Institute of Engineering & Technology
An Autonomous Institution
Accredited by NAAC with '' A'' Grade, Accredited by NBA (ECE, CSE.EEE & MECH)
Approved by A.I.C.T.E. & Permanently Affiliated to J. N. T. U. Gurajada, VIZIANAGARAM
Via 5th APSP Battalion, Jonnada (V), Denkada (M), NH-3, Vizianagaram Dist - 535005, A.P. Website : www.lendi.org
Ph : 08922-241111, 241666, Cell No : 9490344747, 9490304747,e-mail : lendi_2008@yahoo.com

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

STUDENT LABORATORY MANUAL

OF

II B.TECH II SEMESTER (R20)

OBJECT ORIENTED PROGRAMMING THROUGH JAVA LAB

lendi Institute of Engineering & Technology
An Autonomous Institution
Accredited by NAAC with '' A'' Grade, Accredited by NBA (ECE, CSE.EEE & MECH)
Approved by A.I.C.T.E. & Permanently Affiliated to J. N. T. U. Gurajada, VIZIANAGARAM
Via 5th APSP Battalion, Jonnada (V), Denkada (M), NH-3, Vizianagaram Dist - 535005, A.P. Website : www.lendi.org
Ph : 08922-241111, 241666, Cell No : 9490344747, 9490304747,e-mail : lendi_2008@yahoo.com

INSTITUTE

VISION

Producing globally competent and quality technocrats with human values for the holistic needs of industry and society

MISSION

- Creating an outstanding infrastructure and platform for enhancement of skills, knowledge and behaviour of students towards employment and higher studies
- Providing a healthy environment for research, development and entrepreneurship, to meet the expectations of industry and society.
- Transforming the graduates to contribute to the socio-economic development and welfare of the society through value based education.

lendi Institute of Engineering & Technology
An Autonomous Institution
Accredited by NAAC with '' A'' Grade, Accredited by NBA (ECE, CSE.EEE & MECH)
Approved by A.I.C.T.E. & Permanently Affiliated to J. N. T. U. Gurajada, VIZIANAGARAM
Via 5th APSP Battalion, Jonnada (V), Denkada (M), NH-3, Vizianagaram Dist - 535005, A.P. Website : www.lendi.org
Ph : 08922-241111, 241666, Cell No : 9490344747, 9490304747,e-mail : lendi_2008@yahoo.com

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

VISION

To excel in the computing arena and to produce globally competent computer science and Information Technology graduates with Ethical and Human values to serve society.

MISSION

- To impart strong theoretical and practical background in computer science and information technology discipline with an emphasis on software development.
- To provide an open environment to the students and faculty that promotes professional growth
- To inculcate the skills necessary to continue their education and research for contribution to society.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates of Computer Science and Information Technology will acquire strong knowledge to analyze, design, and develop computing products and solutions for real-life problems utilizing the latest tools, techniques, and technologies.

PEO2: Graduates of Computer Science and Information Technology shall have interdisciplinary approach, professional attitude and ethics, communication, teamwork skills and leadership capabilities to solve social issues through their Employment, Higher Studies and Research.

PEO3: Graduates will engage in life-long learning and professional development to adapt to dynamically computing environment.

PROGRAM OUTCOMES (POs)

PO1: Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design & Development: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations

PO4: Investigations: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern Tools: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: Engineer & Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment & Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice

PO9: Individual & Team Work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings

PO10: Communication Skills: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions

PO11: Project mgt. & Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments

PO12: Life Long Learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: Ability to solve contemporary issues utilizing skills

PSO2: To acquire knowledge of the latest tools and technologies to provide technical solutions

PSO3: To qualify in national and international competitive examinations for successful higher studies and Employment

Subject Code	Subject Name	L	T	P	C
R20CSE-PC2204	Object Oriented programming through Java LAB Common to CSE,CSSE & CSIT	0	0	3	1.5

Course Objectives:

- To understand the structure and environment of Java.
- To implement the relationship between objects.
- To apply data hiding strategy in objects.
- To implement text processing and error handling.
- To organize data using different data structures.
- To create multi threaded graphical user interface applications.

Course Outcomes:

1. Create classes and objects for real world entities.
2. Implement polymorphic and abstract behaviour in objects.
3. Implement the parent-child relationships between objects with access protection.
4. Create exceptions for handling runtime errors during text processing.
5. Implement generic data structures for iterating distinct objects.

Exercise-1:

a. Write a Java program to create Class as Registration with properties as Full Name(String) , Gender(char), Age(int), Height(double), Phone Number(long), and isMarried(Boolean) and print their values.
b. Write a Java program to implement Type Casting and Conversion.
c. Write a Java program to implement Wrapper Classes.

