*This book is dedicated to all the students of the* ***'Leaf Classes'*** *family, who have been an inspiration for its creation.*

# TABLE OF CONTENTS

# COMPUTER APPLICATIONS (86)

## CLASS X

*There will be **one** written paper of **two hours** duration carrying **100 marks** and Internal Assessment of **100 marks**.*

### THEORY – 100 Marks

**1. Revision of Class IX Syllabus**

(i) Introduction to Object Oriented Programming concepts, (ii) Elementary Concept of Objects and Classes, (iii) Values and Data types, (iv) Operators in Java, (v) Input in Java, (vi) Mathematical Library Methods, (vii) Conditional constructs in Java, (viii) Iterative constructs in Java, (ix) Nested for loops.

**2. Class as the Basis of all Computation**

Objects and Classes

*Objects encapsulate state and behaviour – numerous examples; member variables; attributes or features. Variables define state; member methods; Operations/methods/messages/ methods define behaviour.*

*Classes as abstractions for sets of objects; class as an object factory; primitive data types, composite data types. Variable declarations for both types; difference between the two types. Objects as instances of a class.*
*Consider real life examples for explaining the concept of class and object.*

**3. User - defined Methods**

Need of methods, syntax of methods, forms of methods, method definition, method calling, method overloading, declaration of methods,
*Ways to define a method, ways to invoke the methods – call by value [with programs] and call by reference [only definition with an example], Object creation - invoking the methods with respect to use of multiple methods with different names to implement modular programming, using data members and member methods, Actual parameters and formal parameters, Declaration of methods - static and non-static, method prototype / signature, - Pure and impure methods, - pass by value [with programs] and pass by reference [only definition with an example], Returning values from the methods , use of multiple methods and more than one method with the same name (polymorphism - method overloading).*

**4. Constructors**

Definition of Constructor, characteristics, types of constructors, use of constructors, constructor overloading.
*Default constructor, parameterized constructor, constructor overloading., Difference between constructor and method.*

**5. Library classes**

Introduction to wrapper classes, methods of wrapper class and their usage with respect to numeric and character data types. Autoboxing and Unboxing in wrapper classes.

*Class as a composite type, distinction between primitive data type and composite data type or class types. Class may be considered as a new data type created by the user, that has its own functionality. The distinction between primitive and composite types should be discussed through examples. Show how classes allow user defined types in programs. All primitive types have corresponding class wrappers. Introduce Autoboxing and Unboxing with their definition and simple examples.*

*The following methods are to be covered:*

*int parseInt(String s),*
*long parseLong(String s),*
*float parseFloat(String s),*
*double parseDouble(String s),*
*boolean isDigit(char ch),*
*boolean isLetter(char ch),*
*boolean isLetterOrDigit(char ch),*
*boolean isLowerCase(char ch),*
*boolean isUpperCase(char ch),*
*boolean isWhitespace(char ch),*

*char toLowerCase (char ch)*

*char toUpperCase(char ch)*

**6. Encapsulation**

Access specifiers and its scope and visibility.

*Access specifiers – private, protected and public. Visibility rules for private, protected and public access specifiers. Scope of variables, class variables, instance variables, argument variables, local variables.*

**7. Arrays**

Definition of an array, declaration, initialization and accepting data of single dimensional array, accessing the elements of single dimensional array.

*Arrays and their uses*, sorting *technique - bubble sort; Search techniques – linear search and binary search, Array as a composite type, length statement to find the size of the array (sorting and searching techniques using single dimensional array only).*

**8. String handling**

String class, methods of String class, implementation of String class methods, String array

*The following String class methods are to be covered:*

*String trim ()*

*String toLowerCase()*

*String toUpperCase()*

*int length( )*

*char charAt (int n)*

*int indexOf(char ch)*

*int lastIndexOf(char ch)*

*String concat(String str)*

*boolean equals (String str)*

*boolean equalsIgnoreCase(String str)*

*int compareTo(String str)*

*int compareToIgnoreCase(String str)*

*String replace (char oldChar,char newChar)*

*String substring (int beginIndex)*

*String substring (int beginIndex, int endIndex)*

*boolean startsWith(String str)*

*boolean endsWith(String str)*

*String valueOf(all types)*

*Outputs based on all the above methods, Programs based on the above methods, extracting and modifying characters of a string, alphabetical order of the strings in an array [Bubble sort technique], searching for a string in a string array using linear search technique. SIMPLE Programs based on extraction of characters.*

**NOTE: PROGRAMS BASED ON EXTRACTION OF WORDS FROM A SENTENCE ARE NOT INCLUDED**.

**INTERNAL ASSESSMENT - 100 Marks**

This segment of the syllabus is totally practical oriented. The accent is on acquiring basic programming skills quickly and efficiently.

**Programming Assignments (Class X)**

The students should complete a minimum of 20 laboratory assignments during the whole year to reinforce the concepts studied in class.

**Suggested list of Assignments:**

The laboratory assignments will form the bulk of the course. Good assignments should have problems which require design, implementation and testing. They should also embody one or more concepts that have been discussed in the theory class. A significant proportion of the time has to be spent in the laboratory. Computing can only be learnt by doing.

The teacher-in-charge should maintain a record of all the assignments done by the student throughout the year and give it due credit at the time of cumulative evaluation at the end of the year.

Some sample problems are given below as examples. The problems are of varying levels of difficulty:

(i) User defined methods

(a) Programs depicting the concept of pure, impure, static, non- static methods.

(b) Programs based on overloaded methods.

(c) Programs involving data members, member methods invoking the methods with respect to the object created.

(ii) Constructors

(a) Programs based on different types of constructors mentioned in the scope of the syllabus.

(b) Programs / outputs based on constructor overloading

(iii) Library classes

(a) Outputs based on all the methods mentioned in the scope of the syllabus.

(b) Programs to check whether a given character is an uppercase/ lowercase / digit etc.

(iv) Encapsulation

Questions based on identifying the different variables like local, instance, arguments, private, public, class variable etc.

(v) Arrays

(a) Programs based on accessing the elements of an array.

(b) Programs based on sort techniques mentioned in the scope of the syllabus.

(c) Programs based on search techniques mentioned in the scope of the syllabus.

(vi) String handling

(a) Outputs based on all the string methods mentioned in the scope of the syllabus.

(b) Programs based on extracting the characters from a given string and manipulating the same.

(c) Palindrome string, pig Latin, alphabetical order of characters, etc.

**Important:** This list is indicative only. Teachers and students should use their imagination to create innovative and original assignments.

**EVALUATION**

The teacher-in-charge shall evaluate all the assignments done by the student throughout the year [both written and practical work]. He/she shall ensure that most of the components of the syllabus have been used appropriately in the assignments. Assignments should be with appropriate list of variables and comment statements. The student has to mention the output of the programs.

**Proposed Guidelines for Marking**

The teacher should use the criteria below to judge the internal work done. Basically, four criteria are being suggested: class design, coding and documentation, variable description and execution or output. The actual grading will be done by the teacher based on his/her judgment. However, one possible way: divide the outcome for each criterion into one of 4 groups: excellent, good, fair/acceptable, poor/unacceptable, then use numeric values for each grade and add to get the total.

