SLIP-1

ANSWER-1

Write a program to accept a number from user and generate multiplication table of a number.

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a number: ");

        int num = scanner.nextInt();

        for (int i = 1; i <= 10; i++) {

            System.out.println(num + " x " + i + " = " + num * i);}}}
```

ANSWER-2

Construct a linked List containing names of colours: red, blue, yellow and orange. Then extend the program to do the following: i. Display the contents of the List using an Iterator ii. Display the contents of the List in reverse order using a ListIterator iii. Create another list containing pink and green. Insert the elements of this list between blue and yellow.

```
 import java.util.*;

public class Main {

    public static void main(String[] args) {

        LinkedList<String> colors = new LinkedList<>(Arrays.asList("red", "blue", "yellow", "orange"));

        Iterator<String> iterator = colors.iterator();

        System.out.println("Original list:");

        while (iterator.hasNext()) {

            System.out.println(iterator.next());}

        ListIterator<String> listIterator = colors.listIterator(colors.size());

        System.out.println("\nList in reverse order:");

        while (listIterator.hasPrevious()) {

            System.out.println(listIterator.previous());}

        List<String> newColors = Arrays.asList("pink", "green");

        int index = colors.indexOf("yellow");

        colors.addAll(index, newColors);

        System.out.println("\nUpdated list:");

        iterator = colors.iterator();

        while (iterator.hasNext()) {

            System.out.println(iterator.next());}}}
```

SLIP-2

ANSWER-1

Write a program to accept ‘n’ integers from the user & store them in an Array List collection. Display the elements of Array List.

```
import java.util.*;
```

```
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements you want to store: ");
        int n = scanner.nextInt();
        ArrayList<Integer> numbers = new ArrayList<>(n);
        System.out.println("Enter " + n + " numbers:");
        for (int i = 0; i < n; i++) {
            numbers.add(scanner.nextInt());}
            System.out.println("The elements of the ArrayList are: " + numbers);}}
```

ANSWER-2

Write a program to accept Doctor Name from the user and check whether it is valid or not.(It should not contain digits and special symbol) If it is not valid then throw user defined Exception - Name is Invalid -- otherwise display the name.

```
import java.util.Scanner;
class InvalidNameException extends Exception {
    InvalidNameException(String s) {
        super(s);}}
public class Main {
    static void validateName(String name) throws InvalidNameException {
        if (!name.matches("[a-zA-Z ]+")) {
            throw new InvalidNameException("Name is Invalid");}}
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the doctor's name: ");
        String name = scanner.nextLine();
        try {
            validateName(name);
            System.out.println("The doctor's name is " + name);}
      catch (InvalidNameException e) {
            System.out.println(e.getMessage());}}}
```

SLIP-3

ANSWER-1

Write a program to accept the 'n' different numbers from user and store it in array. Also print the sum of elements of the array.

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements you want to store: ");
        int n = scanner.nextInt();
        int[] numbers = new int[n];
        System.out.println("Enter " + n + " numbers:");
        for (int i = 0; i < n; i++) {
            numbers[i] = scanner.nextInt();}
        int sum = 0;
        for (int number : numbers) {
            sum += number;}
        System.out.println("The sum of the elements is: " + sum);}}
```

ANSWER-2

Write a program to create class Account (accno, accname, balance). Create an array of 'n' Account objects. Define static method “sortAccount” which sorts the array on the basis of balance. Display account details in sorted order.

```
import java.util.Arrays;
import java.util.Comparator;
class Account {
    int accno;
    String accname;
    double balance;
    public Account(int accno, String accname, double balance) {
        this.accno = accno;
        this.accname = accname;
        this.balance = balance;}
    public String toString() {
        return "Account Number: " + accno + ", Account Name: " + accname + ", Balance: " + balance;}}
public class Main {
    public static void sortAccount(Account[] accounts) {
        Arrays.sort(accounts, Comparator.comparingDouble(a-> a.balance));}
    public static void main(String[] args) {
        Account[] accounts = new Account[3];
        accounts[0] = new Account(1, "Shreyas", 5000.0);
        accounts[1] = new Account(2, "Viraj", 3000.0);
        accounts[2] = new Account(3, "Niraj", 4000.0);
        sortAccount(accounts);
        for (Account account : accounts) {
            System.out.println(account);}}}
```

SLIP-4

ANSWER-1

Write a program to accept the user name and greets the user by name. Before displaying the user's name, convert it to upper case letters. For example, if the user's name is Raj, then display greet message as "Hello, RAJ, nice to meet you!".

