**VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES-Technical Campus**

**VIVEKANANDA SCHOOL OF INFORMATION TECHNOLOGY**



**PRACTICAL FILE**

**Data Structures and Algorithms**

**(BCA 106P)**

**BACHELOR OF COMPUTER APPLICATIONS**

Affiliated to

GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY



**SUBMITTED TO: SUBMITTED BY:**

Dr. Sakshi Name: Saksham Srivastava

Assistant Professor Enrollment no: 05617702024

VSIT, VIPS BCA-II A

**INDEX**

| **Serial** | **Question** | **Signature** |
| --- | --- | --- |
| 1 | Write a Program to find greatest of three numbers using Ternary Operator. |  |
| 2 | Write a Program to display Hello world. |  |
| 3 | WAP to accept two numbers and display the sum. |  |
| 4 | Write a Program to read a floating point number and print Integer and Float part of the number separately. |  |
| 5 | Write a Program to calculate displacement using formulae s=ut + 1/2at² given values of a, u, t by the user. |  |
| 6 | Write a Program to check whether a number is Even or Odd. |  |
| 7 | Write a Program to print table of a number. |  |
| 8 | Write a Program to print factorial of a number. |  |
| 9 | Write a Program to print the following Pattern: |  |
|  | 5 5 5 5 5 |  |
|  | 4 4 4 4 |  |
|  | 3 3 3 |  |
|  | 2 2 |  |
|  | 1 |  |
| 10 | Write a Program to print the following Pattern: |  |
|  | 1 |  |
|  | 1 2 |  |
|  | 1 2 3 |  |
|  | 1 2 3 4 |  |
| 11 | Write a Program to swap two numbers without using the third variable. |  |
| 12 | Write a Program to check whether a string is Palindrome or not. |  |
| 13 | Write a Program to calculate sum and sum of squares of first 15 Even Numbers. |  |
| 14 | Write a Program to check whether a number is prime or not. |  |
| 15 | Write a Program to convert Binary Number to Decimal Number. |  |
| 16 | Write a Program to check whether a number is Armstrong or not. |  |
| 17 | Write a Program to calculate Simple Interest using the concept of classes. |  |
| 18 | Write a Program to illustrate the concept of Static Member Data and Static Member Function. |  |
| 19 | Write a Program to make a Simple Calculator using the concept of classes. |  |
| 20 | Write a Menu Driven Program to Add, Subtract, Multiply two matrices of order 2X2 using concepts of Object Oriented Programming. |  |
| 21 | Write a Program to swap two integer values, two floating point values, two character values using function overloading. |  |
| 22 | WAP that creates a class Accounts with following details: Instance variables: ac\_no., name, ac\_name, balance. Methods: withdrawal(), deposit(), display(). Use constructors to initialize members. |  |
| 23 | WAP to implement constructor overloading. |  |
| 24 | WAP to count the number of objects created in a program. |  |
| 25 | WAP to show call by value & call by reference. |  |
| 26 | WAP to implement method overriding & method overloading. |  |
| 27 | WAP that demonstrates all the usages of “super” keyword. |  |
| 28 | Create a class box having height, width, depth as the instance variables & calculate its volume. Implement constructor overloading in it. Create a subclass named box\_new that has weight as an instance variable. Use super in the box\_new class to initialize members of the base class. |  |
| 29 | WAP that implements multilevel inheritance. |  |
| 30 | Identify the type of inheritance and implement it by modelling the Examination Database: person <---- student <---- exam. |  |
| 31 | Which type of inheritance is this? Illustrate this inheritance by writing a program assuming your own data members. |  |
| 32 | Consider a university where students who participate in the national games or Olympics are given some grace marks. Therefore, the final marks awarded = Exam\_Marks + Sports\_Grace\_Marks. A class diagram representing this scenario is as follow; |  |
| 33 | WAP to implement Run time polymorphism. |  |
| 34 | WAP to implement interface. Create an interface named Shape having area() & perimeter() as its methods. Create three classes circle, rectangle & square that implement this interface. |  |
| 35 | WAP to show multiple inheritance. |  |
| 36 | WAP to implement exception handling. The program should accept two numbers from the user & divide the first number by the second. It should throw an Arithmetic Exception if an attempt is made to divide the number by zero. Use try, catch & finally. Implement multi-catch statements also. |  |
| 37 | Create a user defined exception named “NoMatchException” that is fired when the number entered by the user is not 10. Use the throws & throw keyword. |  |
| 38 | WAP that creates three threads which print numbers from 1 to 5, 6 to 10 and 11 to 15 respectively. Set the name & priority of the threads. |  |
| 39 | WAP to print even & odd numbers using threads. |  |
| 40 | WAP that implements the concept of synchronization in threads using both synchronized method and synchronized block. |  |
| 41 | WAP that demonstrates the use of sleep and join methods in thread. Use minimum three threads. |  |
| 42 | WAP to demonstrate the use of equals(), trim(), length(), substring(), compareTo() of String class. |  |
| 43 | WAP to implement file handling. The program should copy the content from one file to another. |  |

Question 1:

Write a Program to find greatest of three numbers using Ternary Operator.

Source Code:

|  |
| --- |
| // 1. Program to find greatest of three numbers using Ternary Operator  import java.util.Scanner;  class GreatestOfThree {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int a, b, c, greatest;  System.out.print("Enter first number: ");  a = sc.nextInt();  System.out.print("Enter second number: ");  b = sc.nextInt();  System.out.print("Enter third number: ");  c = sc.nextInt();  greatest = (a>b)?((a>c)?a:c):((b>c)?b:c);  System.out.println("Greatest number is: " + greatest);  }  } |

Output:

|  |
| --- |
| Enter first number: 45  Enter second number: 72  Enter third number: 66  Greatest number is: 72 |

Question 2:

Write a Program to display Hello world.

Source Code:

|  |
| --- |
| // 2. Program to display Hello world  class HelloWorld {  public static void main(String[] args) {  System.out.println("Hello World");  }  } |

Output:

|  |
| --- |
| Hello World |

Question 3:

WAP to accept two no.s and display the sum.