**Class design:**

Has a suitable class (or classes) been used?
Are all attributes with the right kinds of types present?
Is encapsulation properly done?
Is the interface properly designed

**Coding and documentation:**

Is the coding done properly? (Choice of names, no unconditional jumps, proper organization of conditions, proper choice of loops, error handling, code layout) Is the documentation complete and readable? (class documentation, variable documentation, method documentation, constraints, known bugs - if any).

**Variable description:**

**Format for variable description:**

| Name of the Variable | Data Type | Purpose/description |
|---|---|---|
| | | |
| | | |

**Execution or Output:**

Does the program run on all sample input correctly?

**Evaluation of practical work will be done as follows:**

| Subject Teacher (Internal Examiner) | | | | 50 marks |
|---|---|---|---|---|
| External Examiner | | | | 50 marks |
| **Criteria (Total-50 marks)** | **Class design (10 marks)** | **Variable description (10 marks)** | **Coding and Documentation (10 marks)** | **Execution OR Output (20 marks)** |
| **Excellent** | **10** | **10** | **10** | **20** |
| **Good** | **8** | **8** | **8** | **16** |
| **Fair** | **6** | **6** | **6** | **12** |
| **Poor** | **4** | **4** | **4** | **8** |

An External Examiner shall be nominated by the Head of the School and may be a teacher from the faculty, but not teaching the subject in the relevant section/class. For example, A teacher of Computer Science of class VIII may be deputed to be the External Examiner for class X.

The total marks obtained out of 100 are to be sent to the Council by the Head of the school.

The Head of the school will be responsible for the online entry of marks on the Council's CAREERS portal by the due date.

**EQUIPMENT**

There should be enough computer systems to provide for a teaching schedule where at least three-fourth of a time available is used for programming and assignments/practical work. The course shall require at least 4 periods of about 40 minutes duration per week. In one week, out of 4 periods the time should be divided as follows:

- 2 periods – Lecture cum demonstration by the Instructor.
- 2 periods – Assignments/Practical work.

*The hardware and software platforms should be such that students can comfortably develop and run programs on those machines.*

*Since hardware and software evolve and change very rapidly the schools shall need to upgrade them as required. Following are the minimal specifications as of now.*

**RECOMMENDED FACILITIES:**

- A lecture cum demonstration room with a MULTIMEDIA PROJECTOR/ an LCD and Overhead Projector (OHP) attached to the computer.
- A white board with white board markers should be available.
- A fully equipped Computer Laboratory that allows one computer per student.
- The computers should have a minimum of 1 GB RAM and at least a P - IV or Equivalent Processor.
- Good Quality printers.
- A scanner, a web cam/a digital camera (Should be provided if possible).

**SOFTWARE**

*Any suitable Operating System can be used.*

*For teaching fundamental concepts of computing using object-oriented approach, Blue J environment (3.2 or higher version) compatible with JDK (5.0 or higher version) as the base or any other editor or IDE, compatible with JDK (5.0 or higher version) as the base may be used. Ensure that the latest versions of software are used.*

# PATTERN PROGRAMS

- Pattern programs in the ICSE syllabus generally involve the use of nested for loops to generate rectangular and triangular patterns, as well as series involving a single variable (nested while and nested do-while loops are not included).
- You may encounter patterns involving numbers, symbols, or strings in exams.
- Pattern generation is often tested through menu-driven programs or user-defined functions.
- The goal is to write the code that generates the pattern, rather than simply displaying it.
- It is important to pay close attention to the use of System.out.print() and System.out.println() statements, as improper use can lead to inaccurate results.
- Careful analysis and understanding of the given pattern is essential for success.
- With practice and attention to detail, you will be able to tackle any pattern-based question on the exam with ease.
- When working on pattern programs that utilize nested for loops, the Variable Description Table should be written in the following format:

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | i | int | Outer loop to handle the number of rows |
| 2. | j | int | Inner loop to handle the number of columns in each row |

QUESTION 1: Define a class to display the following pattern:

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 1; i <= 5; i++) {
      // Inner loop to handle the number of columns in each row
      for (int j = 1; j <= i; j++) {
        // Print the number on the same line
        System.out.print(i + " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 2: Define a class to display the following pattern:

```
5 4 3 2 1
4 3 2 1
3 2 1
2 1
1
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 5; i >= 1; i--) {
      // Inner loop to handle the number of columns in each row
      for (int j = i; j >= 1; j--) {
        // Print the number on the same line
        System.out.print(j + " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 3: Define a class to display the following pattern:

```
1
2 1
3 2 1
4 3 2 1
5 4 3 2 1
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 1; i <= 5; i++) {
      // Inner loop to handle the number of columns in each row
      for (int j = i; j >= 1; j--) {
        // Print the number on the same line
        System.out.print(j + " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 4: Define a class to display the following pattern:

```
5 4 3 2 1
5 4 3 2
5 4 3
5 4
5
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 1; i <= 5; i++) {
      // Inner loop to handle the number of columns in each row
      for (int j = 5; j >= i; j--) {
        // Print the number on the same line
        System.out.print(j + " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 5: Define a class to display the following pattern:

```
*
* #
* # *
* # * #
* # * # *
```

```java
public class StarPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 1; i <= 5; i++) {
      // Inner loop to handle the number of columns in each row
      for (int j = 1; j <= i; j++) {
        if (j % 2 == 1) {
          // Print "*" if j is odd
          System.out.print("* ");
        } else {
          // Print "#" if j is even
          System.out.print("# ");
        }
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 6: Define a class to display the following pattern:

```
1 3 5 7 9
1 3 5 7
1 3 5
1 3
1
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 9; i >= 1; i-=2) {
      // Inner loop to handle the number of columns in each row
      for (int j = 1; j <= i; j += 2) {
        // Print the number on the same line
        System.out.print(j + " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 7: Define a class to display the following pattern:

```
5
4 4
3 3 3
2 2 2 2
1   1   1   1
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 5; i >= 1; i--) {
      // Inner loop to handle the number of columns in each row
      for (int j = 5; j >= i ; j--) {
        // Print the number on the same line
        System.out.print(i + " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 8: Define a class to display the following pattern:

```
9
7 9
5 7 9
3 5 7 9
1 3 5 7 9
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 9; i >= 1; i-=2) {
      // Inner loop to handle the number of columns in each row
      for (int j = i; j <=9; j-=2) {
        // Print the numbers in decreasing order
        System.out.print(j+ " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}

```

QUESTION 9: Define a class to display the following pattern:

```
1
10
101
1010
```

```java
public class NumberPattern {
  public static void main() {
    // Outer loop to handle the number of rows
    for (int i = 1; i <= 4; i++) {
      // Inner loop to handle the number of columns in each row
      for (int j = 1; j <= i; j++) {
        if (j % 2 == 1) {
          // Print "1" if j is odd
          System.out.print("1");
        } else {
          // Print "0" if j is even
          System.out.print("0");
        }
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 10: Define a class to display the following pattern:

```
1
3 1
5 3 1
7 5 3 1
9 7 5 3 1
```

```java
public class NumberPattern
 {
  public static void main()
{
    // Outer loop to handle the number of rows
    for (int i = 1;  i<= 9; i+=2)
    {
      // Inner loop to handle the number of columns in each row
      for (int j = i; j >= 1; j-=2)
      {
        // Print the numbers in decreasing order
        System.out.print(j + " ");
      }
      // Move to the next line after inner loop completes
      System.out.println();
    }
  }
}
```

QUESTION 11: Define a class to display the following pattern:

```
1
00
111
0000
11111
```

```java
public class Pattern {
    public static void main() {
        for (int i = 1; i <= 5; i++) {
            // Print i 1's if i is odd, otherwise print i 0's
            if (i % 2 == 1) {
                for (int j = 1; j <= i; j++) {
                    System.out.print(1);
                }
            } else {
                for (int j = 1; j <= i; j++) {
                    System.out.print(0);
                }
            }

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

QUESTION 12(a): Define a class to display the following pattern:

```
B L U E J
B L U E
B L U
B L
B
```

```java
public class ExtractString {
    public static void main() {
        String s = "BLUEJ";


        for (int i = 4; i >= 0; i--) {
            // Extract substrings and print them
            System.out.println(s.substring(0, i + 1));
        }
    }
}
```

QUESTION 12(b): Define a class to display the following pattern:

```
B L U E J
L U E J
U E J
E J
J
```

```java
public class ExtractString {
    public static void main() {
        String s = "BLUEJ";

        for (int i = 0; i < 5; i++) {
            // Extract substrings and print them
            System.out.println(s.substring(i, 5));
        }
    }
}
```

QUESTION 12(c): Define a class to display the following pattern:

```
J
E E
U U U
L L L L
B B B B B
```

```java
public class ExtractString {
    public static void main() {
        String s = "BLUEJ";

        for (int i = 4; i >= 0; i--) {
            // Extract substrings and print them
            for (int j = 4; j >= i; j--) {
                System.out.print(s.substring(i,i+1) + " ");
            }
            System.out.println();
        }
    }
}
```

QUESTION 13(a): Define a class to display the following pattern:

```
A
B C
D E F
G H I J
K L M N O
```

```java
public class Pattern {
    public static void main() {
        char ch = 'A'; // Start with character A

        for (int i = 1; i <= 5; i++) {
            // Print i characters
            for (int j = 1; j <= i; j++) {
                System.out.print(ch + " ");
                ch++; // Increment character
            }
            System.out.println();
        }
    }
}
```

QUESTION 13(b): Define a class to display the following pattern:

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

```java
public class Pattern {
    public static void main() {
        for (int i = 1; i <= 5; i++) {
            // Print i asterisks
            for (int j = 1; j <= i; j++) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}
```

QUESTION 14: Define a class to generate a triangle or an inverted triangle till n terms based upon the user's choice.

Example 1:
Input: Type 1 for a triangle and Type 2 for an inverted triangle
Enter your choice: 1
Enter the number of terms: 5

Sample Output:

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

Example 2:

Input: Type 1 for a triangle and Type 2 for an inverted triangle
Enter your choice: 2
Enter the number of terms: 6

Sample Output:
```
6 6 6 6 6 6
5 5 5 5 5
4 4 4 4
3 3 3
2 2
1
```

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | choice | int | Choice entered by the user |
| 2. | n | int | Number of terms |
| 3. | i,j | int | Looping Variables |

```java
import java.util.Scanner;

public class Triangle {
    public static void main() {
        Scanner sc = new Scanner(System.in);

        System.out.println("Type 1 for a triangle and");
        System.out.println("Type 2 for an inverted triangle");
        System.out.print("Enter your choice: ");
        int choice = sc.nextInt();

        System.out.print("Enter the number of terms: ");
        int n = sc.nextInt();

        if (choice == 1) {
            // Print a triangle
            for (int i = 1; i <= n; i++) {
                for (int j = 1; j <= i; j++) {
                    System.out.print(i + " ");
                }
                System.out.println();
            }
        } else if (choice == 2) {
            // Print an inverted triangle
            for (int i = n; i >= 1; i--) {
                for (int j = 1; j <= i; j++) {
                    System.out.print(i + " ");
                }
                System.out.println();
            }
        } else {
            System.out.println("Invalid choice.");
        }
    }
}
```

QUESTION 15: Using the switch statement, write a menu driven program for the following:

(i) To print the Floyd's triangle

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

(ii) To display the following pattern

```
I
I C
I C S
I C S E
```

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | choice | int | Choice entered by the user |
| 2. | numRows | int | Number of rows |
| 3. | num | int | counter |
| 4. | i,j | int | Looping Variables |

```java
import java.util.Scanner;

public class MenuDrivenProgram {
  public static void main() {
    Scanner scanner = new Scanner(System.in);

    int choice;

      // Display menu
      System.out.println("1. Print Floyd's triangle");
      System.out.println("2. Display pattern");
      System.out.print("Enter your choice: ");
      choice = scanner.nextInt();

      switch (choice) {
        case 1:
          // Print Floyd's triangle
          int numRows = 5;
          int num = 1;
          for (int i = 1; i <= numRows; i++) {
            for (int j = 1; j <= i; j++) {
              System.out.print(num + " ");
              num++;
            }
            System.out.println();
          }
          break;
        case 2:
          // Display pattern
          String s="ICSE";
          int len=s.length();
          for(int i=0; i<len; i++)
          {
          for(int j=0; j<=i; j++)
          {
          System.out.print(s.charAt(j));
           }
          System.out.println();
           }
           break;
        default:
          System.out.println("Invalid choice. ");
      }

  }
}
```

# NUMBER PROGRAMS

- As per the trend, at least one number program question is always included in the exam paper.
- Number program questions can appear as individual questions, in a user-defined method question, or in a menu-driven question.
- For number program questions, the definition of the type of number (such as Tech number, Niven number, Palindrome number) will be provided in the question, along with an example.
- The ability to extract digits from a number is crucial for success in these types of questions, so it is important to practice this skill along with the types of questions provided in the book.
- With practice and attention to detail, you will be able to tackle any number program question on the exam with ease.

QUESTION 1:

Define a class  to enter two numbers and check whether they are co-prime or not.
[Two numbers are said to be co-prime, if their HCF is 1 (one).]
Sample Input: 14, 15
Sample Output: They are co-prime.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num1, num2 | int | Numbers entered by the user |
| 2. | hcf | int | To store the hcf of the entered numbers |
| 3. | i | int | Looping Variable |

```java
import java.util.*;
 class Coprime
 {
    public static void main()
 {
        // Read in the two numbers
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter first number: ");
        int num1 = scanner.nextInt();
        System.out.print("Enter second number: ");
        int num2 = scanner.nextInt();

        // Find the HCF of the two numbers
        int hcf = 1;
        for (int i = 2; i <= num1 && i <= num2; i++)
        {
            if (num1 % i == 0 && num2 % i == 0)
                hcf = i;
        }

        // Check if the HCF is 1 and print the result
        if (hcf == 1)
            System.out.println("They are co-prime.");
        else
            System.out.println("They are not co-prime.");
 }
}
```

QUESTION 2:

Write a menu driven class to accept a number from the user and check whether it is a Palindrome or a Perfect number.

(a) Palindrome number: (A number is a Palindrome which when read in reverse order is same as in the right order)

Example: 11, 101, 151 etc.

(b) Perfect number: (A number is called Perfect if it is equal to the sum of its factors other than the number itself.)