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter your name: ");

        String userName = scanner.nextLine();

        String userNameUppercase = userName.toUpperCase();

        System.out.println("Hello, " + userNameUppercase + ", nice to meet you!");

        scanner.close();}}
```

ANSWER-2

Write a program which define class Product with data member as id, name and price. Store the information of 5 products and Display the name of product having minimum price (Use array of object).

```
import java.util.Scanner;

class Product {

    int id;

    String name;

    double price;

    public Product(int id, String name, double price) {

        this.id = id;

        this.name = name;

        this.price = price;}}

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        Product[] products = new Product[5];

        for (int i = 0; i < 5; i++) {

            System.out.print("Enter the name of Product " + (i + 1) + ": ");

            String name = scanner.nextLine();

            System.out.print("Enter the price of Product " + (i + 1) + ": ");

            double price = scanner.nextDouble();

            scanner.nextLine();

            products[i] = new Product(i + 1, name, price);}

        Product minPriceProduct = products[0];

        for (int i = 1; i < products.length; i++) {

            if (products[i].price < minPriceProduct.price) {

                minPriceProduct = products[i];}}
        System.out.println("The product with the minimum price is " + minPriceProduct.name
            + " with a price of " + minPriceProduct.price);}}
```

SLIP-5

ANSWER-1

Write a program to accept a number from the user, if number is zero then throw user defined exception —Number is 0, otherwise display factorial of a number.

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter a number: ");

        int number = scanner.nextInt();

        try {

            if (number == 0) {

                throw new NumberIsZeroException("Number is 0");}

            int factorial = 1;

            for (int i = 1; i <= number; i++) {

                factorial *= i;}

            System.out.println("The factorial of " + number + " is " + factorial);

        } catch (NumberIsZeroException e) {

            System.out.println(e.getMessage());}}

    private static class NumberIsZeroException extends Exception {

        public NumberIsZeroException(String message) {

            super(message);}}}
```

ANSWER-2

Define a “Point” class having members – x,y (coordinates). Define default constructor and parameterized constructors. Define subclass “ColorPoint” with member as color. Write display method to display the details of Point.

```
class Point {

    int x;

    int y;

    public Point() {

        x = 0;

        y = 0;}

    public Point(int x, int y) {

        this.x = x;

        this.y = y;}

    public void display() {

        System.out.println("Coordinates: (" + x + ", " + y + ")");}}
```

```
class ColorPoint extends Point {
    String color;
    public ColorPoint(int x, int y, String color) {
        super(x, y);
        this.color = color;}
    @Override
    public void display() {
        super.display();
        System.out.println("Color: " + color);}}
public class Main {
    public static void main(String[] args) {
        Point point1 = new Point();
        System.out.println("Point 1:");
        point1.display();
        Point point2 = new Point(3, 4);
        System.out.println("\nPoint 2:");
        point2.display();
        ColorPoint colorPoint = new ColorPoint(5, 6, "Red");
        System.out.println("\nColor Point:");
        colorPoint.display();}}
```

SLIP-6

ANSWER-1

Accept 'n' integers from the user and store them in a collection. Display them in the sorted order. The collection should not accept duplicate elements. (Use a suitable collection). Search for a particular element using predefined search method in the Collection framework.

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Set<Integer> integerSet = new HashSet<>();
        System.out.print("Enter the number of integers (n): ");
        int n = scanner.nextInt();
        System.out.println("Enter " + n + " integers:");
        for (int i = 0; i < n; i++) {
            int num = scanner.nextInt();
            integerSet.add(num);}
        List<Integer> sortedList = new ArrayList<>(integerSet);
        Collections.sort(sortedList);
        System.out.println("Unique integers in sorted order: " + sortedList);

        System.out.print("Enter an integer to search for: ");

        int searchElement = scanner.nextInt();

        if (integerSet.contains(searchElement)) {

            System.out.println(searchElement + " is found in the collection.");

        } else {

            System.out.println(searchElement + " is not found in the collection.");}

        scanner.close();}}
```

ANSWER-2

Write a program which define class Employee with data member as id, name and salary Store the information of 'n' employees and Display the name of employee having maximum salary (Use array of object).

