AMITY UNIVERSITY

JHARKHAND

LAB MANUAL

Course Title: JAVA

Course Level: UG

Course Code: IT201

Program: B. TECH (CSE)

Semester: IV

Faculty:
MR. DIPRA MITRA
Assistant Professor (CS)

Lab Instructor
MR. RAHUL KUMAR LOHRA

General instructions to students

1. Students should be regular and come prepared for the lab practice.
2. In case a student misses a class, it is his/her responsibility to complete that missed experiment(s).
3. Students should bring the observation book, lab journal and lab manual.
4. Prescribed textbook and class notes can be kept ready for reference if required.
5. They should implement the given experiment individually.
6. Once the experiment(s) get executed, they should show the program and results to the instructors and copy the same in their observation book.
7. Questions for lab tests and exam need not necessarily be limited to the questions in the manual but could involve some variations and / or combinations of the questions.
8. All the students must maintain silence inside the lab.
9. All the students must carry their id card before entering the lab, and college uniform is strictly mandatory, otherwise students will not be permitted to sit inside the lab.
10. No food or beverage items are allowed inside the lab.
11. Keep your bags outside the lab.
12. Do not use cell phone inside the lab. (If anybody found using cell phone inside the lab, his/her mobile phone will be seized by the responsible authority.)
13. Shut down the system and arrange your chair before leaving the lab.
14. While using lab sign in the register with your name, enrollment number and branch to mark your attendance.
15. Do not plug any device without permission.
16. Do not use internet without permission.

Note: above mentioned instructions can be modified based on the context of the lab.

Credit Unit

L (Lecture)	T	P/S (Practical/Studio)	SW/FW	Total Credit Units
3	-	2	-	4

Lab/ Practical/ Studio Assessment

	Continuous Assessment/Internal Assessment				End Term Examination
Components (Drop down)	**Lab Record**	**Performance**	**Viva**	**Attendance**	
Weightage (%)	10	10	5	5	70

Course Objectives:

The objective is to impart programming skills used in this object-oriented language java. The students are expected to learn it enough so that they can develop the web solutions like creating applets etc.

Pre-requisites: Object Oriented Programming Concepts

Student Learning Outcomes:

- Able to recognize the benefits and features of Open Source Technology.
- Interpret, Contrast and compare open source products among themselves
- Understand and demonstrate Version Control System along with its commands

Pedagogy for Course Delivery:

The course would be covered under theory and laboratory. In addition to assigning project–based learning, early exposure to hands-on design to enhance the motivation among the students. It incorporates designing of problems, analysis of solutions submitted by the student's groups and how learning objectives were achieved. Continuous evaluation of the students would be covered under quiz, viva etc.

LIST OF EXPERIMENTS

Week 1: Control statements and arrays

Write java programs for the following. (Inputs are assumed in the program and not necessary through the keyboard)

1. To determine if a given number is prime or not.
2. To display all primes b/w given 2 limits.
3. To generate the first n Fibonacci numbers using recursion. The recursive method is in the same class as main.
4. Create and initialize an integer array. Sort the array and display it. (The array values can be taken from the key board using command line arguments)
5. Create and initialize 2 matrices. Find their sum. Display all 3 matrices.

Week 2: Stacks and Lists

1. Design a class, which represents a stack of integers. Provide constructors as well as methods namely push(), pop(), displayFromTop(), displayFromBot(), stackTop(), isEmpty() and isFull() which respectively pushes an integer onto the stack, pops an integer from the stack, displays the stack contents from top to bottom, displays the stack contents from bottom to top, displays the top element of the stack, determines whether the stack is empty or not and to determine whether the stack is full or not.
2. Design a stack class and provide all the previously listed methods so that it works with a stack of student records. Every student record should have a name and id field.
3. Design a class, which represents a list of integers. Provide a default constructor to this class. Also provide methods namely:
    a) isEmpty()
    b) Display()
    c) AddAtLast()
    d) DeleteAtHead()

    Write a main method, which uses these methods.
4. Implement the Stack operations using a linked list.