Example: 6 = 1 + 2 + 3

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num | int | Number entered by the user |
| 2. | choice | int | User's choice |
| 3. | i | int | Looping Variable |
| 4. | rev | int | Reverse of the entered number |
| 5. | sum | int | Sum of the digits of the entered number |

```java
import java.util.Scanner;

class PalPer
{
    public static void main()
 {
        Scanner in = new Scanner(System.in);
        System.out.println("1. Palindrome number");
        System.out.println("2. Perfect number");
        System.out.print("Enter your choice: ");
        int choice = in.nextInt();
        System.out.print("Enter number: ");
        int num = in.nextInt();

        switch (choice)
        {
            case 1:              int N = num;
                                     int rev = 0;

                                  while(N != 0) {
                                   int digit = N % 10;
                                    N /= 10;
                                   rev = rev * 10 + digit;
                              }

            if (rev == num)
                System.out.println(num + " is palindrome");
            else
                System.out.println(num + " is not palindrome");
            break;

            case 2:              int sum = 0;
                                     for (int i = 1; i <= num / 2; i++)
                                     {
                                     if (num % i == 0)
                                     sum += i;
                                     }

                                      if (num == sum)
                                            System.out.println(num + " is a perfect number");
                                    else
                                          System.out.println(num + " is not a perfect number");
                                     break;

            default:          System.out.println("Incorrect Choice");
                               break;
        }   }  }
```

QUESTION 3:

Define a class to enter two numbers and check whether it is a Harshad number or not.
Harshad/ Niven Number : is an integer that is divisible by the sum of its digits.

For Example :-

The number 18 is a Harshad number in base 10, because the sum of the digits 1 and 8 is 9 (1 + 8 = 9), and 18 is divisible by 9 (since 18 % 9 = 0)

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | number | int | Number entered by the user |
| 2. | sum | int | Sum of the digits of the entered number |
| 3. | n | int | Looping Variable |

```java
import java.util.Scanner;

class HarshadNumber
 {
  public static void main()
 {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter a number to check if it is a Harshad number: ");
    int number = scanner.nextInt();

    int sum = 0;
    int n = number;
    while (n > 0)
   {
      sum += n % 10;
      n /= 10;
    }

    if (number % sum == 0)
       System.out.println(number + " is a Harshad number");
    else
      System.out.println(number + " is not a Harshad number");

  }
}
```

QUESTION 4:

Define a class to enter a number and check whether it is an Emirp number or not.

Emirp number : is a number which is prime when read backwards and frontwards.

For Example : 13 is an emirp number since 13 and 31 both are prime numbers.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num | int | Number entered by the user |
| 2. | c | int | To count number of factors of entered number |
| 3. | i | int | Looping Variable |
| 4. | rev | int | Reverse of the entered number |
| 5. | c2 | int | To count the number of factors of the reverse number |

```java
import java.util.Scanner;

class Emirp
{
    public static void main()
 {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter Number: ");
        int num = in.nextInt();
        int c = 0;
        for (int i = 1; i <= num; i++)
         {
            if (num % i == 0)
             {
                c++;
            }
        }

        if (c == 2)
       {
            int t = num;
            int rev = 0;

            while(t != 0)
              {
                int digit = t % 10;
                t /= 10;
                rev = rev * 10 + digit;
              }

            int c2 = 0;
            for (int i = 1; i <= rev; i++)
              {
                if (rev % i == 0)
                    c2++;
              }

            if (c2 == 2)
                System.out.println("Emirp Number");
            else
                System.out.println("Not Emirp Number");
        }
        else
            System.out.println("Not Emirp Number");
    }
}
```

QUESTION 5:

Define a class to enter two numbers and check whether it is a Pronic number or not.

Pronic Number : A pronic number, oblong number, rectangular number or heteromecic number, is a number which is the product of two consecutive integers, that is, n (n + 1).

Example:
6,12
6=2x3
12=3x4

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | number | int | Number entered by the user |
| 2. | isPronic | boolean | to check whether a number is a Pronic number |
| 3. | i | int | Looping Variable |

```java
import java.util.Scanner;

// Class to check whether a number is a Pronic number
class PronicNumber
{
  public static void main()
 {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter a number to check if it is a Pronic number: ");
    int number = scanner.nextInt();

    // Check if the number is a Pronic number
    boolean isPronic = false;
    for (int i = 0; i * (i + 1) <= number; i++)
    {
      if (i * (i + 1) == number)
      {
        isPronic = true;
        break;
      }
    }

    if (isPronic)
      System.out.println(number + " is a Pronic number");
   else
      System.out.println(number + " is not a Pronic number");
    }
  }
}
```

QUESTION 6:

Define a class to enter two numbers and check whether it is a Disarum number or not.

Disarum Number : A number will be called DISARIUM if sum of its digits powered with their respective position is equal to the original number.

For example 135 is a DISARIUM
(Workings $1^1+3^2+5^3 = 1+9+125 = 135$)

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num | int | Number entered by the user |
| 2. | Count | int | To count number of digits |
| 3. | sum | int | To store the sum |
| 4. | d | int | Digits of the number |
| 5. | N | int | Copy of the entered number |

```java
import java.util.Scanner;

class Disarium
{
    public static void main()
 {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter the number: ");
        int num = in.nextInt();
        int N = num;
        int Count = 0;

        while (num != 0)
       {
            num /= 10;
            Count++;
        }

        num = N;
        int sum = 0;

        while (num != 0)
        {
            int d = num % 10;
            sum += Math.pow(d, Count);
            Count--;
            num /= 10;
        }

        if (sum == N)
            System.out.println("Disarium Number");
        else
            System.out.println("Not a Disarium Number");
    }
}
```

QUESTION 7:

Define a class to enter two numbers and check whether it is a Special number or not.

Special Number : A number is said to be a special number when the sum of factorial of its digits is equal to the number itself.

Example :145 is a special number as 1!+4!+5!
=1+24+120
=145

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | number | int | Number entered by the user |
| 2. | factorial | int | Factorial of a digit |
| 3. | sum | int | To store the sum of the factorials |
| 4. | i | int | Looping variable |
| 5. | digit | int | Digit extracted from the number |

```java
import java.util.Scanner;

// Class to check whether a number is a Special number
class SpecialNumber
 {
  public static void main()
 {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter a number to check if it is a Special number: ");
    int number = scanner.nextInt();

    int sum = 0;
    int n = number;
    while (n > 0)
    {
      int digit = n % 10;
      int factorial = 1;
      for (int i = 1; i <= digit; i++)
      {
        factorial *= i;
      }
      sum += factorial;
      n /= 10;
    }

    if (sum == number)
        System.out.println(number + " is a Special number");
  else
      System.out.println(number + " is not a Special number");
  }
}
```

QUESTION 8:

Define a class to enter a number and check whether it is a Duck number or not.

Duck number : A Duck number is a number which has zero present in it, but there should be no zero present in the beginning of the number.

For example : 3210, 7056, 8430709 are all duck numbers whereas 08237, 04309 are not

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | n | int | Number entered by the user |
| 2. | number | int | Entered number copied |
| 3. | ctr | int | Counter to count number of zeroes |
| 4. | a | int | Digit extracted from the entered number |

```java
import java.util.Scanner;

import java.util.*;
// Class to check whether a number is a Duck number
public class DuckNumber {
  public static void main() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter a number to check if it is a Duck number: ");
    int n = scanner.nextInt();

    int number=n;
    int a,ctr=0;
    while (n > 0)
    {
      a = n % 10;
      if (a == 0)
      {
       ctr++;
      }
      n=n/10;

    }

    if (ctr> 0)
    {
      System.out.println(number + " is a Duck number");
    }
    else
    {
      System.out.println(number + " is not a Duck number");
    }
  }
}
```

QUESTION 9:

Define a class  to enter a three digit number and check whether it is an Armstrong Number or not.