```
import java.util.Scanner;

class Employee {

    int id;

    String name;

    double salary;

    public Employee(int id, String name, double salary) {

        this.id = id;

        this.name = name;

        this.salary = salary;}}

public class Main{

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of employees (n): ");

        int n = scanner.nextInt();

        Employee[] employees = new Employee[n];

        for (int i = 0; i < n; i++) {

            System.out.println("Enter details for Employee " + (i + 1) + ":");

            System.out.print("ID: ");

            int id = scanner.nextInt();

            scanner.nextLine();

            System.out.print("Name: ");

            String name = scanner.nextLine();

            System.out.print("Salary: ");

            double salary = scanner.nextDouble();

            employees[i] = new Employee(id, name, salary);}

        Employee maxSalaryEmployee = employees[0];

        for (int i = 1; i < n; i++) {
```

```
        if (employees[i].salary > maxSalaryEmployee.salary) {
            maxSalaryEmployee = employees[i];}}
    System.out.println("Employee with the maximum salary:");
    System.out.println("Name: " + maxSalaryEmployee.name);
    System.out.println("Salary: " + maxSalaryEmployee.salary);
    scanner.close();}}
```

SLIP-7

ANSWER-1

Create a Hash table containing Employee name and Salary. Display the details of the hash table.

```
import java.util.Hashtable;
import java.util.Map;
public class Main {
  public static void main(String[] args) {
    Hashtable<String, Double> employeeHashTable = new Hashtable<>();
    employeeHashTable.put("Rajesh", 50000.0);
    employeeHashTable.put("Sneha", 60000.0);
    employeeHashTable.put("Amit", 55000.0);
    employeeHashTable.put("Priya", 62000.0);
    employeeHashTable.put("Deepa", 48000.0);
    System.out.println("Employee Details:");
    for (Map.Entry<String, Double> entry : employeeHashTable.entrySet()) {
      String name = entry.getKey();
      double salary = entry.getValue();
      System.out.println("Name: " + name + ", Salary: " + salary);}}}
```

ANSWER-2

Define a class student having rollno, name and percentage. Define Default and parameterized constructor. Accept the 5 student details and display it. (Use this keyword).

```
import java.util.Scanner;
class Student {
  int rollNo;
  String name;
  double percentage;
  public Student() {
    this.rollNo = 0;
    this.name = "";
    this.percentage = 0.0;}
  public Student(int rollNo, String name, double percentage) {
    this.rollNo = rollNo;
```

```
        this.name = name;

        this.percentage = percentage;}

    public void display() {

        System.out.println("Roll No: " + this.rollNo);

        System.out.println("Name: " + this.name);

        System.out.println("Percentage: " + this.percentage + "%");}}

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        Student[] students = new Student[5];

        for (int i = 0; i < 5; i++) {

            System.out.println("Enter details for Student " + (i + 1) + ":");

            System.out.print("Roll No: ");

            int rollNo = scanner.nextInt();

            scanner.nextLine();

            System.out.print("Name: ");

            String name = scanner.nextLine();

            System.out.print("Percentage: ");

            double percentage = scanner.nextDouble();

            scanner.nextLine();

            students[i] = new Student(rollNo, name, percentage);}

        System.out.println("Student Details:");

        for (int i = 0; i < 5; i++) {

            System.out.println("\nDetails for Student " + (i + 1) + ":");

            students[i].display();}

        scanner.close();}}
```

SLIP-8

ANSWER-1(cla)

Write a program to reverse a number. Accept number using command line argument.

