Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session Wise Lab Exercises

Pre-requisite:

A working computer with Java 17 or higher; Sublime Text or Eclipse IDE to be installed

Session 1: Introduction to OOP Java and Basic Programming constructs

1. Write a program to check if a given number is even or odd
2. Write a program to check if a given number is positive, negative, or zero.
3. Write a program to determine whether a given year is a leap year or not.
4. Write a program to check if a given character is a vowel or a consonant.
5. Write a program to find the largest of three numbers entered by the user.
6. Write a program that takes an integer input (1-7) from the user and prints the corresponding day of the week. Use a switch statement to handle the conversion
7. Write a program that takes two integers and an operator (+, -, *, /) as input and performs the corresponding operation. Use a switch statement to handle the operations.
8. Write a program that takes a character input (R, Y, G) and prints the corresponding traffic light action (Stop, Wait, Go). Use a switch statement to handle the conversion.
9. Write a program to print numbers from 10 to 110
10. Write a program to calculate the sum of all numbers from 1 to 100.
11. Write a program to print the multiplication table of a given number.
12. Write a program to find the factorial of a given number.
13. Write a program to check if a given number is prime.
14. Write a program to print the Fibonacci series up to a given number of terms.
15. Write a program to calculate the sum of digits of a given number.
16. Write a program to check if a given number is a palindrome.
17. Write a program to find the sum of all odd numbers between 1 and 50
18. Write a program to find the sum of all even numbers between 1 and 50
19. Write a program to find the sum of all prime numbers between 1 and 100
20. Write programs to generate the various patterns (triangle, pyramid, etc.)

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 2: Classes, Objects and Constructors

1. Define a class of type Student that has rollno, name and age as private data members. Define SetData() and GetData() as public member functions with appropriate functionality. Write a program that declares 2 student objects, initializes the first at run-time and second by reading from console, and then displays both student's data.
2. Create a class Person with attributes name, age and country. Implement methods to set and get these attributes. Create an object of this class, set its attributes, and print out the details.
3. Constructor Overloading: Extend the Person class from the previous problem and add multiple constructors (default, parameterized, etc.) to initialize the attributes. Also, include a method to display the details.
4. Using this: Modify the Person class to include a method that displays the name and age of the object. Use this keyword to differentiate between class variables and method parameters. Implement a method to compare two Person objects based on their age.
5. Static Variable: Create a class BankAccount with accno, accType, Balance and static variable interestRate. Initialize it using a static block. Implement methods to deposit and withdraw funds. Create objects and display details.
6. Static Method: Add a static method to the BankAccount class from the previous problem to calculate interest based on a given balance and interest rate. Also, implement a method to display the account details including balance and interest earned.
7. Using this in Constructors: Create a class Rectangle with attributes length and width. Implement a parameterized constructor that initializes these attributes. Use this to differentiate between class variables and constructor parameters. Include methods to calculate the area and perimeter.
8. Class and methods: Create a class Calculator with relevant data members and a constructor. Implement methods for basic arithmetic operations (addition, subtraction, multiplication, division, modulus) and demonstrate their usage.
9. Composition and Aggregation: Create a class Address with attributes street, city, and state. Then create a class Person with attributes name and an Address object. Demonstrate how to use com Write a Java class representing a Student. Encapsulate the student's name, age, and grade point average (GPA) with private access modifiers. Provide getter and setter methods to access and modify these attributes position to model the relationship between a person and their address
10. Implement a Java program that models a Library. Create classes for Library, Book, and Author. Ensure that the Library class aggregates a collection of Book objects, and each Book object has an aggregation relationship with an Author object.

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 3: Inheritance, super and polymorphism

1. Create class Box and Box3d. Box3d is extended class of Box. The above two classes going to fulfill the following requirement. Include constructor, set value of length, breadth, height. Find out area and volume.

2. Define a base class Person and a derived class employee with single inheritance. Define SetData() member functions in each of the class with different signatures to set the data members and demonstrate overloading of member functions. Define GetData() member functions in each of the class with same signatures to display data and demonstrate overriding of member functions.

3. Write a program to give example for multilevel inheritance in Java.

4. Demonstrate calling the constructor of the base class from the constructor of the derived class. Create objects of person and employee classes to show the order of invocation of constructors.

5. Create a class with a method that prints "This is parent class" and its subclass with another method that prints "This is child class". Now, create an object for each of the class and call

1 - method of parent class by object of parent class
2 - method of child class by object of child class
3 - method of parent class by object of child class

6. Create a class named 'Member' having the following members:

Data members: 1 – Name, 2 – Age, 3 - Phone number, 4 – Address, 5 - Salary

It also has a method named 'printSalary' which prints the salary of the members. Two classes 'Employee' and 'Manager' inherits the 'Member' class. The 'Employee' and 'Manager' classes have data members'specialization' and 'department' respectively. Now, assign name, age, phone number, address and salary to an employee and a manager by making an object of both of these classes and print the same.

