count the number of vowels present in a string | Unit-2 |
| 49 | Write method headers (not the bodies) for the following methods:   1. Return a sales commission, given the sales amount and the commission rate. 2. Display the calendar for a month, given the month and year. 3. Return a square root of a number. 4. Test whether a number is even, and returning true if it is. 5. Display a message a specified number of times. 6. Return the monthly payment, given the loan amount, number of years, and annual interest rate. 7. Return the corresponding uppercase letter, given a lowercase letter. | Unit-2 |
| 50 | A pentagonal number is defined as n(3n–1)/2 for n = 1, 2, . . ., and so on. Therefore, the first few numbers are 1, 5, 12, 22, Write a method with the following header that returns a pentagonal number:  public static intgetPentagonalNumber(int n)  Write a test program that uses this method to display the first 100 pentagonal numbers with 10 numbers on each line. | Unit-2 |
| 51 | Write a method with the following header to display an integer in reverse order:  public static void reverse(int number)  For example, reverse(3456) displays 6543. Write a test program that prompts the user to enter an integer and displays its reversal. | Unit-2 |
| 52 | WAP to find GCD of the 2 numbers using recursion | Unit-2 |
| 53 | WAP to calculate nCr using recursion. nCr = n! / (r! \* (n-r)!) | Unit-2 |
| 54 | WAP to generate Fibonacci series of n given numbers using recursion. | Unit-2 |
| 55 | WAP to find sum of digit of given number using recursion. | Unit-2 |
| 56 | Write a program which takes five numbers as command line argument from user, store them in one  dimensional array and display count of negative numbers. | Unit-2 |
| 57 | Write a program that creates and initializes a four integer element array. Calculate and display the average of  its values. | Unit-2 |
| 58 | Write a program that prompts the user to enter the number of students, the students’ names, and their scores,  and prints student names in decreasing order of their scores. | Unit-3 |
| 59 | Write a program that creates an integer array and then uses a for loop to check whether the array is sorted  from smallest to largest. If so, it prints “sorted” otherwise it prints “Not sorted” | Unit-3 |
| 60 | Write a java program to calculate total and average to five values. Pass input values as constructor  parameter. | Unit-3 |
| 61 | Write a class named Rectangle to represent a rectangle. It contains following members: DATA: width(double) and height (Double) that specify the width and height of the rectangle. Methods:   1. A no-arg constructor that creates a default rectangle. 2. A constructor that creates a rectangle with the specified width and height. 3. A method named getArea() that returns the area of this rectangle. 4. A method named getPerimeter() that returns the perimeter. | Unit-4 |
| 62 | Write a JAVA program to create a super class called figure that storesthe dimensions of a two dimensional object. It also defines a method called area () that computes the area of an object.  The program derives two sub classes from figure. The first is rectangle and the second is Triangle. Each of these subclasses overrides area (), so that it returns the area of a rectangle and a triangle respectively | Unit-5 |
|  |  |
| 63 | Write an application that generates custom exception if any value from its command line arguments is negative. | Unit – 1 |
| 64 | Write a method for computing xy by doing repetitive multiplication. x and y are of type integer and are to be given as command line arguments. Raise and handle exception(s) for invalid values of x and y. Also define method main. Use finally in above program and explain its usage. | Unit – 6 |
| 65 | Write an application that generates custom exception if any value from its command line arguments is negative. | Unit – 6 |
| 66 | Write a method for computing xy by doing repetitive multiplication. x and y are of type integer and are to be given as command line arguments. Raise and handle exception(s) for invalid values of x and y. Also define method main. Use finally in above program and explain its usage. | Unit – 6 |
| 67 | It is required to maintain and process the status of total 9 resources. The status value is to be stored in an integer array of dimension 3x3. The valid status of a resource can be one of the 2 followings: free: indicated by integer value 0 occupied: indicated by integer value 1 inaccessible: indicated by integer value 2  Declare a class called ResourcesStatus, having data member called statusRef, referring to a two dimensional array (3x3) of integers to be used to refer to the above mentioned status values.  Define a member method called processStausCount that counts and displays total number of free resources, total number of occupied resources and total number of inaccessible resources. The exception to be raised and handled if total number of occupied resources exceeds total number of free resources. The handler marks status of all inaccessible resources as free. Accept initial status values from command line arguments and initialize the array. Raise and handle user defined exception if invalid status value given | Unit – 6 |
| 68 | Write a complete program to accept N integer numbers from the command line. Raise and handle exceptions for following cases:  - when a number is –ve  - when a number is evenly divisible by 10  - when a number is greater than 1000 and less than 2000  - when a number is greater than 7000  Skip the number if an exception is raised for it, otherwise add it to find total sum | Unit – 6 |