Week 3: Strings, classes and methods

Note: STORE SOLUTION TO EACH PART AS SEPARATE FILE

1. Design a class which represents an inpatient. Every inpatient record is made up of the following fields:
    i. name (String)
    ii. age (integer)
    iii. hospital number (integer)
    iv. Date of admission (class called Date form java.util) OR (user defined separate inner class)
    v. Room rent (double)

    Test the class by writing suitable main method.

a) Provide default, copy constructor and parameterized constructors to this class. Make suitable change in main to incorporate these changes and test the same.
b) The hospital number is incremented as obtained from a static variable.
c) Make suitable change and run the program.
d) Also provide methods to input a record, to display it, to determine if the date of admission is the todays date. Display suitable message by demonstrating the above class by making suitable change in main.
e) Create an array of inpatient records and store a minimum of 3 records in it. Assign the data to each of the fields through assignment statement (using constructors) and Display all of them. (No change with respect to class defined earlier. Only change main method)
f) Check if two patients have the same information except for hospital number. Also determine the number of records who are admitted today.

2. Create and initialize a String and determine the following. (Built in String methods may be used):
   a) the length of the string.
   b) the number of a's in the string.
   c) convert all lowercase to uppercase and vice versa.
   d) find if a sub string pattern "abc" is present; if so, display the first position of occurrence.

   (Be aware of multiple ways of string creation and accessing single characters)

3. Create and initialize an array of strings. Display them. Every string should contain a sentence. Capitalize every starting letter of the words in the string. Assume that one blank separates every 2 words in a string. Display the resultant array.

Week 4: Inheritance and packages

1. Design a class, which represents a patient. Fields of this class are name, age and hospital number. Provide methods to input and display all fields. Next design a class called Inpatient, which inherits from patient class. Provide fields to represent department name, admission date and room type. Provide methods to input and display these fields. Next design a class called Billing, which inherits from Inpatient class. Provide a field to represent the discharge date. Provide methods to input this field value as well as to display the total amount. The total amount is calculated based on room type and doctor charges as shown below:

RoomType	Consultancy Charges/day	Room Rent /day
Special	Rs. 1000.00	Rs. 200
SemiSpecial	Rs. 500.00	Rs. 100
General	Rs. 100.00	Rs. 50

Show the use of a Billing object in the main (Make use of super keyword).

2. Develop a set of methods, which work with an integer array. The methods to be implemented are:
   i. min (which finds the minimum element in the array)
   ii. max (which finds the maximum element in the array)
   iii. sort (method to sort the array)
   iv. reverse (method to reverse the array)
   v. scale (method to multiply the array)

   Place this in a package called p1. Let this package be present in a folder called "myPackages", which is a folder in your present working directory (eg: c\student\4rthcse01\mypackages\p1). Write a main method to use the methods of package p1. Try various other alternatives too.

3. Create an abstract class Figure with abstract method area and two integer dimensions. Extend this class to inherit three more classes Rectangle Triangle and Square which implements the area method. Show how the area can be computed dynamically during run time for Rectangle, Square and Triangle.
4. Design a list class. (The previously designed one can be used) Make it a super class. Design a class, which represents a double-ended list (List operations can be performed on both ends of the list). Let this class inherit from the list class. Provide all other additional methods to this class. Show the usage of this class in a main method

<u>Week 5: Interfaces, Exception Handling.</u>

1. Design an interface called Stack with 2 methods namely push() and pop() in it. Design a class called FixedStack, which implements a fixed length version of the stack. Also design a class called DynamicStack, which implements a growable version of the stack. Write a main method, which uses both these classes through an interface reference.

2. Design a class, which represents an employee. The data members are:
    a. name (string), age (int), grossSalary (double), takeHomeSalary(float), grade (char). Provide methods called input() and display() which reads all details of a record from the keyboard and displays them respectively. Handle IOException while reading from the keyboard. Provide a menu with the options: Input, Display and Exit to read users choice. (Make use of Wrapper classes)

3. Design a stack class. Provide your own stack exceptions namely push exception and pop exception, which throw exceptions when the stack is full and when the stack is empty respectively. Show the usage of these exceptions in handling a stack object in the main.