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 4: Abstraction – abstract class and interface, anonymous inner classes

Problem 1: Create an abstract class Shape with an abstract method double area(). Then, create two subclasses, Circle and Rectangle, that extend Shape and provide implementations for the area method. Write a main method to create instances of Circle and Rectangle, and display their areas.

Problem 2: Create an abstract class Animal with an abstract method void sound(). Then, create three subclasses, Dog, Cat, and Cow, each implementing the sound method with their respective sounds. Write a main method to create instances of Dog, Cat, and Cow, and invoke the sound method on each instance.

Problem 3: Create an abstract class Appliance with fields for brand and power consumption, and an abstract method void turnOn(). Create three subclasses, WashingMachine, Refrigerator, and Microwave, each providing their own implementation of the turnOn method. Write a main method to create instances of WashingMachine, Refrigerator, and Microwave, and invoke the turnOn method on each instance to display brand and power consumed.

Problem 4: Task: Create an interface Animal with methods makeSound() and eat(). Implement this interface in two classes Dog and Cat.

Problem 5: Create an interface Vehicle with a default method startEngine() that prints "Engine started". Implement this interface in the class Car and override the startEngine() method.

Problem 6: Interface Inheritance - Create an interface Shape with methods draw() and calculateArea(). Create another interface Colorful that extends Shape and adds a method fillColor(). Implement these interfaces in the class Circle.

Problem 7: Interface with Multiple Implementations - Create an interface Payment with a method pay(). Implement this interface in two classes CreditCardPayment and PaypalPayment. Write a PaymentProcessor class that uses the Payment interface to process payments.

Problem 8: Anonymous Inner Class Implementing an Interface - Create an interface Greeting with a method sayHello(). Write a method generateGreeting() in another class that uses an anonymous inner class to implement the Greeting interface and prints a greeting message

Problem 9: Anonymous Inner Class Extending an Abstract Class - Create an abstract class Shape with an abstract method draw(). Write a method createShape() in another class that uses an anonymous inner class to extend Shape and provides an implementation for the draw() method.

Problem 10: Anonymous Inner Class Extending a Regular Class - Create a class Printer with a method printMessage(). Write a method createPrinter() in another class that uses an anonymous inner class to extend Printer and overrides the printMessage() method.

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 5: Packages, Arrays and Strings

1. Basic Package Creation and Usage: Create a package named com.example.math and add a class Calculator with methods for addition and subtraction. Then, create another class MathTest in the com.example.test package to use the Calculator class
2. Using Multiple Classes from Different Packages - Create a package com.example.shapes with a class Circle and another package com.example.utils with a class MathUtils. The Circle class should use MathUtils (it should have a method to provide PI value) to calculate the area of the circle. Then, create a ShapeTest class to demonstrate this functionality.
3. Creating Sub-Packages - Create a main package com.company with sub-packages accounts and hr. The accounts package should have a class Account with a method displayAccountDetails(), and the hr package should have a class Employee with a method displayEmployeeDetails(). Demonstrate the usage of these classes in a CompanyTest class.
4. Problem 4: Package-Private Access and Public Classes - Create a package com.example.access with two classes: PublicClass and PackagePrivateClass. The PublicClass should have a public method showPublicMessage(), and the PackagePrivateClass should have a package-private method showPackagePrivateMessage(). Create a class AccessTest to demonstrate that the package-private method cannot be accessed outside its package.
5. Program to reverse elements an array
6. Program to find sort an array
7. Program to find sum of squares of odd index values
8. Program to find sum of first and second halves of an array
9. Program to find nth largest / smallest element in array
10. Program to add two matrices
11. Program to multiply two matrices
12. Program to find sum of diagonal elements
13. Write a Java program that takes a string as input and returns the reverse of the string.
14. Check if a String is a Palindrome
15. Write a Java program that checks if a given string is a palindrome (reads the same forward and backward).
16. Count the Number of Vowels and Consonants in a String
17. Write a Java program that counts the number of vowels and consonants in a given string.
18. Find the Longest Word in a Sentence- Write a Java program that finds the longest word in a given sentence. (Hint: String [] words = sentence.split("\\s+");)

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 6: Exception Handling

1: Division by Zero

Task: Write a program that takes two integers from the user and performs division. Use a try block to perform the division, and a catch block to handle the ArithmeticException in case of division by zero. Ensure that a finally block prints a message indicating that the operation is complete.

2: Array Index Out of Bounds

Task: Write a program that initializes an array of integers and tries to access an index that is out of bounds. Use a try block to access the array and a catch block to handle the ArrayIndexOutOfBoundsException. Ensure a finally block prints a message indicating the operation is complete

3: Null Pointer Exception

Task: Write a program that initializes a string variable to null and then tries to call a method on it. Use a try block to call the method and a catch block to handle the NullPointerException. Ensure a finally block prints a message indicating the operation is complete

4: Number Format Exception

Task: Write a program that takes a string input from the user and tries to convert it to an integer. Use a try block to perform the conversion and a catch block to handle the NumberFormatException. Ensure a finally block prints a message indicating the operation is complete.