Exercise-2:

a. Write a Java program to take input as Regd.No and print the branch depending upon the department code in that Regd.No using else-if and switch statements. (EgRegNo: 19KD1A0505, 8^{th} character is department Code, 5-CSE, 4-ECE, 3-MECH, 2-EEE etc.
b. Write a Java program to read input integers from Command Line Arguments and print first and second largest numbers.
c. Write a Java program to take input as Integer array and print even indexed even numbers and odd indexed odd numbers.

Exercise-3:

a. Write a Java program to take input as Decimal number and convert into Roman Number.
b. Write a Java program to check whether given number is Extension number. The extension number is the number which is present in the last digit(s) of its square.(Eg. N=25, 625 is Extension number since it contains 25).
c. Write a Java program to take input as Amount in rupees and print their denominations and total number notes.

Exercise-4:

a. Create a Class named Student with properties as Student Id, Student Name, gender, department, Age, Aggregate and methods as insertStudent() for inserting student details and displayStudent() for printing student details.
b. Create a class Student with same properties as above and create a constructor to insert student details and return the data using toString() method.

Exercise-5:

a. Design a Class named Transaction to transfer amount (double) in different ways using Account Number(int) , Phone Number(Long) and qr Code (String) as parameter into a method transferAmount() to achieve Method or Constructor OverLoading.
b. Design a super Class Account and sub Classes as LoanAccount, SavingsAccount and CurrentAccount and implement relationship between parent and child classes. (Implement Packages for the above classes)

Exercise-6:

a. Write a Java program to implement this and super keywords.
b. Write a Java program to implement Static property, method, block and package.
c. Write a Java program to implement final property, method and class.

Exercise-7:

a. Write a Java program to implement Data Abstraction using Abstract class and Interface.
b. Write a Java program to implement Multiple Inheritance through Interfaces.

Exercise-8:

a. Write a Java program to take input as String Sentence S and print largest and shortest word in S.
b. Write a Java program to take input as String S and remove the consecutive repeated characters from S. (Eg. S = Raaaamaaa then, Rama)
c. Write a Java program to take input as String Sentence S and print sentence again with all the words with the first letter as capital letter using StringBuffer.

Exercise-9:

a. Write a Java program to implement Map interface.
b. Write a Java program to implement Set Interface.
c. Write a Java program to implement List Interface.
d. Write a Java program to implement ComparatorInterface.

Exercise-10:

a. Write a Java program to read data from Employee file and print Highest salary employee information. (Employee File Contains: ID, name, Dept, Salary).
b. Write a Java program to implements Serializable Interface to read and write Objects to/from the file.

Exercise-11:

a. Write a Java program to implement try, catch, finally blocks.
b. Write a Java program to create user defined Exception and implement throw and throws handlers.

Exercise-12:

a. Write a Java program to create Thread using Thread Class and Runnable Interface.
b. Write a Java program to implement multi-threading and synchronization.
c. Write a Java program to implement Inter Thread Communication.

Exercise-13:

a. Create an Applet that changes the Font and background Color depending upon the user selection from the input.
b. Write a Java program to implement Event Delegation model through AWT Components.
c. Write a Java program to implement Layout Manager with AWT components.

COURSE OUTCOMES VS POs MAPPING (DETAILED: HIGH: 3, MEDIUM: 2, LOW: 1)

Course	SNO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C105	C105.1	3	3	2	2	2				2				2		2
	C105.2	3	3	2	2	2			1	2				2		2
	C105.3	3	3	2	2	2			1	2				2		2
	C105.4	3	3	2	2	2				2				2		2
	C105.5	3	3	2	2	2				2				2		2
	C105.*	3	3	2	3	3			1	2				2		2

**For Entire Course, PO & PSO Mapping*

S.NO	PO/PSO MAPPED	LEVEL OF MAPPING	JUSTIFICATION
C105.1	PO1	3	Students apply the core knowledge to solve the complex computer science engineering problems by using JAVA OOPs principles
	PO2	3	Able to analyze the real world objects by using different approaches like bottom-up strategy
	PO3	2	Student can able to develop different applications on Decision making statements an looping control statements for real time entities
	PO4	2	Able to analyze complex problem and write source code like building programs on multidimensional arrays
	PO5	2	Will use high end tools like Net Beans IDE, Intellij code to write complex code.
	PO9	2	Student can able to do team work and deliver mini projects on oops concepts
	PSO1	2	Students will Able to grasp advanced programming techniques to solve contemporary issues.
	PSO3	2	Logical thinking and code building Will helpful in campus interviews
C105.2	PO1	3	Students apply the core knowledge to solve the complex computer science engineering problems using various control structures
	PO2	3	Able to analyze the real world objects by using different types inheritance
	PO3	2	Student able to develop different objects for real time entities