```
public class Main {

    public static void main(String[] args) {

        if (args.length != 1) {

            System.out.println("Usage: java ReverseNumber <number>");

            return;}

        int number = Integer.parseInt(args[0]);

        int reversedNumber = reverseNumber(number);

        System.out.println("Original Number: " + number);

        System.out.println("Reversed Number: " + reversedNumber);}
    public static int reverseNumber(int number) {
        int reversed = 0;
        while (number != 0) {
            int digit = number % 10;
            reversed = reversed * 10 + digit;
            number /= 10;}
        return reversed;}}
```

ANSWER-2

Define a class MyDate (day, month, year) with methods to accept and display MyDate object. Accept date as dd, mm, yyyy. Throw user defined exception "InvalidDateException" ifthe date is invalid. Examples of invalid dates : 12 15 2015, 31 6 1990, 29 2 2001

```
class InvalidDateException extends Exception {
    public InvalidDateException(String message) {
        super(message);}}
class MyDate {
    int day;
    int month;
    int year;
    public MyDate(int day, int month, int year) throws InvalidDateException {
        if (!isValidDate(day, month, year)) {
            throw new InvalidDateException("Invalid date: " + day + " " + month + " " + year);}
        this.day = day;
        this.month = month;
        this.year = year;}
    public void display() {
        System.out.println("Date: " + day + "/" + month + "/" + year);}
    private boolean isValidDate(int day, int month, int year) {
        if (year < 1 || month < 1 || month > 12) {
            return false;}
        int[] daysInMonth = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
        if (isLeapYear(year) && month == 2) {
            return day >= 1 && day <= 29;
        } else {
            return day >= 1 && day <= daysInMonth[month];}}
    private boolean isLeapYear(int year) {
        return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);}}
public class Main {
    public static void main(String[] args) {
        try {
            MyDate date1 = new MyDate(12, 15, 2015);

            date1.display();

        } catch (InvalidDateException e) {

            System.out.println("Error: " + e.getMessage());}

        try {

            MyDate date2 = new MyDate(31, 6, 1990);

            date2.display();

        } catch (InvalidDateException e) {

            System.out.println("Error: " + e.getMessage());}

        try {

            MyDate date3 = new MyDate(29, 2, 2001);

            date3.display();

        } catch (InvalidDateException e) {

            System.out.println("Error: " + e.getMessage());}

        try {

            MyDate date4 = new MyDate(31, 12, 2022);

            date4.display();

        } catch (InvalidDateException e) {

            System.out.println("Error: " + e.getMessage());}}}
```

SLIP-9

ANSWER-1

Write a program to accept a number from user. Check whether number is perfect or not. Use BufferedReader class for accepting input from user.

```
import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main {

    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Enter a number: ");

        int num = Integer.parseInt(br.readLine());

        int sum = 1;

        for (int i = 2; i <= num / 2; i++) {

            if (num % i == 0) {

                sum += i;}}

        System.out.println(num + (sum == num ? " is a perfect number." : " is not a perfect number."));

        br.close();}}
```

ANSWER-2(yourfile.txt)

Write a program that displays the number of characters, lines and words of a file.

```
import java.io.*;

public class Main {

    public static void main(String[] args) {

        String filePath = "yourfile.txt";

        int characterCount = 0;

        int wordCount = 0;

        int lineCount = 0;

        try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {

            String line;

            while ((line = reader.readLine()) != null) {

                lineCount++;

                characterCount += line.length();

                String[] words = line.split("\\s+");

                wordCount += words.length;}

        } catch (IOException e) {

            System.err.println("Error reading the file: " + e.getMessage());

            System.exit(1);}

        System.out.println("Number of characters: " + characterCount);

        System.out.println("Number of words: " + wordCount);

        System.out.println("Number of lines: " + lineCount);}}
```

SLIP-10

ANSWER-1

Write a program to accept a number from user. Check whether number is prime or not.

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a number: ");

        int n = scanner.nextInt();

        boolean isPrime = n > 1 && java.math.BigInteger.valueOf(n).isProbablePrime(100);

        System.out.println(n + (isPrime ? " is a prime number." : " is not a prime number."));

        scanner.close();}}
```