Source Code:

|  |
| --- |
| // 3. Program to accept two numbers and display the sum  import java.util.Scanner;  class SumTwoNumbers {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num1, num2, sum;  System.out.print("Enter first number: ");  num1 = sc.nextInt();  System.out.print("Enter second number: ");  num2 = sc.nextInt();  sum = num1 + num2;  System.out.println("Sum is: " + sum);  }  } |

Output:

|  |
| --- |
| Enter first number: 23  Enter second number: 45  Sum is: 68 |

Question 4:

Write a Program to read a floating point number and print Integer and Float part of the number separately.

Source Code:

|  |
| --- |
| // 4. Program to read a floating point number and print Integer and Float part of the number separately  import java.util.Scanner;  class FloatParts {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.print("Enter a floating point number: ");  float num = sc.nextFloat();  int intPart = (int)num;  float floatPart = num - intPart;  System.out.println("Integer part: " + intPart);  System.out.println("Floating part: " + floatPart);  }  } |

Output:

|  |
| --- |
| Enter a floating point number: 123.456  Integer part: 123  Floating part: 0.45600128 |

Question 5:

Write a Program to calculate displacement using formulae s=ut+1/2at2 given values of a,u,t by the user

Source Code:

|  |
| --- |
| // 5. Program to calculate displacement using formulae s=ut+1/2at2 given values of a,u,t by the user  import java.util.Scanner;  class Displacement {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  double u, a, t, s;  System.out.print("Enter initial velocity (u): ");  u = sc.nextDouble();  System.out.print("Enter acceleration (a): ");  a = sc.nextDouble();  System.out.print("Enter time (t): ");  t = sc.nextDouble();  s = (u \* t) + (0.5 \* a \* t \* t);  System.out.println("Displacement is: " + s);  }  } |

Output:

|  |
| --- |
| Enter a floating point number: 123.456  Integer part: 123  Floating part: 0.45600128 |

Question 6:

Write a Program to check weather a number is Even or Odd

Source Code:

|  |
| --- |
| // 6. Program to check whether a number is Even or Odd  import java.util.Scanner;  class EvenOdd {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num;  System.out.print("Enter a number: ");  num = sc.nextInt();  if(num%2==0) {  System.out.println(num + " is Even");  } else {  System.out.println(num + " is Odd");  }  }  } |

Output:

|  |
| --- |
| Enter a number: 7  7 is Odd  Enter a number: 8  8 is Even |

Question 7:

Write a Program to print table of a number

Source Code:

|  |
| --- |
| // 7. Program to print table of a number  import java.util.Scanner;  class MultiplicationTable {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num;  System.out.print("Enter a number: ");  num = sc.nextInt();  for(int i = 1; i <= 10; i++) {  System.out.println(num + " x " + i + " = " + (num \* i));  }  }  } |

Output:

|  |
| --- |
| Enter a number: 5  5 x 1 = 5  5 x 2 = 10  5 x 3 = 15  5 x 4 = 20  5 x 5 = 25  5 x 6 = 30  5 x 7 = 35  5 x 8 = 40  5 x 9 = 45  5 x 10 = 50 |

Question 8:

Write a Program to print factorial of a number

Source Code:

|  |
| --- |
| // 8. Program to print factorial of a number  import java.util.Scanner;  class Factorial {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num, fact = 1;  System.out.print("Enter a number: ");  num = sc.nextInt();  for(int i = 1; i <= num; i++) {  fact \*= i;  }  System.out.println("Factorial of " + num + " is: " + fact);  }  } |

Output:

|  |
| --- |
| Enter a number: 5  Factorial of 5 is: 120 |

Question 9:

Write a Program to print the following Pattern

5 5 5 5 5

4 4 4 4

3 3 3

2 2

1

Source Code:

|  |
| --- |
| // 9. Program to print the following Pattern  class Pattern {  public static void main(String[] args) {  for(int i = 5; i >= 1; i--) {  for(int j = 1; j <= i; j++) {  System.out.print(i + " ");  }  System.out.println();  }  }  } |

Output:

|  |
| --- |
| 5 5 5 5 5  4 4 4 4  3 3 3  2 2  1 |

Question 10:

Write a Program to print the following Pattern

1

1 2

1 2 3

1 2 3 4

Source Code:

|  |
| --- |
| // 10. Program to print the following Pattern  class Pattern2 {  public static void main(String[] args) {  for(int i = 1; i <= 4; i++) {  for(int j = 1; j <= i; j++) {  System.out.print(j + " ");  }  System.out.println();  }  }  } |

Output:

|  |
| --- |
| 1  1 2  1 2 3  1 2 3 4 |

Question 11:

Write a Program to swap two numbers without using the third variable

Source Code:

|  |
| --- |
| // 11. Program to swap two numbers without using the third variable  import java.util.Scanner;  class SwapWithoutThirdVariable {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int a, b;  System.out.print("Enter first number: ");  a = sc.nextInt();  System.out.print("Enter second number: ");  b = sc.nextInt();  a = a + b;  b = a - b;  a = a - b;  System.out.println("After swapping: ");  System.out.println("First number: " + a);  System.out.println("Second number: " + b);  }  } |

Output:

|  |
| --- |
| Enter first number: 5  Enter second number: 10  After swapping:  First number: 10  Second number: 5 |

Question 12:

Write a Program to check weather a string is Palindrome or not

Source Code:

|  |
| --- |
| // 12. Program to check whether a string is Palindrome or not  import java.util.Scanner;  class Palindrome {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  String str, reverse = "";  System.out.print("Enter a string: ");  str = sc.nextLine();  for(int i = str.length() - 1; i >= 0; i--) {  reverse += str.charAt(i);  }  if(str.equals(reverse)) {  System.out.println(str + " is a Palindrome");  } else {  System.out.println(str + " is not a Palindrome");  }  }  } |

Output:

|  |
| --- |
| Enter a string: madam  madam is a Palindrome  Enter a string: hello  hello is not a Palindrome |