			using proper class and object hierarchy and constructors
	PO4	2	Able to analyze complex problem and write source code by using various types of polymorphism
	PO5	2	Will use high end tools like BlueJ ,Net Beans IDE to write complex code.
	PO8	1	Students will apply ethical values while implementation of applications using java
	PO9	2	Student can able to do team work and deliver mini projects on inheritance concepts
	PSO1	2	Students will Able to grasp advanced programming techniques to solve contemporary issues like standalone J2SE applications
	PSO3	2	OOPS concepts and interfaces implementations concepts Will helpful in campus interviews to solve dynamic problems
C105.3	PO1	3	Students apply the core knowledge to solve the complex computer science engineering problems using String classes
	PO2	3	Able to analyze the real world objects using string builder and string buffer
	PO3	2	Student able to develop different objects for real time entities using collections
	PO4	2	Able to analyze complex problem and write source code by dealing with iterator and List iterator
	PO5	2	Will use high end tools like Blue J, Eclipse IDE to write complex code.
	PO8	1	Students will apply ethical values while implementation of applications using java
	PO9	2	Student can able to do team work and deliver mini projects on string handling functions and string builder and buffer
	PSO1	2	Students will able to grasp advanced programming techniques to solve contemporary issues using various string methods
	PSO3	2	String handling Functions Will helpful in campus interviews to reduce the LOC
C105.4	PO1	3	Students apply the core knowledge to solve the complex computer science engineering problems using I/O Streams
	PO2	3	Able to analyze the real world objects using character and byte streams
	PO3	2	Student able to develop different objects for real time entities using serialization and de-serialization
	PO4	2	Able to analyze complex problem and write source code using read and write streams
	PO5	2	Will use high end tools like Net Beans IDE, Blue J to write complex code.
	PO9	2	Student can able to do team work and deliver mini projects on

			file operation systems
	PSO1	2	Students will able to grasp advanced programming techniques to solve contemporary issues like error handling
	PSO3	2	I/O handling functions will helpful in campus interviews
C105.5	PO1	3	Students apply the core knowledge to solve the complex computer science engineering problems using multi threading
	PO2	3	Able to analyze the real world objects using thread synchronization
	PO3	2	Student able to develop different objects for real time entities like creation for GUI components
	PO4	2	Able to analyze complex problem and write source code to build real time applications
	PO5	2	Will use high end tools like BlueJ,Net Bans IDE to write complex code.
	PO9	2	Student can able to do team work and deliver mini projects on threads and synchronization
	PSO1	2	Students will Able to grasp advanced programming techniques to solve contemporary issues to deal with various thread related applications and applets and
	PSO3	2	Layout and building menus will helpful in campus interviews

Week-1

(a) AIM: Write a Java program to create Class as Registration with properties as Full Name (String),Gender (char), Age (int), Height (double), Phone Number (long), and isMarried (Boolean) and print their values.

Description

- This program defines a class Registration that holds personal details like name, age, gender, etc.
- It demonstrates the use of different data types in Java.
- Object-oriented concepts such as class, object, and instance variables are applied.
- The values are assigned to instance variables through object creation.
- The output displays the personal information of a person.

Algorithm

Step-1: Define a class Registration with variables for name, gender, age, height, phone number, and marital status.

Step-2: In the main method, create an object of Registration.

Step-3: Assign values to the object's variables.

Step-4: Use System.out.println() to display the data.

Step-5: End the program.

(b) AIM : Write a Java program to implement Type Casting and Conversion.

Description

- This program demonstrates both implicit (widening) and explicit (narrowing) type casting.
- It converts an int to a double (implicit).
- It converts a double to an int (explicit).
- This helps in understanding how data types are managed in Java.
- Essential for mathematical and memory-sensitive applications.

Algorithm

Step-1: Declare and initialize an integer variable.

Step-2: Convert it to double (implicit casting).

Step-3: Declare and initialize a double variable.

Step-4: Convert it to int (explicit casting).

Step-5: Print both results.

c. AIM:Write a Java program to implement Wrapper Classes.

Description

- This program introduces wrapper classes in Java like Integer, Double, and Character.
- It performs boxing (primitive to object) and unboxing (object to primitive).
- Demonstrates auto-boxing and auto-unboxing.
- These features are helpful when working with collections.
- It simplifies object handling of primitive types.

Algorithm

Step-1: Declare a primitive int and convert it to Integer (boxing).

Step-2: Declare a Double object and convert it to primitive (unboxing).

Step-3: Use auto-boxing and auto-unboxing for a character.

Step-4: Print all the converted values.

Step-5: End the program.

Viva Questions

1) What are primitive data types in Java, and give examples?

2) Explain the difference between implicit and explicit type casting.

3) What is the purpose of wrapper classes in Java?

4) What is auto-boxing and unboxing in Java?

5) Why do we need type conversion in programming?