ANSWER-2

Define a class SavingAccount (acno, name, balance). Define appropriate operations as, withdraw(), deposit(), and viewbalance(). The minimum balance must be 500. Create an object and perform operation. Raise user defined —InsufficientFundException when balance is not sufficient for withdraw operation.

```
class InsufficientFundException extends Exception {
    public InsufficientFundException(String message) {
        super(message);}}
public class Main {
    private int acno;
    private String name;
    private double balance;
    private static final double MIN_BALANCE = 500.0;
    public Main(int acno, String name, double balance) {
        this.acno = acno;
        this.name = name;
        this.balance = balance;}
    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposit successful. New balance: " + balance);}
    public void withdraw(double amount) throws InsufficientFundException {
        if (balance- amount < MIN_BALANCE) {
            throw new InsufficientFundException("Insufficient funds to withdraw " + amount);}
        balance-= amount;
        System.out.println("Withdrawal successful. New balance: " + balance);}
    public void viewBalance() {
        System.out.println("Account Number: " + acno);
        System.out.println("Account Holder: " + name);
        System.out.println("Current Balance: " + balance);}
    public static void main(String[] args) {
        Main account = new Main(12345, "Divya", 1000.0);
        try {
            account.viewBalance();
            account.deposit(500.0);
            account.withdraw(800.0);
        } catch (InsufficientFundException e) {
            System.out.println("Error: " + e.getMessage());}}}
```

SLIP-11

ANSWER-1

Write a program create class as MyDate with dd,mm,yy as data members. Write parameterized constructor. Display the date in dd-mm-yy format. (Use this keyword)

```
public class Main {
    private int dd;
    private int mm;
    private int yy;
    public Main(int dd, int mm, int yy) {
        this.dd = dd;
        this.mm = mm;
        this.yy = yy;}
    public void displayDate() {
        String formattedDate = String.format("%02d-%02d-%02d", dd, mm, yy);
        System.out.println("Date: " + formattedDate);}
    public static void main(String[] args) {
        Main date = new Main(8, 10, 23);
        date.displayDate();}}
```

ANSWER-2

Create an abstract class Shape with methods area & volume. Derive a class Sphere (radius). Calculate and display area and volume.

```
abstract class Shape {
    abstract double area();
    abstract double volume();}
class Sphere extends Shape {
    private double radius;
    public Sphere(double radius) {
        this.radius = radius;}
    @Override
    double area() {
        return 4 * Math.PI * radius * radius;}
    @Override
    double volume() {
        return (4.0 / 3.0) * Math.PI * Math.pow(radius, 3);}}
public class Main {
    public static void main(String[] args) {
        double radius = 5.0;
        Sphere sphere = new Sphere(radius);
        System.out.println("Sphere Area: " + sphere.area());
        System.out.println("Sphere Volume: " + sphere.volume());}}
```

SLIP-12

ANSWER-1

Create a package named “Series” having a class to print series of Square of numbers. Write a program to generate “n” terms series.

```
public class Main {
    public static void main(String[] args) {
        int n = 20;
        SquareSeries.printSquareSeries(n);}}
class SquareSeries {
    public static void printSquareSeries(int n) {
        for (int i = 1; i <= n; i++) {
            System.out.print(i * i + " ");}
        System.out.println();}}
```

ANSWER-2

Create an abstract class Shape with methods area & volume. Derive a class Cylinder (radius, height). Calculate area and volume.

```
abstract class Shape {
    abstract double area();
    abstract double volume();}
class Cylinder extends Shape {
    private double radius;
    private double height;
    public Cylinder(double radius, double height) {
        this.radius = radius;
        this.height = height;}
    @Override
    double area() {
        return 2 * Math.PI * radius * (radius + height);}
    @Override
    double volume() {
        return Math.PI * radius * radius * height;}}
public class Main {
    public static void main(String[] args) {
        double radius = 5.0;
        double height = 10.0;
        Cylinder cylinder = new Cylinder(radius, height);
        System.out.println("Cylinder Surface Area: " + cylinder.area());
        System.out.println("Cylinder Volume: " + cylinder.volume());}}
```

SLIP-13

ANSWER-1

Construct a Linked List having names of Fruits: Apple, Banana, Guava and Orange. Display the contents of the List using an Iterator.

```
import java.util.LinkedList;
import java.util.Iterator;
public class Main {
    public static void main(String[] args) {
        LinkedList<String> fruitList = new LinkedList<>();
        fruitList.add("Apple");

        fruitList.add("Banana");

        fruitList.add("Guava");

        fruitList.add("Orange");

        Iterator<String> iterator = fruitList.iterator();

        while (iterator.hasNext()) {

            System.out.println(iterator.next());}}}
```