Question 13:

Write a Program to calculate sum and sum of squares of first 15 Even Numbers

Source Code:

|  |
| --- |
| // 13. Program to calculate sum and sum of squares of first 15 Even Numbers  class EvenNumbersSum {  public static void main(String[] args) {  int sum = 0, sumOfSquares = 0;  for(int i = 2; i <= 30; i += 2) {  sum += i;  sumOfSquares += i \* i;  }  System.out.println("Sum of first 15 even numbers: " + sum);  System.out.println("Sum of squares of first 15 even numbers: " + sumOfSquares);  }  } |

Output:

|  |
| --- |
| Sum of first 15 even numbers: 240  Sum of squares of first 15 even numbers: 9000 |

Question 14:

Write a Program to check weather a number is prime or not

Source Code:

|  |
| --- |
| // 14. Program to check whether a number is prime or not  import java.util.Scanner;  class PrimeNumber {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num;  boolean isPrime = true;  System.out.print("Enter a number: ");  num = sc.nextInt();  for(int i = 2; i <= num / 2; i++) {  if(num % i == 0) {  isPrime = false;  break;  }  }  if(isPrime && num > 1) {  System.out.println(num + " is a Prime number");  } else {  System.out.println(num + " is not a Prime number");  }  }  } |

Output:

|  |
| --- |
| Enter a number: 7  7 is a Prime number  Enter a number: 9  9 is not a Prime number |

Question 15:

Write a Program to convert Binary Number to Decimal Number

Source Code:

|  |
| --- |
| // 15. Program to convert Binary Number to Decimal Number  import java.util.Scanner;  class BinaryToDecimal {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  String binary;  int decimal = 0;  int length, power;  System.out.print("Enter a binary number: ");  binary = sc.nextLine();    length = binary.length();    // Traverse the binary number from left to right  for (int i = 0; i < length; i++) {  char bit = binary.charAt(i);  if (bit == '1') {  power = length - i - 1;  decimal += Math.pow(2, power);  }  }    System.out.println("Decimal number is: " + decimal);  }  } |

Output:

|  |
| --- |
| Enter a binary number: 1011  Decimal number is: 11 |

Question 16:

Write a Program to check weather a number is Armstrong or not.

Source Code:

|  |
| --- |
| // 16. Program to check whether a number is Armstrong or not  import java.util.Scanner;  class ArmstrongNumber {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num, sum = 0, temp, remainder, n = 0;  System.out.print("Enter a number: ");  num = sc.nextInt();  temp = num;    // Find the number of digits  while (temp != 0) {  temp /= 10;  n++;  }    temp = num;  // Calculate the sum of powers of the digits  while (temp != 0) {  remainder = temp % 10;  sum += Math.pow(remainder, n);  temp /= 10;  }    // Check if the number is an Armstrong number  if (sum == num) {  System.out.println(num + " is an Armstrong number");  } else {  System.out.println(num + " is not an Armstrong number");  }  }  } |

Output:

|  |
| --- |
| Enter a number: 153  153 is an Armstrong number  Enter a number: 123  123 is not an Armstrong number |

Question 17:

Write a Program to calculate Simple Interest using the concept of classes.

Source Code:

|  |
| --- |
| // 17. Program to calculate Simple Interest using the concept of classes  import java.util.Scanner;  class SimpleInterest {  double principal, rate, time, interest;  // Constructor to initialize values  SimpleInterest(double p, double r, double t) {  principal = p;  rate = r;  time = t;  }  // Method to calculate interest  void calculateInterest() {  interest = (principal \* rate \* time) / 100;  }  // Method to display the result  void displayInterest() {  System.out.println("Simple Interest is: " + interest);  }  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  double p, r, t;    System.out.print("Enter Principal amount: ");  p = sc.nextDouble();  System.out.print("Enter Rate of Interest: ");  r = sc.nextDouble();  System.out.print("Enter Time period (in years): ");  t = sc.nextDouble();    SimpleInterest si = new SimpleInterest(p, r, t);  si.calculateInterest();  si.displayInterest();  }  } |

Output:

|  |
| --- |
| Enter Principal amount: 1000  Enter Rate of Interest: 5  Enter Time period (in years): 2  Simple Interest is: 100.0 |

Question 18:

Write a Program to illustrate the concept of Static Member Data and Static Member Function.

Source Code:

|  |
| --- |
| // 18. Program to illustrate the concept of Static Member Data and Static Member Function  class StaticExample {  static int count = 0; // Static variable  // Static method  static void incrementCount() {  count++;  }  public static void main(String[] args) {  System.out.println("Initial count: " + count);  StaticExample.incrementCount();  StaticExample.incrementCount();  System.out.println("Count after incrementing: " + count);  }  } |

Output:

|  |
| --- |
| Initial count: 0  Count after incrementing: 2 |

Question 19:

Write a Program to make a Simple Calculator using the concept of classes.

Source Code:

|  |  |
| --- | --- |
| // 19. Program to make a Simple Calculator using the concept of classes  import java.util.Scanner;  class Calculator {  // Method for Addition  public double add(double a, double b) {  return a + b;  }  // Method for Subtraction  public double subtract(double a, double b) {  return a - b;  }  // Method for Multiplication  public double multiply(double a, double b) {  return a \* b;  }  // Method for Division  public double divide(double a, double b) {  if (b != 0) {  return a / b;  } else {  System.out.println("Error: Division by zero");  return 0;  }  }  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  double num1, num2;  int choice;  Calculator calc = new Calculator(); | // Menu for operations  System.out.println("Simple Calculator");  System.out.println("1. Add");  System.out.println("2. Subtract");  System.out.println("3. Multiply");  System.out.println("4. Divide");  System.out.print("Enter your choice: ");  choice = sc.nextInt();  System.out.print("Enter first number: ");  num1 = sc.nextDouble();  System.out.print("Enter second number: ");  num2 = sc.nextDouble();  switch (choice) {  case 1:  System.out.println("Result: " + calc.add(num1, num2));  break;  case 2:  System.out.println("Result: " + calc.subtract(num1, num2));  break;  case 3:  System.out.println("Result: " + calc.multiply(num1, num2));  break;  case 4:  System.out.println("Result: " + calc.divide(num1, num2));  break;  default:  System.out.println("Invalid choice!");  }  }  } |

Output:

|  |
| --- |
| Simple Calculator  1. Add  2. Subtract  3. Multiply  4. Divide  Enter your choice: 1  Enter first number: 10  Enter second number: 5  Result: 15.0 |

Question 20:

Write a Menu Driven Program to Add, Subtract, Multiply two matrices of order 2X2 using concepts of Object Oriented Programming.