Week-2

(a) AIM : Write a Java program to take input as Regd.No and print the branch depending upon the department code in that Regd.No using else-if and switch statements. (EgRegNo: 19KD1A0505, 8th character is department Code, 5-CSE, 4-ECE, 3-MECH, 2-EEE etc.

Description

- This program extracts the department branch from a given registration number.
- It uses both else-if and switch-case structures for logical comparison.
- The 8th character in the registration number is used to determine the branch.
- The program identifies branches like CSE, ECE, MECH, and EEE.
- It also includes validation for incorrect or short registration numbers.

Algorithm

Step-1: Take input for registration number from the user.

Step-2: Check if the registration number has at least 8 characters.

Step-3: Extract the 8th character and store it in a variable.

Step-4: Use else-if and switch-case to determine the branch.

Step-5: Print the identified branch and end the program.

(b) AIM : Write a Java program to read input integers from Command Line Arguments and print first and second largest numbers.

Description

This program finds the first and second largest numbers among the command-line arguments.

It checks if enough arguments are passed.

It parses the string arguments to integers.

Uses logic to track the two largest unique numbers.

Useful for basic array handling and number comparison.

Algorithm

Step-1: Check if at least two arguments are provided.

Step-2: Initialize first and second largest variables to minimum value.

Step-3: Parse each argument and compare to update first and second largest.

Step-4: Display the two largest numbers after comparison.

Step-5: End the program.

(c) AIM : Write a Java program to take input as Integer array and print even indexed even numbers and odd indexed odd numbers.

Description

- This program filters even numbers at even indices and odd numbers at odd indices.
- It takes an array input from the user.
- Iterates through the array to check conditions based on index parity.
- Displays matching numbers with their indices.
- Useful for practice with arrays and conditional logic.

Algorithm

Step-1: Take the array size input from user.

Step-2: Read array elements using a loop.

Step-3: Traverse the array to check even numbers at even indices.

Step-4: Then check odd numbers at odd indices.

Step-5: Display results and exit.

Viva Questions

1) How does the switch-case structure work in Java?

2)What is the use of command line arguments in Java?

3)How do you handle array indexing and boundaries?

4) What happens if two largest numbers are equal in the second program?

5) What are valid cases to use else-if vs switch statements?

Week-3

(a) AIM : Write a Java program to take input as Decimal number and convert into Roman Number.

Description

- Converts a decimal number (1–3999) into its equivalent Roman numeral format.
- Uses arrays for Roman values and their corresponding symbols.
- Iteratively subtracts values from the number while appending symbols.
- Utilizes StringBuilder for efficient string manipulation.
- Applies validation for number range to avoid invalid inputs.

Algorithm

Step-1: Read input number using Scanner.

Step-2: Check if number is within valid range (1 to 3999).

Step-3: Create arrays for decimal and Roman equivalents.

Step-4: Subtract and append Roman symbols until number becomes 0.

Step-5: Display the resulting Roman numeral.

(b) AIM : Write a Java program to check whether given number is Extension number. The extension number is the number which is present in the last digit(s) of its square. (Eg.N=25, 625 is Extension number since it contains 25).

Description

- Identifies whether a given number is an extension number.
- An extension number is a number whose square ends with the number itself.
- Uses string manipulation to compare number and its square.
- Demonstrates basic arithmetic and string operations.
- Provides meaningful result based on comparison**.**

Algorithm

Step-1: Read the number using Scanner.

Step-2: Compute the square of the number.

Step-3: Convert both number and square to string.

Step-4: Use endsWith() to check match.

Step-5: Print whether it is an extension number.

c. AIM : Write a Java program to take input as Amount in rupees and print their denominations and total number notes.

Description

- Calculates the number of currency notes needed for a given amount.
- Uses an array of note denominations from highest to lowest.
- Uses integer division and modulus to compute counts.
- Displays detailed note breakdown and total notes used.
- Helps understand loops, conditionals, and modular arithmetic.

Algorithm

Step-1: Read the total amount from user input.
Step-2: Create an array of denominations.
Step-3: Use a loop to divide and find count of each note.
Step-4: Reduce the amount using modulo after each iteration.
Step-5: Display denomination breakdown and total notes.

Viva Questions

1)How does the Roman numeral conversion logic work in Java?.

2) What is an Extension Number? Give an example.

3) How are string methods like ends With() useful in Java?

4) How do you handle division and remainders for denominations?

5) Why is StringBuilder preferred over string concatenation in loops?

Week-4

(a) AIM : Create a Class named Student with properties as Student Id, Student Name, gender, department, Age, Aggregate and methods as insertStudent() for inserting student details and displayStudent() for printing student details.This program demonstrates the use of class methods to handle student data.