ANSWER-2

Define an interface "Operation" which has methods area(),volume(). Define a constant PI having value 3.142. Create a class circle (member – radius) which implements this interface. Calculate and display the area and volume.

```
interface Operation {

    double PI = 3.142;

    double area();

    double volume();}

class Circle implements Operation {

    private double radius;

    public Circle(double radius) {

        this.radius = radius;}

    @Override

    public double area() {

        return PI * radius * radius;}

    @Override

    public double volume() {

        return 0; }}

public class Main {

    public static void main(String[] args) {

        double radius = 5.0;

        Circle circle = new Circle(radius);

        System.out.println("Circle Area: " + circle.area());

        System.out.println("Circle Volume: " + circle.volume());}}
```

SLIP-14

ANSWER-1

Write a program to create JDBC connection. On successful connection with database display appropriate message to user.

```
import java.sql.*;

public class Main {

    public static void main(String[] args) {

        String url = "jdbc:mysql://localhost:3306/";

        String dbName = "test_database";
        String username = "your_username";

        String password = "your_password";

        try {

            Connection connection = DriverManager.getConnection(url, username, password);

            Statement statement = connection.createStatement();

            statement.executeUpdate("CREATE DATABASE IF NOT EXISTS " + dbName);

            statement.executeUpdate("USE " + dbName);

            statement.executeUpdate("CREATE TABLE IF NOT EXISTS users (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(50), age 
INT)");

            statement.executeUpdate("INSERT INTO users (name, age) VALUES ('John', 30), ('Alice', 25), ('Bob', 28)");

            System.out.println("Database created with table and data inserted!");

            connection.close();

        } catch (SQLException e) {

            System.out.println("Failed to create database or insert data.");

            e.printStackTrace();}}}
```

ANSWER-2

Define an interface “Operation” which has methods area(),volume(). Define a constant PI having a value 3.142. Create a class cylinder (members – radius, height) which implements this interface. Calculate and display the area and volume.

```
interface Operation {

    double PI = 3.142;

    double area();

    double volume(); }

class Cylinder implements Operation {

    private double radius;

    private double height;

    public Cylinder(double radius, double height) {

        this.radius = radius;

        this.height = height;}

    @Override

    public double area() {

        return 2 * PI * radius * (radius + height);}

    @Override

    public double volume() {

        return PI * radius * radius * height;}}

public class Main {

    public static void main(String[] args) {

        double radius = 5.0;

        double height = 10.0;
        Cylinder cylinder = new Cylinder(radius, height);

        System.out.println("Cylinder Surface Area: " + cylinder.area());

        System.out.println("Cylinder Volume: " + cylinder.volume());}}
```

SLIP-15

ANSWER-1

Construct a Linked List having names of Fruits: Apple, Banana, Guava and Orange. Display the contents of the List in reverse order using an appropriate interface.

```
import java.util.LinkedList;

import java.util.List;

import java.util.ListIterator;

public class Main {

    public static void main(String[] args) {

        LinkedList<String> fruitList = new LinkedList<>();

        fruitList.add("Apple");

        fruitList.add("Banana");

        fruitList.add("Guava");

        fruitList.add("Orange");

        ListIterator<String> iterator = fruitList.listIterator(fruitList.size());

        while (iterator.hasPrevious()) {

            System.out.println(iterator.previous());}}}
```

ANSWER-2

Write a program to create a super class Employee (members – name, salary). Derive a sub- class as Developer (member – projectname). Create object of Developer and display the details of it.

```
class Employee {

    protected String name;

    protected double salary;

    public Employee(String name, double salary) {

        this.name = name;

        this.salary = salary;}

    public void displayDetails() {

        System.out.println("Employee Name: " + name);

        System.out.println("Employee Salary: $" + salary);}}

class Developer extends Employee {

    private String projectName;

    public Developer(String name, double salary, String projectName) {

        super(name, salary);

        this.projectName = projectName;}

    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Developer Project: " + projectName);}}
public class Main {
    public static void main(String[] args) {
        Developer developer = new Developer("Shreyas", 75000.0, "Project X");
        developer.displayDetails();}}
```

SLIP-16

ANSWER-1(cla)