Source Code:

|  |  |
| --- | --- |
| // 20. Menu Driven Program to Add, Subtract, Multiply two matrices of order 2X2  import java.util.Scanner;  class MatrixOperations {  int[][] matrix1 = new int[2][2];  int[][] matrix2 = new int[2][2];  int[][] result = new int[2][2];  void inputMatrix() {  Scanner sc = new Scanner(System.in);  System.out.println("Enter elements for first 2x2 matrix:");  for (int i = 0; i < 2; i++) {  for (int j = 0; j < 2; j++) {  matrix1[i][j] = sc.nextInt();  }  }  System.out.println("Enter elements for second 2x2 matrix:");  for (int i = 0; i < 2; i++) {  for (int j = 0; j < 2; j++) {  matrix2[i][j] = sc.nextInt();  }  }  }  void addMatrices() {  for (int i = 0; i < 2; i++) {  for (int j = 0; j < 2; j++) {  result[i][j] = matrix1[i][j] + matrix2[i][j];  }  }  }  void subtractMatrices() {  for (int i = 0; i < 2; i++) {  for (int j = 0; j < 2; j++) {  result[i][j] = matrix1[i][j] - matrix2[i][j];  }  }  }  void multiplyMatrices() {  for (int i = 0; i < 2; i++) {  for (int j = 0; j < 2; j++) {  result[i][j] = matrix1[i][0] \* matrix2[0][j] + matrix1[i][1] \* matrix2[1][j];  }  }  } | void displayResult() {  for (int i = 0; i < 2; i++) {  for (int j = 0; j < 2; j++) {  System.out.print(result[i][j] + " ");  }  System.out.println();  }  }  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  MatrixOperations obj = new MatrixOperations();  obj.inputMatrix();  System.out.println("1. Add Matrices");  System.out.println("2. Subtract Matrices");  System.out.println("3. Multiply Matrices");  System.out.print("Enter your choice: ");  int choice = sc.nextInt();  switch (choice) {  case 1:  obj.addMatrices();  break;  case 2:  obj.subtractMatrices();  break;  case 3:  obj.multiplyMatrices();  break;  default:  System.out.println("Invalid choice");  return;  }  System.out.println("Result: ");  obj.displayResult();  }  } |

Output:

|  |
| --- |
| Enter elements for first 2x2 matrix:  1  2  3  4  Enter elements for second 2x2 matrix:  5  6  7  8  1. Add Matrices  2. Subtract Matrices  3. Multiply Matrices  Enter your choice: 1  Result:  6 8  10 12 |

Question 21:

Write a Program to swap two integer values, two floating point values, two character values using function overloading.

Source Code:

|  |
| --- |
| // 21. Program to swap two integer values, two floating point values, two character values using function overloading  class Swap {  void swap(int a, int b) {  int temp = a;  a = b;  b = temp;  System.out.println("Swapped integers: a = " + a + ", b = " + b);  }  void swap(float a, float b) {  float temp = a;  a = b;  b = temp;  System.out.println("Swapped floating point values: a = " + a + ", b = " + b);  }  void swap(char a, char b) {  char temp = a;  a = b;  b = temp;  System.out.println("Swapped characters: a = " + a + ", b = " + b);  }  public static void main(String[] args) {  Swap obj = new Swap();  // Swapping integers  obj.swap(10, 20);  // Swapping floating point values  obj.swap(10.5f, 20.5f);  // Swapping characters  obj.swap('A', 'B');  }  } |

Output:

|  |
| --- |
| Swapped integers: a = 20, b = 10  Swapped floating point values: a = 20.5, b = 10.5  Swapped characters: a = B, b = A |

Question 22:

WAP that creates a class Accounts with following details:Instance variables: ac\_no., name, ac\_name, balance .Methods: withdrawal(), deposit(),display().Use constructors to initialize members.

Source Code:

|  |  |
| --- | --- |
| // 22. Program to create a class Accounts with withdrawal, deposit, and display methods  import java.util.Scanner;  class Accounts {  int ac\_no;  String name, ac\_name;  double balance;  // Constructor to initialize values  Accounts(int ac\_no, String name, String ac\_name, double balance) {  this.ac\_no = ac\_no;  this.name = name;  this.ac\_name = ac\_name;  this.balance = balance;  }  // Method to deposit money  void deposit(double amount) {  balance += amount;  System.out.println("Deposited: " + amount);  }  // Method to withdraw money  void withdrawal(double amount) {  if (amount > balance) {  System.out.println("Insufficient balance.");  } else {  balance -= amount;  System.out.println("Withdrawn: " + amount);  }  } | // Method to display account details  void display() {  System.out.println("Account No: " + ac\_no);  System.out.println("Account Holder: " + name);  System.out.println("Account Name: " + ac\_name);  System.out.println("Balance: " + balance);  }  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);    // Creating an account object  Accounts acc = new Accounts(12345, "John Doe", "Saving", 5000.0);    // Display account details  acc.display();    // Perform some transactions  acc.deposit(1500);  acc.withdrawal(2000);    // Display updated account details  acc.display();  }  } |

Output:

|  |
| --- |
| Account No: 12345  Account Holder: John Doe  Account Name: Saving  Balance: 5000.0  Deposited: 1500.0  Withdrawn: 2000.0  Account No: 12345  Account Holder: John Doe  Account Name: Saving  Balance: 4500.0 |

Question 23:

WAP to implement constructor overloading.