Description

- insertStudent() method takes input from the user and assigns it to fields.
- displayStudent() method prints all the details of the student object.
- Scanner class is used to read input values from the console.
- It reinforces the basics of class, object creation, and method handling in Java.

Algorithm

Step-1: Create a Student class with fields for ID, name, gender, department, age, and aggregate.

Step-2: Define a method insertStudent() to take inputs from the user.

Step-3: Define a method displayStudent() to print the entered data.

Step-4: Create an object of Student class and call both methods from main().

Step-5: Display formatted output and end program.

(b) AIM : Create a class Student with same properties as above and create a constructor to insert student details and return the data using toString() method.

Description

- This program uses a constructor to initialize object attributes while creating the object.
- The toString() method is overridden to provide a formatted output when printing the object.
- Simplifies object output by avoiding multiple print statements.
- Demonstrates constructor, this keyword, and method overriding.
- Output is displayed by simply printing the object reference.

Algorithm

Step-1: Create Student class with variables for ID, name, gender, department, age, and aggregate.

Step-2: Write a parameterized constructor to initialize the data.

Step-3: Override the toString() method to return formatted string of details.

Step-4: Create object of Student class by passing values.

Step-5: Print the object directly to view student details.

Viva Questions

1)What is the purpose of insertStudent() and displayStudent() methods?

2) What is a constructor and how is it different from a method?

3)Why do we override the toString() method?

4) What is the use of this keyword in Java?

5) How does method overriding help in output formatting?

Week-5

a)AIM : Design a Class named Transaction to transfer amount (double) in different ways using Account Number(int) , Phone Number(Long) and qr Code (String) as parameter into a method transferAmount() to achieve Method or Constructor Over Loading.

Description

- This program demonstrates method overloading by defining multiple transferAmount() methods.
- Each method accepts different parameter types: account number, phone number, or QR code.
- It allows fund transfer to different destinations using a single method name.
- Java decides which method to call based on argument types.
- Method overloading improves code readability and modularity

Algorithm

Step-1: Define a class Transaction with three overloaded methods transferAmount().

Step-2: Each method should take different argument types: int, long, or String.

Step-3: In main(), create an object of Transaction.

Step-4: Call all three methods with appropriate parameters.

Step-5: Print the transaction messages accordingly

(b) AIM : Design a super Class Account and sub Classes as LoanAccount, SavingsAccount and CurrentAccount and implement relationship between parent and child classes.(Implement Packages for the above classes)

Description

- This program uses inheritance with a superclass Account and three subclasses.
- Each subclass (LoanAccount, SavingsAccount, CurrentAccount) extends Account and adds unique functionality.
- Demonstrates reusability and modularity using packages.
- The main program imports bank.* to access classes and execute operations.
- It covers real-world banking operations like balance, loan, interest, and overdraft.

Algorithm

Step-1: Create a package bank and define Account class with fields and displayDetails() method.

Step-2: Create three subclasses: LoanAccount, SavingsAccount, and CurrentAccount.

Step-3: Add specific methods in each subclass to show loan, interest, or overdraft.

Step-4: Create a Main class and import bank.*.

Step-5: Instantiate all subclasses and display their information.

Viva Questions

1) What is method overloading in Java?.

2) How does Java resolve which overloaded method to call?

3) What is inheritance in Java?.

4) Why do we use packages in Java?

5) Can we override methods in subclasses?

Week-6

(a) AIM : Write a Java program to implement this and super keywords.

Description

- This program demonstrates the use of this and super keywords in Java.
- The this keyword refers to the current class variable.
- The super keyword is used to access variables from the parent class.
- A subclass Student extends a superclass Person.
- It clearly shows how parent and child variables are differentiated.

Algorithm

Step-1: Create a class Person with a variable name and method show().

Step-2: Create a class Student that extends Person.

Step-3: Use this.name to refer to the current class variable.

Step-4: Use super.name to refer to the parent class variable.

Step-5: In main(), create a Student object and call display().

(b) AIM : Write a Java program to implement Static property, method, block and package.

Description

- This program demonstrates the usage of static property, method, and block.
- A static variable college is shared by all objects.
- A static block runs once when the class is loaded.
- The static method changeCollege() changes the static variable for all objects.
- It uses package statics for modularity.

Algorithm

Step-1: Create a class StaticExample in package statics.

Step-2: Declare static variables and methods.

Step-3: Create objects with different data.

Step-4: Display data before and after calling static method.

Step-5: Observe that static block runs only once.

(c) AIM : Write a Java program to implement final property, method and class.

Description

- This program explains how to use the final keyword in Java.
- final class cannot be extended.
- final variables cannot be reassigned after initialization.
- final methods cannot be overridden.
- It ensures security and prevents accidental changes.

Algorithm

Step-1: Create a final class Animal with a final variable type.

Step-2: Define a final method sound() inside it.