Define a class MyNumber having one private integer data member. Write a parameterized constructor to initialize to a value. Write isEven() to check given number is even or not. Use command line argument to pass a value to the object.

```
public class Main {
    private int value;
    public Main(int value) {
        this.value = value;}
    public boolean isEven() {
        return value % 2 == 0;}
    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Please provide a single integer as a command-line argument.");
            return;}
        try {
            int inputValue = Integer.parseInt(args[0]);
            Main number = new Main(inputValue);
            if (number.isEven()) {
                System.out.println(inputValue + " is an even number.");
            } else {
                System.out.println(inputValue + " is an odd number.");}
        } catch (NumberFormatException e) {
            System.out.println("Invalid input. Please provide a valid integer as a command-line argument.");}}}
```

ANSWER-2

Write a program to create a super class Employee (members – name, salary). Derive a sub- class Programmer (member – proglanguage). Create object of Programmer and display the details of it.

```
class Employee {
    protected String name;
    protected double salary;
    public Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;}
    public void displayDetails() {
        System.out.println("Employee Name: " + name);
        System.out.println("Employee Salary: " + salary);}}
class Programmer extends Employee {
    private String progLanguage;
    public Programmer(String name, double salary, String progLanguage) {
        super(name, salary);
        this.progLanguage = progLanguage;}
    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Programmer Language: " + progLanguage);}}
public class Main {
    public static void main(String[] args) {
        Programmer programmer = new Programmer("Shreyas", 80000000.0, "Java");
        programmer.displayDetails();}}
```

SLIP-17

ANSWER-1(cla)

Define a class MyNumber having one private integer data member. Write a parameterized constructor to initialize to a value. Write isOdd() to check given number is odd or not. Use command line argument to pass a value to the object.

```
public class Main {
    private int value;
    public Main(int value) {
        this.value = value;}
    public boolean isOdd() {
        return value % 2 != 0;}
    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Please provide a single integer as a command-line argument.");
            return;}
        try {
            int inputValue = Integer.parseInt(args[0]);
            Main number = new Main(inputValue);
            if (number.isOdd()) {
                System.out.println(inputValue + " is an odd number.");
            } else {
                System.out.println(inputValue + " is an even number.");}
        } catch (NumberFormatException e) {
            System.out.println("Invalid input. Please provide a valid integer as a command-line argument.");}}}
```

ANSWER-2

Define a class Student with attributes rollno and name. Define default and parameterized constructor. Keep the count of Objects created. Create objects using parameterized constructor and Display the object count after each object is created.

```
class Student {
    private int rollno;
    private String name;
    private static int objectCount = 0;
    public Student() {
        objectCount++; }
    public Student(int rollno, String name) {
        this.rollno = rollno;
        this.name = name;
        objectCount++; }
    public static int getObjectCount() {
        return objectCount;}}
public class Main {
    public static void main(String[] args) {
        Student student1 = new Student(101, "Shreyas");
        System.out.println("Object Count after student1: " + Student.getObjectCount());
        Student student2 = new Student(102, "Viraj");
        System.out.println("Object Count after student2: " + Student.getObjectCount());
        Student student3 = new Student();
        System.out.println("Object Count after student3: " + Student.getObjectCount());}}
```

SLIP-18

ANSWER-1(cla)

Write a program to print the factors of a number. Accept a number using command line argument.

```
public class Main {
    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Please provide a single integer as a command-line argument.");
            return;}
        try {
            int number = Integer.parseInt(args[0]);
            if (number < 1) {
                System.out.println("Please provide a positive integer as a command-line argument.");
                return;}
            System.out.print("Factors of " + number + ": ");
            for (int i = 1; i <= number; i++) {
                if (number % i == 0) {
                    System.out.print(i + " ");}}
        } catch (NumberFormatException e) {
            System.out.println("Invalid input. Please provide a valid integer as a command-line argument.");}}}
```

ANSWER-2(abc.txt)

Write a program to read the contents of “abc.txt” file. Display the contents of file in uppercase as output.