Source Code:

|  |
| --- |
| // 23. Program to implement constructor overloading  class ConstructorOverloading {  int a, b;  // Default constructor  ConstructorOverloading() {  a = 0;  b = 0;  System.out.println("Default Constructor: a = " + a + ", b = " + b);  }  // Parameterized constructor  ConstructorOverloading(int a, int b) {  this.a = a;  this.b = b;  System.out.println("Parameterized Constructor: a = " + a + ", b = " + b);  }  public static void main(String[] args) {  // Calling the default constructor  ConstructorOverloading obj1 = new ConstructorOverloading();    // Calling the parameterized constructor  ConstructorOverloading obj2 = new ConstructorOverloading(10, 20);  }  } |

Output:

|  |
| --- |
| Default Constructor: a = 0, b = 0  Parameterized Constructor: a = 10, b = 20 |

Question 24:

WAP to count the no. of objects created in a program.

Source Code:

|  |
| --- |
| // 24. Program to count the number of objects created in a program  class ObjectCount {  static int count = 0; // Static variable to count objects  // Constructor increments count whenever an object is created  ObjectCount() {  count++;  }  public static void main(String[] args) {  // Creating objects  ObjectCount obj1 = new ObjectCount();  ObjectCount obj2 = new ObjectCount();  ObjectCount obj3 = new ObjectCount();  // Displaying the total count of objects created  System.out.println("Number of objects created: " + count);  }  } |

Output:

|  |
| --- |
| Number of objects created: 3 |

Question 25:

WAP to show call by value & call by reference.

Source Code:

|  |
| --- |
| // 25. Program to show Call by Value & Call by Reference  class CallByValueAndReference {    // Call by value method  void callByValue(int a) {  a = a + 10; // Modify the value of 'a' locally  System.out.println("Inside callByValue method: a = " + a);  }  // Call by reference method  void callByReference(int[] arr) {  arr[0] = arr[0] + 10; // Modify the value at arr[0]  System.out.println("Inside callByReference method: arr[0] = " + arr[0]);  }  public static void main(String[] args) {  CallByValueAndReference obj = new CallByValueAndReference();    // Call by Value  int x = 5;  System.out.println("Before callByValue: x = " + x);  obj.callByValue(x); // Passing value of x  System.out.println("After callByValue: x = " + x);    // Call by Reference  int[] arr = {5}; // Array to demonstrate call by reference  System.out.println("Before callByReference: arr[0] = " + arr[0]);  obj.callByReference(arr); // Passing the array reference  System.out.println("After callByReference: arr[0] = " + arr[0]);  }  } |

Output:

|  |
| --- |
| Before callByValue: x = 5  Inside callByValue method: a = 15  After callByValue: x = 5  Before callByReference: arr[0] = 5  Inside callByReference method: arr[0] = 15  After callByReference: arr[0] = 15 |

Question 26:

WAP to implement method over ridding & method overloading.

Source Code:

|  |
| --- |
| // 26. Program to implement method overloading & method overriding  class MethodExample {  // Method Overloading: Same method name, different parameters  void display(int a) {  System.out.println("Integer: " + a);  }  void display(String s) {  System.out.println("String: " + s);  }  // Method Overriding: Overriding the method of the superclass  void show() {  System.out.println("Method in MethodExample class");  }  }  class MethodOverridingChild extends MethodExample {    // Overriding the show method in the child class  @Override  void show() {  System.out.println("Method in MethodOverridingChild class");  }  public static void main(String[] args) {  // Demonstrating Method Overloading  MethodExample obj1 = new MethodExample();  obj1.display(10);  obj1.display("Hello, World!");  // Demonstrating Method Overriding  MethodOverridingChild obj2 = new MethodOverridingChild();  obj2.show(); // Calls the overridden method in the child class  }  } |

Output:

|  |
| --- |
| Integer: 10  String: Hello, World!  Method in MethodOverridingChild class |

Question 27:

WAP that demonstrates all the usages of “super” keyword.

Source Code:

|  |
| --- |
| // 27. Program that demonstrates all usages of the "super" keyword  class Animal {  String name;  // Constructor of the superclass  Animal(String name) {  this.name = name;  System.out.println("Animal Constructor: " + name);  }  // Method in the superclass  void makeSound() {  System.out.println("Animal makes sound");  }  }  class Dog extends Animal {    // Constructor of the subclass calling the superclass constructor using super  Dog(String name) {  super(name); // Calling the superclass constructor  System.out.println("Dog Constructor: " + name);  }  // Overriding the method from the superclass  @Override  void makeSound() {  super.makeSound(); // Calling the superclass method  System.out.println("Dog barks");  }  void displayName() {  System.out.println("Animal Name: " + super.name); // Accessing the superclass variable using super  }  public static void main(String[] args) {  Dog dog = new Dog("Buddy");  dog.makeSound();  dog.displayName();  }  } |

Output:

|  |
| --- |
| Animal Constructor: Buddy  Dog Constructor: Buddy  Animal makes sound  Dog barks  Animal Name: Buddy |

Question 28:

Create a class box having height, width , depth as the instance variables & calculate its volume. Implement constructor overloading in it. Create a subclass named box\_new that has weight as an instance variable. Use super in the box\_new class to initialize members of the base class..

Source Code:

|  |  |
| --- | --- |
| // 28. Program to create a class Box with constructor overloading and subclass Box\_New  class Box {  double height, width, depth;  // Constructor to initialize dimensions  Box(double height, double width, double depth) {  this.height = height;  this.width = width;  this.depth = depth;  }  // Constructor to initialize only height and width, default depth is 1  Box(double height, double width) {  this.height = height;  this.width = width;  this.depth = 1; // Default depth value  }  // Method to calculate volume of the box  double volume() {  return height \* width \* depth;  }  } | class Box\_New extends Box {  double weight;  // Constructor of subclass, calling superclass constructor using 'super'  Box\_New(double height, double width, double depth, double weight) {  super(height, width, depth); // Calling the Box constructor  this.weight = weight;  }  // Method to display details of the box  void display() {  System.out.println("Box dimensions: " + height + " x " + width + " x " + depth);  System.out.println("Volume: " + volume());  System.out.println("Weight: " + weight);  }  }  public class Main {  public static void main(String[] args) {  // Creating an object of the subclass Box\_New  Box\_New box = new Box\_New(10, 5, 3, 15);  box.display();  }  } |

Output:

|  |
| --- |
| Box dimensions: 10.0 x 5.0 x 3.0  Volume: 150.0  Weight: 15.0 |

Question 29:

WAP that implements multilevel inheritance.