Step-3: Create an object and access both the property and method.

Step-4: Attempting to override or change will lead to a compile-time error.

Step-5: Observe the secure behavior of final elements.

Viva Questions

1) What is the use of this **keyword in Java?**

2) What does super **keyword do in Java?**

3) What is a static block in Java?

4) Can a final method be overridden? Why or why not?

5) Why do we use the final keyword with classes or variables?

Week-7

(a). AIM : Write a Java program to implement Data Abstraction using Abstract class and Interface.

Description

- This program demonstrates data abstraction in Java using both abstract classes and interfaces.
- Abstract classes can include both abstract and concrete methods.
- Interfaces define methods without implementation, ensuring full abstraction.
- It shows how different types of objects can be handled polymorphically.
- This is key in real-world applications for hiding implementation details.

Algorithm (Using Abstract Class)

Step-1: Create an abstract class Vehicle with one abstract and one concrete method.

Step-2: Create a class Bike that extends Vehicle and implements the start() method.

Step-3: In main(), create an object using the Vehicle reference pointing to Bike.

Step-4: Call start() and fuelType() methods.

Step-5: Observe output that shows abstraction in action.

Algorithm (Using Interface)

Step-1: Define an interface Animal with an abstract method sound().

Step-2: Create a class Dog that implements Animal.

Step-3: Override the sound() method to define behavior.

Step-4: In main(), create an object of Dog and call sound().

Step-5: Observe the output from the overridden method.

.

(b) AIM : Write a Java program to implement Multiple Inheritance through Interfaces.

Description

- Java does not support multiple inheritance through classes but supports it through interfaces.
- This program uses two interfaces Printable and Showable.
- A class Document implements both interfaces and provides their method definitions.
- Demonstrates how Java achieves multiple inheritance safely using interfaces.
- Useful in real-world applications requiring combined behaviors

Algorithm

Step-1: Create two interfaces: Printable and Showable.

Step-2: Define one abstract method in each interface.

Step-3: Create a class Document that implements both interfaces.

Step-4: Implement the print() and show() methods.

Step-5: Create an object in main() and invoke both methods.

Viva Questions

1) What is data abstraction in Java?.

2) How is an abstract class different from an interface?

3) Can you create an object of an abstract class?

4) How does Java support multiple inheritance?.

5)Why is multiple inheritance through classes not allowed in Java?

Week-8

(a). AIM : Write a Java program to take input as String Sentence S and print largest and shortest word in S

Description

- This program takes a sentence as input from the user.
- It splits the sentence into words using whitespace as delimiter.
- The program identifies the word with the maximum and minimum length.
- Uses a simple loop and comparison to find the result.
- Demonstrates basic string manipulation in Java.

Algorithm

Step-1: Read the input sentence using Scanner.

Step-2: Split the sentence into words using split() method.

Step-3: Initialize shortest and largest to the first word.

Step-4: Loop through all words and compare lengths.

Step-5: Display the shortest and largest words.

(b) AIM : Write a Java program to take input as String S and remove the consecutive repeated characters from S. (Eg. S = Raaaamaaa then, Rama)

Description

- This program removes adjacent repeating characters from a string.
- It uses a StringBuilder to construct the result efficiently.
- It compares each character with the previous one.
- If different, it appends to the result; if same, skips it.
- This helps in data cleaning and processing.

Algorithm

Step-1: Read a string from the user.

Step-2: Initialize a StringBuilder and previous character variable.

Step-3: Loop through each character of the string.

Step-4: Append to result only if current character is different from previous.

Step-5: Display the resulting string.

(c) AIM : Write a Java program to take input as String Sentence S and print sentence again with all the words with the first letter as capital letter using StringBuffer.

Description

- This program capitalizes the first letter of every word in a sentence.
- It uses StringBuffer to build the formatted output.
- Splits the sentence into words and modifies each one.
- Ensures all remaining letters are in lowercase.
- Enhances formatting of user-input sentences.

Algorithm

Step-1: Read a sentence using Scanner.

Step-2: Split the sentence into words using split().

Step-3: Convert the first character of each word to uppercase.

Step-4: Convert the rest of the word to lowercase.

Step-5: Reconstruct the sentence and display it

Viva Questions

1) How do you split a sentence into words in Java?

2)What is the use of StringBuilder or StringBuffer in Java?

3) How does the program detect duplicate characters?

4)How do you capitalize the first letter of a word in Java?.

Week-9

(a) AIM : Write a Java program to implement Map interface.

Description

- The program demonstrates the use of the Map interface through HashMap.
- Each key-value pair stores a student's registration number and name.
- It shows how to add entries and iterate using entrySet().
- Maps do not allow duplicate keys but allow duplicate values.
- Used widely in applications like dictionaries or student records.

Algorithm

Step-1: Import java.util.* package.

