List of Practical

Course: BCA

Subject Name and Code: Programming in Java Lab (BCAC 0819)

Year/Semester: II/III

Objective: The objective of this course is that the students will understand fundamentals of programming such as variables, conditional and iterative execution, methods, etc. It will also provide the foundation of good programming skills by discussing key issues to the design of object oriented programming.

Experiment 1: (If Else)

Compulsory:

Program 1: WAP to print a message "Hello JAVA".

Program 2: Write a program to display whether a number is even or odd.

Program 3 Write the following program using if else if ladder. Accept an hour from the user and output the following as indicated below. Include the last condition in the else section.

i) Hour greater than or equal to 0 and less than 12	"Good Morning"
ii) Hour greater than or equal to 12 and less than 18	"Good Afternoon"
iii) Hour greater than or equal to 18 and less than 24	"Good Evening"
iv) Any other input	"Time is out of range"

Additional Programs:

Program 4 : A student receives marks in three subjects. The program will calculate the total marks, average marks, and determine the grade based on the average:

Average >= 90: Grade A
Average >= 80 and < 90: Grade B
Average >= 70 and < 80: Grade C
Average >= 60 and < 70: Grade D
Average < 60: Grade

Experiment 2: Nested If Else

Compulsory:

Program 1: : WAP to find maximum of three numbers

Program 2: A movie theatre has the following ticket pricing rules:

Children under 12 years old pay $5.
Seniors 65 years and older pay $7.
Regular adults (12-64 years old) pay $10.
Members get a $2 discount on all ticket prices.
We'll prompt the user to input their age and whether they have a membership card. Based on this input, the program will determine and print the ticket price.

Program3: Create a program using switch case statement to identify the day of the week.

Additional Programs:

Program4: WAP to implement that performs basic arithmetic operations (addition, subtraction, multiplication, division) based on user input. use switch to select the arithmetic operation

Experiment 3: Simple and Nested Loop

Compulsory:

Program1 : Write a Program in Java to Calculate the Factorial of an Integer using a for loop

Program 2: You are tasked with developing a program that simulates a login system. The user is given three attempts to enter the correct password (In integer format eg. 4545). If the user enters the correct password within three attempts, they are granted access. If the user fails to enter the correct password in three attempts, they are locked out.

Additional Programs:

Program 3: WAP to display table of a number in a format n x i=m

Program 4: WAP to find sum of digits of a number

Program 5: WAP to implement type conversion in java.

Experiment 4: Patterns

Compulsory:

Program 1: Using For…..Loop display the following pattern :

```
*
**
***
****
*****
```

Program 2: WAP to print the pattern

```
11111
12111
11311
```

11141
11115

Additional Programs:

Program3: WAP to print the pattern
11111
12111
12311
12341
12345

Program 4: WAP to print the pattern
#####
$####
$$###
$$$##
$$$$#

Experiment 5: Array

Compulsory:

Program 1: WAP to read and print an array. Also find greatest and smallest element in array.

Program 2: You're developing a simple scoring system for a cricket match. The match involves a single over (6 balls) and you're required to record the runs scored on each ball. The system should also calculate the total runs scored in the over and determine if the over included any dot balls (a ball where no runs are scored).

Task:

Create a 1-dimensional array to store the runs scored on each of the 6 balls in the over.
Calculate the total runs scored in the over.
Count the number of dot balls (balls where zero runs were scored).
Determine the highest run scored on a single ball.

Runs scored in the over (ball by ball):

Ball 1: 1 run
Ball 2: 4 runs
Ball 3: 0 runs (dot ball)
Ball 4: 6 runs
Ball 5: 2 runs
Ball 6: 0 runs (dot ball)

Additional Programs:

- Program 4: Write a program to search an element in an array.
- Program 5: Write a program to sort an array using bubble sort.
- Program 6: WAP to read and print a matrix

- Program 7: WAP to find sum of two matrices

Experiment 6: Strings

Compulsory:

Program1: Write a Java program that will accept command-line arguments and display the same.

Program2: You are developing a simple application to manage the player lineup for a football (soccer) match. The coach needs to maintain a list of the starting 11 players for the match. After finalizing the lineup, the coach wants to:

Display the names of all the players in the starting lineup.
Check if a specific player is in the starting lineup.
Update the lineup by replacing a player who got injured during the warm-up with a substitute.
Task:

Create a String[] array to store the names of the 11 starting players.
Implement a method to display all players in the lineup.
Implement a method to check if a specific player is in the starting lineup.
Implement a method to replace an injured player with a substitute.

Starting Lineup:

Player 1: John
Player 2: David
Player 3: Mike
Player 4: Chris
Player 5: Alex
Player 6: Ryan
Player 7: James
Player 8: Sam
Player 9: Robert
Player 10: Daniel
Player 11: Steve
Substitute:

Substitute Player: Luke (to replace injured player James)

Additional Programs:

- Program 3: Write a program to accept number through Command line and display its factorial.
- Program 4: Write a program to extract a substring from a given string.