Source Code:

|  |
| --- |
| // 29. Program to implement multilevel inheritance  class Animal {  void eat() {  System.out.println("Animal eats food.");  }  }  class Mammal extends Animal {  void sleep() {  System.out.println("Mammal sleeps.");  }  }  class Dog extends Mammal {  void bark() {  System.out.println("Dog barks.");  }  public static void main(String[] args) {  Dog dog = new Dog();  dog.eat(); // Inherited from Animal  dog.sleep(); // Inherited from Mammal  dog.bark(); // Defined in Dog  }  } |

Output:

|  |
| --- |
| Animal eats food.  Mammal sleeps.  Dog barks. |

Question 30:

Identify the type of inheritance and implement it by modelling the Examination Database.

Source Code:

|  |  |
| --- | --- |
| // 30. Implementing single inheritance for the Examination Database  class Person {  String name;  int age;  // Constructor to initialize Person details  Person(String name, int age) {  this.name = name;  this.age = age;  }  void displayPersonInfo() {  System.out.println("Name: " + name);  System.out.println("Age: " + age);  }  }  class Student extends Person {  int rollNo;  // Constructor to initialize Student details  Student(String name, int age, int rollNo) {  super(name, age); // Calling the superclass constructor  this.rollNo = rollNo;  }  void displayStudentInfo() {  displayPersonInfo(); // Calling Person's method  System.out.println("Roll Number: " + rollNo);  }  } | class Exam extends Student {  int marks;  // Constructor to initialize Exam details  Exam(String name, int age, int rollNo, int marks) {  super(name, age, rollNo); // Calling the superclass constructor  this.marks = marks;  }  void displayExamInfo() {  displayStudentInfo(); // Calling Student's method  System.out.println("Marks: " + marks);  }  }  public class Main {  public static void main(String[] args) {  // Creating an Exam object  Exam exam = new Exam("John Doe", 20, 101, 85);  // Displaying the complete examination information  exam.displayExamInfo();  }  } |

Output:

|  |
| --- |
| Name: John Doe  Age: 20  Roll Number: 101  Marks: 85 |

Question 31:

Which type of inheritance is this? Illustrate this inheritance by writing a program assuming your own data members.

Source Code:

|  |  |
| --- | --- |
| // 31. Program illustrating Hierarchical and Multilevel Inheritance  class Vehicle {  void displayVehicle() {  System.out.println("This is a vehicle.");  }  }  class LightMotor extends Vehicle {  void displayLightMotor() {  System.out.println("This is a light motor vehicle.");  }  }  class HeavyMotor extends Vehicle {  void displayHeavyMotor() {  System.out.println("This is a heavy motor vehicle.");  }  }  class GearMotor extends LightMotor {  void displayGearMotor() {  System.out.println("This is a gear motor vehicle.");  }  }  class NonGearMotor extends LightMotor {  void displayNonGearMotor() {  System.out.println("This is a non-gear motor vehicle.");  }  }  class Passenger extends HeavyMotor {  void displayPassenger() {  System.out.println("This is a passenger heavy motor vehicle.");  }  } | class Goods extends HeavyMotor {  void displayGoods() {  System.out.println("This is a goods heavy motor vehicle.");  }  }  public class Main {  public static void main(String[] args) {  GearMotor gearMotor = new GearMotor();  NonGearMotor nonGearMotor = new NonGearMotor();  Passenger passenger = new Passenger();  Goods goods = new Goods();  gearMotor.displayVehicle();  gearMotor.displayLightMotor();  gearMotor.displayGearMotor();  System.out.println();  nonGearMotor.displayVehicle();  nonGearMotor.displayLightMotor();  nonGearMotor.displayNonGearMotor();  System.out.println();  passenger.displayVehicle();  passenger.displayHeavyMotor();  passenger.displayPassenger();  System.out.println();  goods.displayVehicle();  goods.displayHeavyMotor();  goods.displayGoods();  }  } |

Output:

|  |  |
| --- | --- |
| This is a vehicle.  This is a light motor vehicle.  This is a gear motor vehicle.  This is a vehicle.  This is a light motor vehicle.  This is a non-gear motor vehicle. | This is a vehicle.  This is a heavy motor vehicle.  This is a passenger heavy motor vehicle.  This is a vehicle.  This is a heavy motor vehicle.  This is a goods heavy motor vehicle. |

Question 32:

Consider a university where students who participate in the national games or Olympics are given some grace marks. Therefore, the final marks awarded = Exam\_Marks + Sports\_Grace\_Marks. A class diagram representing this scenario is as follow.

Source Code:

|  |  |
| --- | --- |
| // 32. Program illustrating Multiple Inheritance using Interface  interface Sports {  int getGraceMarks();  }  class Student {  String name;  int rollNo;  Student(String name, int rollNo) {  this.name = name;  this.rollNo = rollNo;  }  void displayStudent() {  System.out.println("Name: " + name);  System.out.println("Roll Number: " + rollNo);  }  }  class Exam extends Student {  int examMarks;  Exam(String name, int rollNo, int examMarks) {  super(name, rollNo);  this.examMarks = examMarks;  }  void displayExamMarks() {  System.out.println("Exam Marks: " + examMarks);  }  } | class Result extends Exam implements Sports {  int graceMarks;  Result(String name, int rollNo, int examMarks, int graceMarks) {  super(name, rollNo, examMarks);  this.graceMarks = graceMarks;  }  public int getGraceMarks() {  return graceMarks;  }  void displayResult() {  displayStudent();  displayExamMarks();  System.out.println("Grace Marks from Sports: " + getGraceMarks());  System.out.println("Final Marks: " + (examMarks + getGraceMarks()));  }  }  public class Main {  public static void main(String[] args) {  Result result = new Result("Alice", 101, 80, 10);  result.displayResult();  }  } |

Output:

|  |
| --- |
| Name: Alice  Roll Number: 101  Exam Marks: 80  Grace Marks from Sports: 10  Final Marks: 90 |

Question 33:

WAP to implement Run time polymorphism.