| 69 | Declare a class called book having author\_name as private data member. Extend book class to have two sub classes called book\_publication&paper\_publication. Each of these classes have private member called title. Write a complete program to show usage of dynamic method dispatch (dynamic polymorphism) to display book or paper publications of given author. Use command line arguments for inputting data. . | Unit – 6 |
| 70 | Write a program that reads file name from user, through command line argument and displays/reads content of the text file on console. | Unit – 10 |
| 71 | Write a program that reads file name from user, through command line argument and displays/reads content of the text file on console. | Unit – 10 |
| 72 | Write a program to replace all “word1” by “word2” from a file1, and output is written to file2 file and display the no. of replacement. | Unit – 10 |
| 73 | Write a program that counts the no. of words in a text file. The file name is passed as a command line argument. The program should check whether the file exists or not. The words in the file are separated by white space characters. | Unit – 10 |
| 74 | Write a program to read the content of a file into a character array and write it into another file. Get names of the files from command line | Unit – 10 |
| 75 | Read employee salary and calculate the income tax based on 10% of income and store it in tax.txt file for five different employees | Unit – 10 |
| 76 | The abstract Vegetable class has three subclasses named Potato, Brinjal and Tomato. Write an application that demonstrates how to establish this class hierarchy. Declare one instance variable of type String that indicates the color of a vegetable. Create and display instances of these objects. Override the toString() method of Object to return a string with the name of the vegetable and its color | Unit – 6 |
| 77 | Write a program that illustrates interface inheritance. Interface P is extended by P1 And P2. Interface P12 inherits from both P1 and P2.Each interface declares one constant and one method. Class Q implements P12.Instantiate Q and invokes each of its methods. Each method displays one of the constants | Unit – 5 |
| 78 | The Transport interface declares a deliver() method. The abstract class Animal is the superclass of the Tiger, Camel, Deer and Donkey classes. The Transport interface is implemented by the Camel and Donkey classes. Write a test program that initialize an array of four Animal objects. If the object implements the Transport interface, the deliver() method is invoked. . | Unit – 6 |
| 79 | Write a abstract class named Person and its two subclasses named student and Employee. A person has a name, address, phone number and email address. A student has enrollment course. An Employee has an office, salary, and designation. Define constructors and methods for input and display for both classes. Define constructor and methods for input and display for both classes. Write a main program to give demonstration of all. | Unit – 4 |
| 80 | Write a complete multi-threaded program to meet following requirements:  - Read matrix [A] m x n  - Create m number of threads  - Each thread computes summation of elements of one row, i.e. ith row of the matrix is  processed by ith thread. Where 0 <= i< m.  - Print the results | Unit – 7 |
| 81 | Write an application that executes two threads. One thread displays "Good Morning" every 1000 milliseconds & another thread displays "Good Afternoon" every 3000 milliseconds. Create the threads by implementing the Runnable interface. | Unit – 7 |
| 82 | Write a complete multi-threaded program to meet following requirements:  o Two threads of same type are to be instantiated in the method main.  o Each thread acts as a producer as well as a consumer.  o A shared buffer can store only one integer information along with the source &  destination of the information at a time.  o The information produced is to be consumed by appropriate consumer.  o Both producers produce information for both consumers.  o Each thread produces 5 information | Unit – 7 |
| 83 | Write a multithreaded program to print all odd positive numbers in ascending order up to n, where n is a positive integer number given as a command line argument. Instantiate requited number of threads, where each thread except the last, examines next 50 numbers and the last thread examines remaining numbers up to n. | Unit – 7 |
| 84 | Write a complete multi threaded program to meet following requirements for producerconsumer threads:  - Three threads – one producer and two consumers to be instantiated in the method main.  - At a time, the producer produces one integer information along with consumer\_id to represent  id of a consumer that will consume produced information.  - Information and consumer\_id are stored in a shared buffer.  - The information produced is to be consumed by appropriate consumer only, as specified by  the producer.  - The producer thread produces total 6 information | Unit – 7 |
| 85 | Write a complete multi threaded program to meet following requirements for producerconsumer threads:  - Three threads – one producer and two consumers to be instantiated in the method main.  - At a time, the producer produces one integer information along with consumer\_id to represent  id of a consumer that will consume produced information.  - Information and consumer\_id are stored in a shared buffer.  - The information produced is to be consumed by appropriate consumer only, as specified by  the producer.  - The producer thread produces total 6 information | Unit – 7 |