Armstrong Number : An Armstrong number of three digits is an integer such that the sum of the cubes of its digits is equal to the number itself.

For example : 371 is an Armstrong number since 3*3*3 + 7*7*7 + 1*1*1 = 371. Another - 153 = 1*1*1 + 5*5*5 + 3*3*3

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | number | int | Number entered by the user |
| 2. | sum_of_cubes | int | Sum of the cubes of the digits |
| 3. | digit | int | Digit extracted from the entered number |
| 4. | n | int | Looping variable |

```java
import java.util.*;
public class ArmstrongNumber
{
  public static void main()
 {
    Scanner in = new Scanner(System.in);
        System.out.print("Enter the Number: ");
        int number = in.nextInt();

    // compute the sum of the cubes of the digits
    int sum_of_cubes = 0;
    int n = number;
    while (n > 0)
    {
      int digit = n % 10;
      sum_of_cubes += digit * digit * digit;
      n /= 10;
    }

    // check if the sum is equal to the number
    if (sum_of_cubes == number)
      System.out.println(number + " is an Armstrong number.");
      else
      System.out.println(number + " is not an Armstrong number.");

  }
}
```

QUESTION 10:

Define a class to enter a number and check whether it is a Composite Number or not.

Composite Number : A number is said to be a composite, if it has one or more then one factor excluding 1 and the number itself.
Example : 4,6,8,9...

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num | int | Number entered by the user |
| 2. | isComposite | boolean | a flag to track if the number is composite or not |
| 3. | i | int | Looping variable |

```java
import java.util.*;
class CompositeNumberChecker
 {
    public static void main()
    {
        // Prompt the user to enter a number
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = scanner.nextInt();

        // Initialize a flag to track if the number is composite or not
        boolean isComposite = false;

        // Check if the number has any factors other than 1 and itself
        for (int i = 2; i < num; i++)
        {
            if (num % i == 0)
            {
                isComposite = true;
                break;
            }
        }

        // Print the result
        if (isComposite)
            System.out.println(num + " is a composite number");
         else
            System.out.println(num + " is not a composite number");

```

QUESTION 11:

Define a class to display all the ‘Buzz Numbers’ between p and q (where p<q). A ‘Buzz Number’ is the number which ends with 7 or is divisible by 7.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | p | int | Lower limit of the range |
| 2. | q | int | Upper Limit of the range |
| 3. | i | int | Looping variable |

```java
import java.util.Scanner;

class MainBuzzNumbers
 {
  public static void main()
  {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter the lower bound: ");
    int p = scanner.nextInt();

    System.out.print("Enter the upper bound: ");
    int q = scanner.nextInt();

      for (int i = p; i <= q; i++)
      {
      if (i % 7 == 0 || i % 10 == 7)
        System.out.println(i);
      }
  }
}
```

QUESTION 12:

Write a program to input a number and check and print whether it is a Dudeney number or not.
A Dudeney number is a positive integer that is a perfect cube such that the sum of its digits is equal to the cube root of the number. Example:

Consider the number 512.
Sum of digits = 5 + 1 + 2 = 8
Cube root of 512 = 8
As Sum of digits = Cube root of Number hence 512 is a Dudeney number.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num | int | Number entered by the user |
| 2. | sum | int | To store the sum of the digits |
| 3. | cbr | int | Cube root of the entered number |
| 4. | copy | int | Looping variable for extracting the digits |

```java
import java.util.Scanner;

public class Dudeney
{
  public static void main()
  {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter a number: ");
    int num = scanner.nextInt();

    int sum = 0;
    int copy = num;
    while (copy > 0)
    {
      sum += copy % 10;
      copy /= 10;
    }

    int cbr = (int) Math.cbrt(num);
    if (sum == cbr && Math.pow(cbr, 3) == num)
        System.out.println(num + " is a Dudeney number");
     else
      System.out.println(num + " is not a Dudeney number");
  }
}
```

QUESTION 13:

Write a program to generate and print all four digits tech numbers.

A tech number has an even number of digits. If the number is split in two equal halves, then the square of sum of these halves is equal to the number itself.

Example:
Consider the number 3025
Square of sum of the halves of 3025 = $(30 + 25)^2$
= $(55)^2$
= 3025 is a tech number.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | i | int | Looping variable |
| 2. | half1 | int | First two digits of the number |
| 3. | half2 | int | Last two digits of the number |
| 4. | square | int | the square of the sum of the halves |

```java
public class TechNumbers
{
  public static void main()
  {
    for (int i = 1000; i <= 9999; i++)
    {
      // Split the number into two halves
      int half1 = i / 100;
      int half2 = i % 100;
      // Calculate the square of the sum of the halves
      int square = (half1 + half2) * (half1 + half2);
      // Check if the square is equal to the number
      if (square == i)
      {
        System.out.println(i + " is a tech number");
      }
    }
  }
}
```

QUESTION 14:

Define a class to input a number and display the new number after reversing the digits of the original number. The program also displays the absolute difference between the original number and the reversed number.

Sample Input: 941
Sample Output:149
Absolute Difference= 792

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num | int | Number entered by the user |
| 2. | reverse | int | To store the reverse of the number |
| 3. | difference | int | the absolute difference between the original and reversed numbers |
| 4. | copy | int | Looping variable for extracting the digits |

```java
import java.util.Scanner;

public class ReverseNumber {
  public static void main() {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter a number: ");
    int num = scanner.nextInt();

    // Reverse the digits of the number
    int reverse = 0;
    int copy = num;
    while (copy > 0)
    {
      reverse = reverse * 10 + copy % 10;
      copy /= 10;
    }

    // Calculate the absolute difference between the original
    // and reversed numbers
    int difference = Math.abs(num - reverse);

    System.out.println("Reversed number: " + reverse);
    System.out.println("Absolute difference: " + difference);
  }
}
```

QUESTION 15:

Define a class to input two numbers. Calculate and find their GCD.

The Greatest Common Divisor (GCD) of two integers is calculated by the continued division method. Divide the larger number by the smaller, the remainder then divides the previous divisor. The process repeats unless the remainder reaches to zero. The last divisor results in GCD.

Sample Input: 45, 20
Sample Output: GCD=5

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num1, num2 | int | Numbers entered by the user |
| 2. | reverse | int | To store the reverse of the number |
| 3. | temp | int | Temporary variable used for swapping |
| 4. | gcd | int | For storing the GCD |
| 5. | remainder | int | For storing the remainder |

```java
public class GCD
{
  public static void main()
  {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the first number: ");
    int num1 = scanner.nextInt();
    System.out.print("Enter the second number: ");
    int num2 = scanner.nextInt();

    // Ensure that num1 is the larger number
    if (num1 < num2)
    {
      int temp = num1;
      num1 = num2;
      num2 = temp;
    }

    // Calculate the GCD using the continued division method
    int gcd = 1;
    while (num2 > 0)
    {
      int remainder = num1 % num2;
      num1 = num2;
      num2 = remainder;
      gcd = num1;
    }

    System.out.println("GCD: " + gcd);
  }
}
```

# ARRAY PROGRAMS

- This topic is important and requires proper preparation in order to perform well in exams.
- Two questions on this topic can be expected in Section B and around 6 mark questions in Section A.
- It is important to practice implementing linear search, binary search, and bubble sort techniques in order to have a good understanding of array programming.
- Pay attention to looping and index numbering when writing programs, as these are important concepts in array programming.
- With proper practice, students will find this topic easy and will be able to confidently attempt any array program question in exams.