Source Code:

|  |
| --- |
| // 33. Program to implement Run Time Polymorphism  class Animal {  void sound() {  System.out.println("Animal makes a sound");  }  }  class Dog extends Animal {  void sound() {  System.out.println("Dog barks");  }  }  class Cat extends Animal {  void sound() {  System.out.println("Cat meows");  }  }  public class Main {  public static void main(String[] args) {  Animal a;    a = new Dog();  a.sound(); // Calls Dog's sound method    a = new Cat();  a.sound(); // Calls Cat's sound method  }  } |

Output:

|  |
| --- |
| Dog barks  Cat meows |

Question 34:

WAP to implement interface. Create an interface named Shape having area() & perimeter() as its methods. Create three classes circle, rectangle & square that implement this interface.

Source Code:

|  |  |
| --- | --- |
| // 34. Program to implement Interface  interface Shape {  void area();  void perimeter();  }  class Circle implements Shape {  double radius;  Circle(double radius) {  this.radius = radius;  }  public void area() {  System.out.println("Area of Circle: " + (3.14 \* radius \* radius));  }  public void perimeter() {  System.out.println("Perimeter of Circle: " + (2 \* 3.14 \* radius));  }  }  class Rectangle implements Shape {  double length, breadth;  Rectangle(double length, double breadth) {  this.length = length;  this.breadth = breadth;  }  public void area() {  System.out.println("Area of Rectangle: " + (length \* breadth));  }  public void perimeter() {  System.out.println("Perimeter of Rectangle: " + (2 \* (length + breadth)));  }  } | class Square implements Shape {  double side;  Square(double side) {  this.side = side;  }  public void area() {  System.out.println("Area of Square: " + (side \* side));  }  public void perimeter() {  System.out.println("Perimeter of Square: " + (4 \* side));  }  }  public class Main {  public static void main(String[] args) {  Circle c = new Circle(5);  Rectangle r = new Rectangle(4, 6);  Square s = new Square(4);  c.area();  c.perimeter();  System.out.println();  r.area();  r.perimeter();  System.out.println();  s.area();  s.perimeter();  }  } |

Output:

|  |
| --- |
| Area of Circle: 78.5  Perimeter of Circle: 31.400000000000002  Area of Rectangle: 24.0  Perimeter of Rectangle: 20.0  Area of Square: 16.0  Perimeter of Square: 16.0 |

Question 35:

WAP to show multiple inheritance.

Source Code:

|  |
| --- |
| // 35. Program to show Multiple Inheritance using Interface  interface A {  void displayA();  }  interface B {  void displayB();  }  class C implements A, B {  public void displayA() {  System.out.println("Display from Interface A");  }  public void displayB() {  System.out.println("Display from Interface B");  }  }  public class Main {  public static void main(String[] args) {  C obj = new C();  obj.displayA();  obj.displayB();  }  } |

Output:

|  |
| --- |
| Display from Interface A  Display from Interface B |

Question 36:

WAP to implement exception handling. The program should accept two numbers from the user & divide the first no. by the second. It should throw a Arithmetic Exception if an attempt is made to divide the no. by zero. Use try, catch & finally .Implement multi-catch statements also .

Source Code:

|  |
| --- |
| // 36. Program to implement exception handling  import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  try {  System.out.print("Enter the first number: ");  int num1 = sc.nextInt();  System.out.print("Enter the second number: ");  int num2 = sc.nextInt();  int result = num1 / num2;  System.out.println("Result: " + result);  } catch (ArithmeticException e) {  System.out.println("Arithmetic Exception: Division by zero is not allowed.");  } catch (Exception e) {  System.out.println("Exception occurred: " + e.getMessage());  } finally {  System.out.println("Program execution completed.");  }  }  } |

Output:

|  |
| --- |
| Enter the first number: 10  Enter the second number: 2  Result: 5  Program execution completed.  Enter the first number: 10  Enter the second number: 0  Arithmetic Exception: Division by zero is not allowed.  Program execution completed. |

Question 37:

Create a user defined exception named “NoMatchException” that is fired when the number entered by the user is not 10.Use the throws & throw keyword.

Source Code:

|  |
| --- |
| // 37. Program to create a user defined exception NoMatchException  import java.util.Scanner;  class NoMatchException extends Exception {  NoMatchException(String message) {  super(message);  }  }  public class Main {  public static void main(String[] args) throws NoMatchException {  Scanner sc = new Scanner(System.in);  System.out.print("Enter a number: ");  int num = sc.nextInt();  if (num != 10) {  throw new NoMatchException("Number is not 10. Exception Fired!");  } else {  System.out.println("Number matched successfully!");  }  }  } |

Output:

|  |
| --- |
| Enter a number: 10  Number matched successfully!  Enter a number: 5  Exception in thread "main" NoMatchException: Number is not 10. Exception Fired!  at Main.main(Main.java:14) |

Question 38:

WAP that creates three threads which print no.s from 1 to 5, 6 to 10 and 11 to 15 respectively .Set the name & priority of the threads.