Experiment 7: Class

Compulsory:

Program1: WAP to create a **Rectangle** class. Define functions for

Reading length and breadth for rectangle
Calculate Area
Display length, breadth and area

Additional Programs:

Program2: This program defines a Student class with attributes like student ID, name, age, and grade. It also provides methods to set and get these attributes, along with a method to display the student's details.

Program3: You're tasked with developing a basic banking application. The application should allow users to:

Open a new bank account.
Deposit money into their account.
Withdraw money from their account.
Check their account balance.

Experiment 8: Constructor

Compulsory:

Program 1: In this program, we'll design a Book class that demonstrates the use of constructors to initialize objects. The Book class will have attributes like title, author, and price. We will use different types of constructors:

Default Constructor: Initializes attributes with default values.
Parameterized Constructor: Initializes attributes with specific values.

Additional Programs:

Program2: In this program, we'll create a Car class that makes use of constructors. The Car class will have attributes like brand, model, and year of manufacture. We'll include:

A default constructor to initialize attributes with default values.
A parameterized constructor to initialize attributes with specific values.

Program3 : In this program, we'll design a Person class that makes use of both default and parameterized constructors. The Person class will have attributes for the person's name, age,

and address. We'll provide methods to initialize these attributes, display the person's details, and modify the attributes.

Experiment 9: Static and This

Compulsory:

Program 1: Write a Java program that demonstrates the use of a static data member to count the number of objects created for a class.

Solution

Program 2: WAP Java program that demonstrates the use of ‘this’ pointer in a class. We'll create a Person class where we use ‘this’ pointer to distinguish between instance variables and constructor parameters.

Solution

Additional Programs:

Program 1: WAP that use ‘this’ keyword to return the current class instance
https://www.geeksforgeeks.org/this-reference-in-java/

Solution

Program 2: WAP that use ‘this’ keyword to invoke the current class method

Solution

Experiment 10: Inheritance

Compulsory:

Program 1: WAP to implement single inheritance in Java

Solution

Program 2: Animal and Dog Classes

Solution

Problem Statement: You need to manage different types of animals and their characteristics. The base class will represent a general animal, while the derived class will represent a specific type of animal, in this case, a dog. The program will demonstrate how the dog inherits attributes and behaviors from the animal.

Solution

Additional Programs:

Program 1:
Scenario: Banking System
Problem Statement:
You are developing a simple banking system to manage different types of accounts. The base class Account will represent a general bank account with basic attributes, while the SavingsAccount class will represent a specific type of account that has additional features such as interest rate and deposit interest.

Solution

Program 2: WAP to make use of super to call base class constructor

Solution

Experiment 11: Overloading and Overriding

Compulsory:

Program 1: WAP to implement method overloading in Java

Solution

Program 2: WAP to implement method overriding in Java

Solution

Program 3:

Scenario: Banking Application
Problem Statement:
You need to implement a banking system that allows customers to make deposits into their accounts using different methods. The program will demonstrate method overloading by providing multiple deposit methods that handle different parameter types.

Solution

Additional Programs:

Program 1: WAP to implement run time polymorphism in java.

Solution

Experiment 12:Abstract and Final

Compulsory:

Program 1: WAP to make use of abstract class in Java

Solution

Program 2: WAP to make use of final variables in Java

Solution

Program 3 :
Scenario: Shape Area and Perimeter Calculation
Problem Statement:
You need to implement a system that can calculate the area and perimeter of different shapes. The abstract class Shape will define the method signatures, and the derived classes Circle and Rectangle will provide their specific implementations.

Solution

Additional Programs:

Program 1:
Scenario: Banking System
Problem Statement:
You need to implement a banking system that supports different types of accounts. The abstract class Account will define the methods for basic account operations. The derived classes will provide specific implementations for handling deposits and withdrawals.

Solution

Experiment 13: Package and Interface

Compulsory:

Program 1: WAP to make use of interfaces in Java

Solution

Program 2:
Scenario: Transportation System
Problem Statement:

You need to implement a transportation system that allows different types of vehicles to start and stop. The interface Transportable will define methods for these actions. The classes Car and Bicycle will provide specific implementations for starting and stopping.

Solution

Additional Programs:

Program 1: WAP to implement multiple inheritance using interfaces in Java

Solution

Program 2 : WAP to make use of packages in Java

Solution

Experiment 14: Exception handling and Multithreading

Compulsory:

Program 1: WAP to implement Exception handling in Java.

Solution

Program 2: WAP to implement Multithreading in Java

Solution

Program 3 : Scenario: Student Name Management

Problem Statement:

You need to manage a list of student names in an array. The program will allow users to input an index to retrieve the corresponding student's name. If the index is out of bounds, the program will handle the exception gracefully.

Solution

Additional Programs: Applets

Program 1: WAP to create and display an Applet in Java

Solution

Program 2: WAP to show Indian Flag on an Applet.

Solution