Step-2: Create a HashMap of Integer and String.

Step-3: Add key-value pairs using put().

Step-4: Use enhanced for loop with entrySet() to iterate and print entries.

Step-5: Display the output.

(b) AIM : Write a Java program to implement Set Interface.

Description

- This program demonstrates the use of Set using HashSet.
- Set does not allow duplicate elements.
- Order of elements is not guaranteed due to hash-based storage.
- Useful when storing unique items like city names.
- Simple operations like add and iterate are used.

Algorithm

Step-1: Import java.util.*.

Step-2: Create a HashSet<String> for cities.

Step-3: Add multiple elements (including duplicates).

Step-4: Iterate and print the values using enhanced for loop.

Step-5: Observe that duplicate entries are removed.

(c) AIM : Write a Java program to implement List Interface

Description

- This program shows how List allows duplicates and maintains insertion order.
- ArrayList is used to store a list of fruits.
- Elements can be repeated and are printed in the order added.
- Demonstrates simple usage of add() and for-each loop.
- Useful for handling sequential collections.

Algorithm

Step-1: Import java.util.*.

Step-2: Create an ArrayList of type String.

Step-3: Add fruit names to the list.

Step-4: Use enhanced for loop to print each element.

Step-5: Observe that duplicates are allowed.

(d) AIM : Write a Java program to implement Comparator Interface

Description

- This program demonstrates how to sort objects using Comparator.
- A custom comparator is created to sort Student objects by name.
- It overrides the compare() method in NameComparator class.
- Collections.sort() is used with the comparator.
- Useful for customized sorting based on user-defined criteria.

Algorithm

Step-1: Create a class Student with id and name.

Step-2: Create a class NameComparator that implements Comparator.

Step-3: Override compare() to sort students by name.

Step-4: Create a List<Student> and add student objects.

Step-5: Sort using Collections.sort() and print the result.

Viva Questions

1) What is the difference between List, Set, and Map?

2) What is the use of HashSet in Java?.

3) How is a HashMap different from a TreeMap?

4) What is the role of Comparator in Java?

5) Can a Map contain duplicate values?

Week-10

(a) AIM : Write a Java program to read data from Employee file and print Highest salary employee information. (Employee File Contains: ID, name, Dept, Salary).

description

- This program reads employee details from a file named employees.txt.
- Each line contains data like employee ID, name, department, and salary.
- It processes each line and identifies the employee with the highest salary.
- Uses BufferedReader and string parsing for input processing.
- Demonstrates file handling and object comparison in Java

Algorithm

Step-1: Create an Employee class with fields and toString() method.

Step-2: Open the file using BufferedReader.

Step-3: Read each line and split data by commas.

Step-4: Convert string data into appropriate types and create Employee objects.

Step-5: Compare salaries and track the highest-paid employee.

Step-6: Display the result.

(b) AIM : Write a Java program to implements Serializable Interface to read and write Objects to/from the file.

Description

- This program demonstrates Java's Serializable interface.
- An Employee object is saved to a file using serialization.
- The same object is later restored using deserialization.
- ObjectOutputStream and ObjectInputStream are used.
- Serialization helps in saving object states to persistent storage.

Algorithm

Step-1: Create an Employee class implementing Serializable.

Step-2: Instantiate the class and prepare for serialization.

Step-3: Use ObjectOutputStream to write the object to a file.

Step-4: Use ObjectInputStream to read the object back.

Step-5: Display the deserialized object's details.

Viva Questions

1)What is the purpose of the Serializable interface?

2)Can we serialize static variables in Java?.

3) What is the use of ObjectInputStream and ObjectOutputStream?

4) What happens if a class does not implement Serializable?

5)How do you find the highest salary from a file?

Week - 11

(a) AIM : Write a Java program to implement try, catch, finally blocks.

Description

- Demonstrates exception handling in Java using try, catch, and finally.
- An Array Index Out Of Bounds Exception is thrown and caught.
- The finally block runs whether an exception is thrown or not.
- Shows how programs recover gracefully from runtime errors.
- Ensures execution of cleanup code through finally.

Algorithm

Step-1: Declare an array with integer values.

Step-2: Attempt to access an invalid index inside the try block.

Step-3: Catch the ArrayIndexOutOfBoundsException in the catch block.

Step-4: Print the exception message.

Step-5: In the finally block, print a confirmation message.

(b) AIM : Write a Java program to create user defined Exception and implement throw and throws handlers.

Description

- Demonstrates the creation of a user-defined exception InvalidAgeException.
- Uses throw to manually throw an exception when age is invalid.
- Uses throws to declare the exception in method signature.
- Exception is caught and handled in the main() method.
- Useful for custom error messages and validation scenarios.

Algorithm

Step-1: Create a custom exception class InvalidAgeException by extending Exception.