QUESTION 1:

Define a class to create a single dimensional array of size 10. Accept 10 integers in the array. Calculate and display the sum of all odd elements present in the array.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | arr[ ] | int array | To store 10 integers |
| 2. | i | int | Looping variable |
| 3. | sum | int | To store the sum of odd elements |

```java
import java.util.Scanner;

class ArraySum {
  public static void main() {
    // Create an array of size 10
    int[] arr = new int[10];

    // Accept 10 integers in the array
    Scanner sc = new Scanner(System.in);
    for (int i = 0; i < 10; i++) {
      System.out.print("Enter a number: ");
      arr[i] = sc.nextInt();
    }

    // Calculate and display the sum of all odd elements in the array
    int sum = 0;
    for (int i = 0; i < 10; i++) {
      if (arr[i] % 2 == 1) {
        sum += arr[i];
      }
    }
    System.out.println("Sum of all odd elements: " + sum);
  }
}
```

QUESTION 2:

Define a class to input 20 numbers in an array. Display the smallest and the largest element in the array. Also display the average.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | numbers[ ] | int array | To store 20 integers |
| 2. | i | int | Looping variable |
| 3. | smallest | int | To store the smallest element |
| 4. | largest | int | To store the largest element |
| 5. | sum | int | To store the sum of odd elements |
| 6. | average | int | To store the average of the array elements |

```java
import java.util.Scanner;

class NumberArray {
  public static void main() {
    int numbers[] = new int[20];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 20; i++) {
      System.out.print("Enter a number: ");
      numbers[i] = scanner.nextInt();
    }

    int smallest = numbers[0];
    int largest = numbers[0];
    int sum = 0;

    for (int i=0;i<20;i++) {
      if (numbers[i] < smallest) {
        smallest = numbers[i];
      }
      if (numbers[i] > largest) {
        largest = numbers[i];
      }
      sum += numbers[i];
    }
    double average = (double) sum / numbers.length;

    System.out.println("Smallest number: " + smallest);
    System.out.println("Largest number: " + largest);
    System.out.println("Average: " + average);
  }
}
```

QUESTION 3:

Define a class to input 20 numbers in an array and a number to search. Check and print whether the number is present in the array or not using Linear Search Technique.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | numbers[ ] | int array | To store 20 integers |
| 2. | i | int | Looping variable |
| 3. | search | int | Integer to be searched |
| 4. | found | boolean | Flag to check whether the element is present or not |

```java
import java.util.Scanner;

 class LinearSearch {
  public static void main() {
    int[] numbers = new int[20];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 20; i++) {
      System.out.print("Enter a number: ");
      numbers[i] = scanner.nextInt();
    }

    System.out.print("Enter a number to search: ");
    int search = scanner.nextInt();

    boolean found = false;
    for (int i = 0; i < 20; i++) {
      if (numbers[i] == search) {
        found = true;
        break;
      }
    }

    if (found) {
      System.out.println(search + " is present in the array.");
    } else {
      System.out.println(search + " is not present in the array.");
    }
  }
}
```

QUESTION 4:

Define a class to input 20 numbers in ascending order in an array and a number to search. Check and print whether the number is present in the array or not using Binary Search Technique.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | numbers[ ] | int array | To store 20 integers |
| 2. | i | int | Looping variable |
| 3. | low | int | Lower limit while searching |
| 4. | high | int | Upper limit while searching |
| 5. | found | boolean | Flag to check whether the element is present or not |

```java
import java.util.Scanner;

class BinarySearch {
  public static void main() {
    int numbers[] = new int[20];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 20; i++) {
      System.out.print("Enter a number: ");
      numbers[i] = scanner.nextInt();
    }

    System.out.print("Enter a number to search: ");
    int search = scanner.nextInt();

    int low = 0;
    int high = numbers.length - 1;
    boolean found = false;

    while (low <= high) {
      int mid = (low + high) / 2;
      if (numbers[mid] == search) {
        found = true;
        break;
      } else if (numbers[mid] < search) {
        low = mid + 1;
      } else {
        high = mid - 1;
      }
    }

    if (found) {
      System.out.println(search + " is present in the array.");
    } else {
      System.out.println(search + " is not present in the array.");
    }
  }
}
```

QUESTION 5:

Define a class to input 10 numbers in an array and sort them in ascending order using Bubble Sort technique. Display the sorted array.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | numbers[ ] | int array | To store 10 integers |
| 2. | i, j | int | Looping variables |
| 3. | temp | int | Temporary variable used for swapping the values |

```java
class BubbleSort {
  public static void main() {
    int[] numbers = new int[10];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 10; i++)
   {
      System.out.print("Enter a number: ");
      numbers[i] = scanner.nextInt();
    }

    for (int i = 0; i < numbers.length-1; i++)
    {
      for (int j = 0; j < numbers.length - i - 1; j++)
      {
        if (numbers[j] > numbers[j + 1])
        {
          int temp = numbers[j];
          numbers[j] = numbers[j + 1];
          numbers[j + 1] = temp;
        }
      }
    }

    System.out.print("Sorted array: ");
    for (int i = 0; i < numbers.length; i++) {
      System.out.print(numbers[i] + " ");
    }
  }
}
```

QUESTION 6:

Define a class to input 10 numbers in an array. Calculate and display the sum of all the multiples of 5 and the product of all negative numbers.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | numbers[ ] | int | To store 10 integers |
| 2. | i | int | Looping variable |
| 3. | sum | int | To store the sum of all the multiples of 5 |
| 4. | product | int | To store the product of all negative numbers |

```java
import java.util.Scanner;

public class Numbers {
  public static void main() {
    int[] numbers = new int[10];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 10; i++) {
      System.out.print("Enter a number: ");
      numbers[i] = scanner.nextInt();
    }

    int sum = 0;
    int product = 1;
    for (int i = 0; i < 10; i++) {
      if (numbers[i] % 5 == 0) {
        sum += numbers[i];
      }
      if (numbers[i] < 0) {
        product *= numbers[i];
      }
    }

    System.out.println("Sum of multiples of 5: " + sum);
    System.out.println("Product of negative numbers: " + product);
  }
}
```

QUESTION 7:

Define a class to input 10 integers into an array. Calculate the sum of odd numbers present in even positions.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | numbers[ ] | int | To store 10 integers |
| 2. | i | int | Looping variable |
| 3. | sum | int | To store the sum of odd numbers present in even positions |

```java
import java.util.Scanner;

class Numbers {
  public static void main() {
    int[] numbers = new int[10];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 10; i++) {
      System.out.print("Enter a number: ");
      numbers[i] = scanner.nextInt();
    }

    int sum = 0;
    for (int i = 0; i < numbers.length; i += 2) {
      if (numbers[i] % 2 == 1) {
        sum += numbers[i];
      }
    }

    System.out.println("Sum of odd numbers in even positions: " + sum);
  }
}
```