Source Code:

|  |
| --- |
| // 38. Program to create three threads which print numbers from 1–5, 6–10 & 11–15 with names and priorities  class NumberPrinter extends Thread {  int startNum, endNum;  NumberPrinter(String name, int priority, int startNum, int endNum) {  super(name);  setPriority(priority);  this.startNum = startNum;  this.endNum = endNum;  }  public void run() {  for(int i = startNum; i <= endNum; i++) {  System.out.println(getName() + ": " + i);  }  }  }  public class Main {  public static void main(String[] args) {  NumberPrinter t1 = new NumberPrinter("Thread-1", Thread.MAX\_PRIORITY, 1, 5);  NumberPrinter t2 = new NumberPrinter("Thread-2", Thread.NORM\_PRIORITY, 6, 10);  NumberPrinter t3 = new NumberPrinter("Thread-3", Thread.MIN\_PRIORITY, 11, 15);  t1.start();  t2.start();  t3.start();  }  } |

Output:

|  |
| --- |
| Thread-1: 1  Thread-1: 2  Thread-1: 3  Thread-1: 4  Thread-1: 5  Thread-2: 6  Thread-2: 7  Thread-2: 8  Thread-2: 9  Thread-2: 10  Thread-3: 11  Thread-3: 12  Thread-3: 13  Thread-3: 14  Thread-3: 15 |

Question 39:

WAP to print even & odd numbers using threads.

Source Code:

|  |
| --- |
| // 39. Program to use synchronization to control access to shared resource  class Counter {  private int count = 0;  // Synchronized method to ensure only one thread can access at a time  synchronized void increment() {  count++;  System.out.println("Count: " + count);  }  }  class MyThread extends Thread {  Counter counter;  MyThread(Counter counter) {  this.counter = counter;  }  public void run() {  for (int i = 0; i < 5; i++) {  counter.increment();  }  }  }  public class Main {  public static void main(String[] args) {  Counter counter = new Counter();    // Creating threads  MyThread t1 = new MyThread(counter);  MyThread t2 = new MyThread(counter);  // Starting threads  t1.start();  t2.start();  }  } |

Output:

|  |
| --- |
| Count: 1  Count: 2  Count: 3  Count: 4  Count: 5  Count: 6  Count: 7  Count: 8  Count: 9  Count: 10 |

Question 40:

WAP that implements the concept of synchronization in threads using both syncronized method and synchronized block.

Source Code:

|  |
| --- |
| // 40. Program to implement synchronization using both synchronized method and synchronized block  class Counter {  private int count = 0;  // Synchronized method  synchronized void increment() {  count++;  System.out.println("Count: " + count);  }  // Synchronized block  void incrementUsingBlock() {  synchronized (this) {  count++;  System.out.println("Count using block: " + count);  }  }  }  class MyThread extends Thread {  Counter counter;  MyThread(Counter counter) {  this.counter = counter;  }  public void run() {  counter.increment(); // Using synchronized method  counter.incrementUsingBlock(); // Using synchronized block  }  }  public class Main {  public static void main(String[] args) {  Counter counter = new Counter();    // Creating threads  MyThread t1 = new MyThread(counter);  MyThread t2 = new MyThread(counter);  // Starting threads  t1.start();  t2.start();  }  } |

Output:

|  |
| --- |
| Count: 1  Count using block: 2  Count: 3  Count using block: 4 |

Question 41:

WAP that demonstrates the use of sleep and join methods in thread. Use minimum three threads.

Source Code:

|  |
| --- |
| // 41. Program to demonstrate the use of sleep and join methods in thread  class MyThread extends Thread {  String name;  MyThread(String name) {  this.name = name;  }  public void run() {  try {  System.out.println(name + " started.");  Thread.sleep(2000); // Sleeping for 2 seconds  System.out.println(name + " finished.");  } catch (InterruptedException e) {  System.out.println(e);  }  }  }  public class Main {  public static void main(String[] args) throws InterruptedException {  MyThread t1 = new MyThread("Thread-1");  MyThread t2 = new MyThread("Thread-2");  MyThread t3 = new MyThread("Thread-3");  t1.start();  t2.start();  t3.start();  // Ensuring t1 finishes before t2 starts  t1.join();  t2.join();  t3.join();  System.out.println("All threads finished.");  }  } |

Output:

|  |
| --- |
| Thread-1 started.  Thread-2 started.  Thread-3 started.  Thread-1 finished.  Thread-2 finished.  Thread-3 finished.  All threads finished. |

Question 42:

WAP to demonstrate the use of equals(), trim() ,length() , substring(), compareTo() of String class.

Source Code:

|  |
| --- |
| // 42. Program to demonstrate equals(), trim(), length(), substring(), compareTo() of String class  import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.print("Enter first string (may include leading/trailing spaces): ");  String s1 = sc.nextLine();  System.out.print("Enter second string: ");  String s2 = sc.nextLine();  System.out.println("equals: " + s1.equals(s2));  System.out.println("trim: '" + s1.trim() + "'");  System.out.println("length: " + s1.length());  if (s1.length() >= 4) {  System.out.println("substring(1,4): " + s1.substring(1, 4));  } else {  System.out.println("substring(1,4): String too short");  }  System.out.println("compareTo: " + s1.compareTo(s2));  }  } |

Output:

|  |
| --- |
| Enter first string (may include leading/trailing spaces): Hello World  Enter second string: Hello World  equals: false  trim: 'Hello World'  length: 13  substring(1,4): Hel  compareTo: -32 |

Question 43:

WAP to implement file handling . The program should copy the content from one file to another.

Source Code:

|  |
| --- |
| // 43. Program to copy content from one file to another using file handling  import java.io.\*;  public class Main {  public static void main(String[] args) {  BufferedReader reader = null;  BufferedWriter writer = null;  try {  reader = new BufferedReader(new FileReader("input.txt"));  writer = new BufferedWriter(new FileWriter("output.txt"));  String line;  while ((line = reader.readLine()) != null) {  writer.write(line);  writer.newLine();  }  System.out.println("File copied successfully.");  } catch (IOException e) {  System.out.println("An error occurred: " + e.getMessage());  } finally {  try {  if (reader != null) reader.close();  if (writer != null) writer.close();  } catch (IOException e) {  System.out.println("Error closing files: " + e.getMessage());  }  }  }  } |

Output:

|  |
| --- |
| File copied successfully. |