Step-2: Define a method validateAge(int age) that throws the exception if age < 18.

Step-3: In main(), call validateAge() with invalid age.

Step-4: Catch and handle the thrown exception.

Step-5: Display the message to the user.

Viva Questions

1) What is the use of try, catch, and finally in Java?

2) What is an ArrayIndexOutOfBoundsException?

3) What is a user-defined exception?

4) What is the difference between throw and throws?.

5)Why is exception handling important in Java?

Week - 12

(a) AIM : Write a Java program to create Thread using Thread Class and Runnable Interface.

Description

- This program demonstrates creating threads using two approaches: Thread class and Runnable interface.
- In the first approach, the thread class is extended, and the run() method is overridden.
- In the second approach, the Runnable interface is implemented and passed to the Thread constructor.
- Threads execute concurrently and print messages.
- It highlights Java's multithreading support and syntax flexibility.

Algorithm

Step-1: Create a class extending Thread and override the run() method.

Step-2: In main(), create an object and call start().

Step-3: Create another class implementing Runnable.

Step-4: In main(), pass this class object to a new Thread and call start().

Step-5: Observe parallel execution.

(b) AIM : Write a Java program to implement multi threading and synchronization.

Description

- This program uses synchronized methods to avoid race conditions in multithreading.
- Multiple threads increment a shared counter, and synchronization ensures one thread at a time.
- It prevents inconsistent or incorrect results in concurrent environments.
- join() ensures both threads complete before printing final result.
- Helps in demonstrating thread-safe operations.

Algorithm

Step-1: Create a Counter class with synchronized increment() and getCount() methods.

Step-2: Create IncrementThread class extending Thread.

Step-3: In the run() method, call increment() 1000 times.

Step-4: In main(), create two threads sharing the same Counter object.

Step-5: Start threads, join them, and print final count.

(c) AIM : Write a Java program to implement Inter Thread Communication

Description

- Implements communication between threads using wait() and notify().
- One thread acts as a producer and the other as a consumer.
- Synchronization ensures data integrity and timing between producer and consumer.
- wait() pauses execution until data is produced.
- notify() resumes execution of the waiting thread.

Algorithm

Step-1: Create a SharedData class with methods addData() and getData().

Step-2: Use wait() and notify() to manage access.

Step-3: Create Producer thread that adds data.

Step-4: Create Consumer thread that reads data.

Step-5: Run both threads and monitor output.

Viva Questions

1) What are the two ways to create a thread in Java?

2) Why is synchronization used in multithreading?

3) What is the difference between wait() and sleep()?

4) What is inter-thread communication?

5) Can we call run() method directly instead of start()?

Week - 13

(a) AIM : Create an Applet that changes the Font and background Colour depending upon the user selection from the input.

Description

- This applet allows the user to change font and background color interactively.
- The user selects options from dropdown menus and clicks “Apply”.
- It uses AWT components like Choice, Button, Label.
- Font changes using setFont() and background using setBackground().
- Demonstrates GUI interaction and event handling in applets.

Algorithm

Step-1: Create a FontColorApplet class extending Applet.

Step-2: Initialize AWT components: Choice, Button, and Label.

Step-3: Populate font and color options.

Step-4: Add components to the applet and attach ActionListener.

Step-5: In actionPerformed(), update the font and background based on selection.

(b) AIM : Write a Java program to implement Event Delegation model through AWT Components.

Description

- This program shows the event delegation model using a separate listener class.
- When the user clicks a button, the action is handled by ButtonClickListener.
- Uses ActionListener and addActionListener() method for event handling.
- Demonstrates separation of event logic and UI.
- Commonly used in GUI applications for modular design.

Algorithm

Step-1: Create a Frame and a Button.

Step-2: Implement ActionListener in a separate class.

Step-3: Add the listener to the button using addActionListener().

Step-4: Set frame size, layout, and visibility.

Step-5: Attach a WindowAdapter to close the frame on exit.

(c) AIM : Write a Java program to implement Layout Manager with AWT components.

Description

- This program uses FlowLayout, a layout manager that arranges components in a flow.
- Components are placed in a single row and wrap to the next line if needed.
- Demonstrates usage of layout managers in GUI design.
- Uses AWT components and Frame class.
- Simplifies UI alignment without manual positioning.

Algorithm

Step-1: Create a Frame and set layout to FlowLayout.

Step-2: Create buttons and add them to the frame.

Step-3: Set the frame's size and make it visible.

Step-4: Attach a WindowListener to handle window closing.

Step-5: Observe how buttons align in flow layout.

Viva Questions

1.What is the purpose of the Choice component in AWT?

2.How does the Event Delegation Model work in Java?.

3.What is the use of setLayout() method in Java AWT?

4.What is the difference between FlowLayout and GridLayout?