QUESTION 8:

Define a class to input n integers in an array and store the odd numbers and even numbers separately in two different arrays.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | n | int | Size of the array |
| 2. | numbers[ ] | int array | To store n integers |
| 3. | i | int | Looping variable |
| 4. | oddNumbers[ ] | int array | To store the odd elements |
| 5. | evenNumbers[ ] | int array | To store the even elements |

```java
import java.util.Scanner;

class Numbers {
  public static void main() {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter the number of integers: ");
    int n = scanner.nextInt();

    int numbers[] = new int[n];
    for (int i = 0; i < n; i++) {
      System.out.print("Enter a number: ");
      numbers[i] = scanner.nextInt();
    }

    int oddNumbers[] = new int[n];
    int evenNumbers[] = new int[n];
    int oddIndex = 0;
    int evenIndex = 0;

    for (int i = 0; i < n; i++) {
      if (numbers[i] % 2 == 0) {
        evenNumbers[evenIndex] = numbers[i];
        evenIndex++;
      } else {
        oddNumbers[oddIndex] = numbers[i];
        oddIndex++;
      }
    }

    System.out.print("Odd numbers: ");
    for (int i = 0; i < oddIndex; i++) {
      System.out.print(oddNumbers[i] + " ");
    }
    System.out.println();

    System.out.print("Even numbers: ");
    for (int i = 0; i < evenIndex; i++) {
      System.out.print(evenNumbers[i] + " ");
    }
    System.out.println();
  }
}
```

QUESTION 9:

Define a class to accept 20 characters in an array. Display the frequency of the vowels present in the array.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | ch[ ] | char | To store 20 characters |
| 2. | i | int | Looping variable |
| 3. | aCount | int | Frequency of vowel a |
| 4. | eCount | int | Frequency of vowel e |
| 5. | iCount | int | Frequency of vowel i |
| 6. | oCount | int | Frequency of vowel o |
| 7. | uCount | int | Frequency of vowel u |

```java
import java.util.Scanner;

class Characters {
  public static void main() {
    char ch[] = new char[20];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 20; i++) {
      System.out.print("Enter a character: ");
      ch[i] = scanner.next().charAt(0);
    }

    int aCount = 0;
    int eCount = 0;
    int iCount = 0;
    int oCount = 0;
    int uCount = 0;

    for (int i = 0; i < 20; i++) {
      if (ch[i] == 'a' || ch[i] == 'A') {
        aCount++;
      } else if (ch[i] == 'e' || ch[i] == 'E') {
        eCount++;
      } else if (ch[i] == 'i' || ch[i] == 'I') {
        iCount++;
      } else if (ch[i] == 'o' ||ch[i] == 'O') {
        oCount++;
      } else if (ch[i] == 'u' || ch[i] == 'U') {
        uCount++;
      }
    }

    System.out.println("Frequency of vowels:");
    System.out.println("A: " + aCount);
    System.out.println("E: " + eCount);
    System.out.println("I: " + iCount);
    System.out.println("O: " + oCount);
    System.out.println("U: " + uCount);
  }
}
```

QUESTION 10:

Create three arrays A, B and C all of size 10. Accept integers in two arrays A and B. Fill all the elements of array C with the sum of numbers present in corresponding elements of array A and B.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | A[ ] | int | To store 10 integers from user |
| 2. | B[ ] | int | To store 10 integers from user |
| 3. | C[ ] | int | To store the sum of corresponding elements of A[ ] and B[ ] |
| 4. | i | int | Looping variable |

```java
import java.util.Scanner;

class Arrays {
  public static void main() {
    int[] A = new int[10];
    int[] B = new int[10];
    int[] C = new int[10];
    Scanner scanner = new Scanner(System.in);

    for (int i = 0; i < 10; i++) {
      System.out.print("Enter a number for array A: ");
      A[i] = scanner.nextInt();
      System.out.print("Enter a number for array B: ");
      B[i] = scanner.nextInt();
    }

    for (int i = 0; i < 10; i++) {
      C[i] = A[i] + B[i];
    }

    System.out.print("Array C: ");
    for (int i = 0; i < 10; i++) {
      System.out.print(C[i] + " ");
    }
  }
}


```

QUESTION 11:

Define a class to store 20 temperatures in °F in a Single Dimensional Array. Convert all the temperatures after converting them into °C in another array. Display the °F and corresponding °C temperatures in tabular form.

(Hint: (c/5) = (f - 32) / 9)

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | temperaturesF[ ] | double | To store the temperatures in °F |
| 2. | temperaturesC[ ] | double | To store the temperatures in °C |
| 3. | i | int | Looping variable |

```java
import java.util.Scanner;

public class Converter
{
    public static void main()
    {
        Scanner sc = new Scanner(System.in);
        // create a Scanner object to read input

       double temperaturesF[] = new double[20];
       // create an array to store the temperatures in °F
       double temperaturesC[] = new double[20];
       // create an array to store the temperatures in °C

        // use a for loop to input the temperatures in °F
        for (int i = 0; i < 20; i++)
        {
            System.out.print("Enter temperature in °F: ");
            temperaturesF[i] = sc.nextDouble();
        }
        // use a for loop to convert the temperatures from °F to °C
        //and store the results in another array
        for (int i = 0; i < 20; i++)
        {
            temperaturesC[i] = (temperaturesF[i] - 32) / (9.0 / 5.0);
        }
        // Display the results in tabular form
        System.out.println("°F\t\t°C");
        for (int i = 0; i < 20; i++) {
            System.out.println(temperaturesF[i] + "\t\t" + temperaturesC[i]);
        }
    }
}
```

QUESTION 12:

The class teacher wants to store the marks obtained in English, Maths and Science of her class having 40 students. Write a program to input marks in English, Science and Maths by using three single dimensional arrays. Calculate and print the following information:

(i) Average marks secured by each student.
(ii) Class average in each subject.
[Hint: Class average is the average marks obtained by 40 students in a particular subject.]

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | englishMarks[ ] | int | To store the marks in English |
| 2. | mathMarks[ ] | int | To store the marks in Math |
| 3. | scienceMarks[ ] | int | To store the marks in Science |
| 4. | i | int | Looping variable |
| 5. | totalEnglishMarks | int | Total of English marks |
| 6. | totalMathMarks | int | Total of Math marks |
| 7. | totalScienceMarks | int | Total of Science marks |

```java
import java.util.Scanner;

public class ClassMarks {

    public static void main()
    {
        Scanner sc = new Scanner(System.in); // create a Scanner object to read input

        int englishMarks[] = new int[40]; // create an array to store the marks in English
        int mathMarks[] = new int[40]; // create an array to store the marks in Math
        int scienceMarks[] = new int[40]; // create an array to store the marks in Science

        // use a for loop to input the marks for 40 students
        for (int i = 0; i < 40; i++)
        {
         System.out.print("Enter marks in English, Math and Science for student "+(i + 1) + ": ");
          englishMarks[i] = sc.nextInt();
          mathMarks[i] = sc.nextInt();
          scienceMarks[i] = sc.nextInt();
        }

        // Calculate and print the average marks secured by each student
        System.out.println("\n Average marks secured by each student: ");
        for (int i = 0; i < 40; i++)
       {
            double average = (englishMarks[i] + mathMarks[i] + scienceMarks[i]) / 3.0;
            System.out.println("Student " + (i + 1) + ": " + average);
        }

        // Calculate and print the class average in each subject
        int totalEnglishMarks = 0;
        int totalMathMarks = 0;
        int totalScienceMarks = 0;
        for (int i = 0; i < 40; i++) {
            totalEnglishMarks += englishMarks[i];
            totalMathMarks += mathMarks[i];
            totalScienceMarks += scienceMarks[i];
        }
        double classAverageEnglish = totalEnglishMarks / 40.0;
        double classAverageMath = totalMathMarks / 40.0;
        double classAverageScience = totalScienceMarks / 40.0;
        System.out.println("\n Class average in each subject: ");
        System.out.println("English: " + classAverageEnglish);
        System.out.println("Math: " + classAverageMath);
        System.out.println("Science: " + classAverageScience);
    }
}
```