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class Main {
    public static void main(String[] args) {
        String filename = "abc.txt";
        try (BufferedReader br = new BufferedReader(new FileReader(filename))) {
            String line;
            while ((line = br.readLine()) != null) {
                System.out.println(line.toUpperCase());}
        } catch (IOException e) {
            System.err.println("Error reading the file: " + e.getMessage());}}}
```

SLIP-19

ANSWER-1

Write a program to accept the 'n' different numbers from user and store it in array. Display maximum number from an array.

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements (n): ");
        int n = scanner.nextInt();
        if (n <= 0) {
            System.out.println("Please enter a valid positive number for 'n'.");
            return;}
        int[] numbers = new int[n];
        for (int i = 0; i < n; i++) {
            System.out.print("Enter number " + (i + 1) + ": ");
            numbers[i] = scanner.nextInt();}
        int maxNumber = numbers[0];
        for (int i = 1; i < n; i++) {
            if (numbers[i] > maxNumber) {
                maxNumber = numbers[i];}}
        System.out.println("Maximum number in the array: " + maxNumber);
        scanner.close();}}
```

ANSWER-2

Create an abstract class “order” having members id, description. Create a subclass “Purchase Order” having member as customer name. Define methods accept and display. Create 3 objects each of Purchase Order. Accept and display the details.

```
abstract class Order {
    protected int id;
    protected String description;
    public Order(int id, String description) {
        this.id = id;
        this.description = description;}
    public abstract void accept();
    public abstract void display();}
class PurchaseOrder extends Order {
    private String customerName;
    public PurchaseOrder(int id, String description, String customerName) {
        super(id, description);
        this.customerName = customerName;}
    @Override
    public void accept() {}
    @Override
    public void display() {
        System.out.println("Order ID: " + id);
        System.out.println("Order Description: " + description);
        System.out.println("Customer Name: " + customerName);
        System.out.println();}}
public class Main {
    public static void main(String[] args) {
        PurchaseOrder order1 = new PurchaseOrder(1, "Product A", "Shreyas");
        PurchaseOrder order2 = new PurchaseOrder(2, "Product B", "Viraj");
        PurchaseOrder order3 = new PurchaseOrder(3, "Product C", "Niraj");
        System.out.println("Purchase Orders:");
        order1.display();
        order2.display();
        order3.display();}}
```

SLIP-20

ANSWER-1(cla)

Write a program to accept 3 numbers using command line argument. Sort and display the numbers.

```
public class Main {

    public static void main(String[] args) {

        if (args.length != 3) {

            System.out.println("Please provide exactly three numbers as command-line arguments.");

            return;}

        try {

            double num1 = Double.parseDouble(args[0]);

            double num2 = Double.parseDouble(args[1]);

            double num3 = Double.parseDouble(args[2]);

            double[] numbers = {num1, num2, num3};

            java.util.Arrays.sort(numbers);

            System.out.println("Sorted Numbers:");

            for (double number : numbers) {

                System.out.println(number);}

        } catch (NumberFormatException e) {

            System.out.println("Invalid input. Please provide valid numeric values as command-line arguments.");}}}
```

ANSWER-2

Create an employee class (id,name,deptname,salary). Define a default and parameterized constructor. Use 'this' keyword to initialize instance variables. Keep a count of objects created. Create objects using parameterized constructor and display the object count after each object is created. Also display the contents of each object.

```
import java.util.Scanner;

class Employee {

    private int id;

    private String name;

    private String deptName;

    private double salary;

    private static int objectCount = 0;

    public Employee() {

        this.id = 0;

        this.name = "";

        this.deptName = "";

        this.salary = 0.0;

        objectCount++; }

    public Employee(int id, String name, String deptName, double salary) {

        this();

        this.id = id;
        this.name = name;
        this.deptName = deptName;
        this.salary = salary;
        objectCount++; }
    public void display() {
        System.out.println("Employee ID: " + id);
        System.out.println("Employee Name: " + name);
        System.out.println("Department Name: " + deptName);
        System.out.println("Salary: $" + salary);}
    public static int getObjectCount() {
        return objectCount;}}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of employees to create: ");
        int numberOfEmployees = scanner.nextInt();
        scanner.nextLine();
        for (int i = 0; i < numberOfEmployees; i++) {
            System.out.println("\nEnter details for Employee #" + (i + 1));
            System.out.print("Enter Employee ID: ");
            int id = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Enter Employee Name: ");
            String name = scanner.nextLine();
            System.out.print("Enter Department Name: ");
            String deptName = scanner.nextLine();
            System.out.print("Enter Salary: ");
            double salary = scanner.nextDouble();
            scanner.nextLine();
            Employee employee = new Employee(id, name, deptName, salary);
            System.out.println("Object Count after employee" + (i + 1) + ": " + Employee.getObjectCount());
            employee.display();}
        scanner.close();}}
```