<u>Week 6 Threads and Input/output</u>

1. Write a java program to create and initialize a matrix of integers. Create n threads where n is equal to the number of rows in the matrix. Each of these threads should compute a distinct row sum. The main thread computes the complete sum by looking into the partial sums given by the threads.
2. Write a java program to illustrate the classic producer/ consumer problem, which takes care of synchronization.
3. Write information of n employees to a file. This information should be read from the keyboard. The data members of employee class are:
    a. name (string), age (int), GrossSalary (double), TakeHomeSalary(float), Grade (char). (Each record can be stored on a separate line with a blank separating every 2 fields). Transfer the records with grade 'A' to another file. Also display those records on the screen.
4. Write a program to display the listing of a given directory. Recursion can be used.
5. Assume that there exists a file with different file names stored in it. Every file name is placed on a separate line. Create two threads, which scan the first half, and the second half of the file respectively, to search the filenames which end with .cpp. Write those file names onto the screen.

6. Count the number of single characters, numbers (sequence of 1 or more digits), words and lines from a file and place the result on the screen.

7. Write a program to copy one file to other using

i. char stream classes ii. Byte stream classes

Week 7: Applets and Event Handling

1. Write an applet to display your biodata in the middle. Have suitable choice of background and foreground colors.
2. Develop an applet which scrolls a message "Java Programming", horizontally from left to right across the applet window. Set the background and foreground colors of the banner to cyan and red respectively. Also, in the status window display a message "this is the status window". Modify the banner applet such that the message to be scrolled is read from the user.
3. Write a Java program to demonstrate the mouse event handlers.
4. Write a Java program to demonstrate the keyboard event handlers.

Week 8: AWT and Swings

1. Write Java programs to demonstrate the usage of following AWT controls:
   i. Buttons, Check Boxes,
   ii. Checkbox Group,
   iii. Combo Boxes,
   iv. Radio Buttons,
   v. Lists, Text Field,
   vi. Text Area,
   vii. Tabbed Panes,
   viii. Scroll Panes.
2. Write a Java program to demonstrate the usage of Menu Bar and Menus. Use Dialog Boxes to display the Menu options selected.
3. Write Java program that allows the user to draw a rectangle by dragging the mouse on the application window. The upper-left coordinate should be the location where the user presses the mouse button, and the lower-right coordinate should be the location where the user releases the mouse button. Also display the area of the rectangle.
4. Modify the above program to draw a shape with the mouse. The shape Should be allowed to choose from an oval, an arc, a line, a rectangle with rounded corners and predefined polygon. Also display the mouse coordinates in the status bar.

Week 9: AWT and Swings

1. Write a Java program that displays a Circle of random size and calculates and displays the area, radius, diameter and circumference.
2. Enhance the above program by allowing the user to alter the radius with a scrollbar. The program should work for all radii in the range 100 to 200. As the radius changes, the diameter, area and circumference should be updated and displayed. The initial radius should be 150.
3. Write a Java program to simulate a static analog clock whose display is controlled by a user controlled static digital clock.

Week 10: AWT and Swings

1. Write a Java program to implement a simple calculator.
2. Write an applet to test username and password.
3. Write an applet that obtains two floating point numbers from the user and displays the sum, product, difference and quotient of these numbers.

Week 11: JDBC

Create and populate a simple student Database that contains name, password and other details using MS ACCESS. (Use windowed applications of java)

1. Write a Java program which runs the user supplied query on student database.
2. Write a menu driven Java program to do the following operations on the student database:
    a) Insert a new record.
    b) Display the specified records.
    c) Delete the specified records.
    d) Update the specified records.
    e) Exit.

****

Text Books & References:

1. JAVA, The Complete Reference by Patrick Naughton & Herbert Schild, TMH
2. Introduction to JAVA Programming a primar, Balaguruswamy.
3. “Introduction to JAVA Programming” Daniel/Young PHI
4. Jeff Frentzen and Sobotka, “Java Script”, Tata McGraw Hill.