5: Nested Try Blocks with Multiple Exceptions

Task: Write a program that demonstrates the use of nested try blocks. The program should perform the following tasks:

try should have two separate try blocks.

In the first nested try:

Divide two integers, handling any potential ArithmeticException.

Within the second try block, initialize an array and attempt to access an out-of-bounds index, handling the ArrayIndexOutOfBoundsException.

Ensure that appropriate messages are printed for each exception, and that a final message is printed indicating the completion of the operation

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 7: IO Streams

1. Write a Program to read the same program file and find the no. of lines, words and characters. Write the result in in to a text file (result.txt)

2. Write a program to read the same program file and write it to another file with the lines number added before each line, starting from 1.

3. Write a Java program to read first 3 lines from a file.

4. Write a Java program to find the longest word in a text file

5. Write a program to implement Caeser cipher using files.

   Write to the file (enc_mssage.txt) with using caeser cipher with the displacement value = 3.

   Read the file (enc_message.txt) and decode the Cipher text and write it into a file (dec_message.txt)

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 8: Multithreading

1. Write a program to print "Good morning" and "Welcome" continuously on the screen in Java using threads.

2. Demonstrate gerPriority() and setPriority() , currentThread() methods in Java threads.

3. Write a Java program to implement sleep () and join() methods

4. Write a Word Count program and count words from two or more files files. Write separate threads to count words from a specific file and display word count of each file

5. Write a multi-threaded Java program to print prime numbers between 1 to 1000 and Fibonacci series unto 1000 using two separate threads and find common numbers using the third thread.

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 9: Collection Framework

1: Create and Traverse a List of User-Defined Objects

Write a Java program to create a list of Student objects, add 5 students to it, and traverse the list to print each student's details (id, name, grade).

2: Find the Student with the Highest Grade

Write a Java program to find the student with the highest grade in a list of Student objects

3: Sort Students by Name

Write a Java program to sort a list of Student objects by their name.

4: Create and Traverse a Set of User-Defined Objects

Write a Java program to create a set of Employee objects, add 5 employees to it, and traverse the set to print each employee's details (id, name, department).

5. Using HashSet

Problem: Create a Book class with attributes such as id, title, author, and price. Write a Java program to store a collection of Book objects in a HashSet. Implement methods to add, remove, and search for books by their id.

6. Using TreeSet

Problem: Create an Employee class with attributes such as id, name, salary, and department. Write a Java program to store a collection of Employee objects in a TreeSet and sort them by their name. Implement methods to add, remove, and search for employees by their name.

7. Using LinkedHashSet

Problem: Create a Student class with attributes such as rollNumber, name, and grade. Write a Java program to store a collection of Student objects in a LinkedHashSet while maintaining the order of insertion. Implement methods to add, remove, and display students.

8. Using HashMap

Problem: Create a Product class with attributes such as productId, productName, category, and price. Write a Java program to store a collection of Product objects in a HashMap where the key is the productId. Implement methods to add, remove, and search for products by their productId.

9. Using TreeMap

Problem: Create a Country class with attributes such as countryCode, countryName, and population. Write a Java program to store a collection of Country objects in a TreeMap where the key is the countryCode. Implement methods to add, remove, and search for countries by their countryCode.

Java Demystified

A 5-day Faculty Development Program
(3rd Oct – 9th Oct, 2024)

Session 10: JDBC – Java Database Connectivity

1. Write a program to select specific columns (at least 2 cols) from customers table with multiple conditions (at least two conditions)
2. Write a program to demonstrate sub queries. Provide customer details who is having maximum balance using subquery.
3. Write a program to select data from multiple columns of multiple tables. Use customers and student tables to select 2 fields from each table
4. Program to delete and update a record of student table based on data read from user
5. Write program to read multiple student details (10 records) from user and insert them to student table. Use for loop and PreparedStatement
6. Demonstrate select query using Prepared Statement by reading data from user (Read gpa and provide details of students whose gpa less than that entered)
7. Write a program to perform jdbc batch processing - a. update customers balance (add 10000) where city is hyd and pune. b. Delete customers if balance is lessthan 50000. c & d. Insert 2 customer records and Update customers with balance with + 15000 if cid between 3000 and 7000
8. Demonstrate transaction management - with student table (take suitable case study).
9. Program to Read data from a text file (student.txt) and insert into student table (Use prepared statement)
10. Program to demonstrate where, group by, order by, having with customers table
    (find customers group by city and provide details)
    (find rich customer group by city)
    (Use aggregate functions)