QUESTION 13:

Define a class to accept 10 different decimal numbers (double data type) in a Single Dimensional Array A. Truncate the fractional part of each number of the array A and store their integer part in another array B.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | decimalNumbers[ ] | double | To store the decimal numbers |
| 2. | truncatedNumbers[ ] | int | To store the truncated numbers |
| 3. | i | int | Looping variable |

```java
import java.util.Scanner;

public class DecimalTruncator
{
    public static void main()
    {
     Scanner sc = new Scanner(System.in);
     // create a Scanner object to read input
     double decimalNumbers[] = new double[10];
     // create an array to store the decimal numbers
       int truncatedNumbers[] = new int[10];
       // create an array to store the truncated numbers
        // use a for loop to input the decimal numbers
        for (int i = 0; i < 10; i++) {
            System.out.print("Enter decimal number " + (i + 1) + ": ");
            decimalNumbers[i] = sc.nextDouble();
        }
        // use a for loop to truncate the decimal numbers and
        //store the integer part in another array
        for (int i = 0; i < 10; i++) {
            truncatedNumbers[i] = (int) decimalNumbers[i];
        }

        // display the original decimal numbers and the truncated numbers
        System.out.println("Original decimal numbers: ");
        for (int i = 0; i < 10; i++) {
            System.out.print(decimalNumbers[i] + " ");
        }
        System.out.println("\nTruncated numbers: ");
        for (int i = 0; i < 10; i++) {
            System.out.print(truncatedNumbers[i] + " ");
        }
    }
}
```

QUESTION 14:

Define a class to accept 20 alphabets in an array X. Store all lower case letters in array Y and upper case letters in array Z. Display all three arrays.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | alphabets[ ] | char | To store the alphabets |
| 2. | lower[ ] | char | To store the lowercase alphabets |
| 3. | upper[ ] | char | To store the uppercase alphabets |
| 4. | i | int | Looping variable |
| 5. | l | int | index for the lowercase uppercase arrays |
| 6. | u | int | index for the uppercase arrays |

```java
import java.util.Scanner;

public class Alphabets
 {
    public static void main()
    {
        Scanner sc = new Scanner(System.in);
        // create a Scanner object to read input

        char alphabets[] = new char[20];
        // create an array to store the alphabets
        char lower[] = new char[20];
        // create an array to store the lowercase alphabets
        char upper[] = new char[20];
        // create an array to store the uppercase alphabets

        // use a for loop to input the alphabets
        for (int i = 0; i < 20; i++) {
            System.out.print("Enter alphabet " + (i + 1) + ": ");
            alphabets[i] = sc.next().charAt(0);
        }
        // use a for loop to separate the lowercase and uppercase alphabets
        int l = 0, u = 0; // index for the lowercase and uppercase arrays
        for (int i = 0; i < 20; i++) {
            if (Character.isUpperCase(alphabets[i])) {
                upper[u] = alphabets[i];
                u++;
            } else {
                lower[l] = alphabets[i];
                l++;
            }
        }
        // display all three arrays
        System.out.print("Alphabets: ");
        for (int i = 0; i < 20; i++) {
            System.out.print(alphabets[i] + " ");
        }
        System.out.print("\nLowercase Alphabets: ");
        for (int i = 0; i < l; i++) {
            System.out.print(lower[i] + " ");
        }
        System.out.print("\nUppercase Alphabets: ");
        for (int i = 0; i < u; i++) {
            System.out.print(upper[i] + " ");
        }
    }
}
```

QUESTION 15:

Define a class to input 10 city names in a single dimensional array. Arrange these in alphabetical order using Bubble Sort Technique. Display the sorted array.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | cities[ ] | int | To store the city names |
| 2. | temp | String | Temporary variable used for swapping |
| 3. | i | int | Looping variable |

```java
import java.util.Scanner;

public class CitySorter
{
    public static void main()
    {
        Scanner sc = new Scanner(System.in);
        // create a Scanner object to read input
        String cities[] = new String[10];
        // create an array to store the city names
        // use a for loop to input the city names
        for (int i = 0; i < 10; i++)
        {
            System.out.print("Enter city name " + (i + 1) + ": ");
            cities[i] = sc.nextLine();
        }
        // use the bubble sort technique to sort the city names alphabetically
        for (int i = 0; i < cities.length - 1; i++)
        {
            for (int j = 0; j < cities.length - i - 1; j++)
            {
                if (cities[j].compareTo(cities[j + 1]) > 0)
                {
                    String temp = cities[j];
                    cities[j] = cities[j + 1];
                    cities[j + 1] = temp;
                }
            }
        }
        // display sorted array
        System.out.println("\nSorted cities: ");
        for (int i = 0; i < 10; i++)
        {
            System.out.println(cities[i]);
        }
    }
}
```

# STRING PROGRAMS

- It is advised to focus on preparing for word-based String programs from this topic.
- The ability to extract characters from a word or String is essential for success in these types of questions
- The programs provided in the book align with the current syllabus and by practicing and understanding these questions, you will be well prepared to tackle String program questions on the exam
- In order to be successful in String programs, it is important to be familiar with String and Character functions.
- Preparing these functions will also help in answering Section A of the question paper.

QUESTION 1:

Define a class to input a sentence and count the number of words present in it.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | sentence | String | Sentence entered by the user |
| 2. | i | int | Looping variable |
| 3. | wordCount | int | To store the count of words |

```java
import java.util.Scanner;

class WordCounter {
  public static void main() {
    // Create a Scanner object to read input
    Scanner scanner = new Scanner(System.in);

    // Prompt the user to enter a sentence
    System.out.print("Enter a sentence: ");

    // Read the sentence
    String sentence = scanner.nextLine();

    // Initialize a counter to 0
    int wordCount = 0;

    // Iterate through the characters in the sentence
    for (int i = 0; i < sentence.length(); i++) {
      // If the current character is a space, it means
      //we have encountered the end of a word
      if (sentence.charAt(i) == ' ') {
        // Increment the word count
        wordCount++;
      }
    }

    // Add 1 to the word count to account for the last
    //word in the sentence (which won't have a space after it)


    wordCount++;

    // Print the number of words
    System.out.println("Number of words: " + wordCount);
  }
}
```

QUESTION 2:

Define a class to input a string from the user and convert all the upper case characters into lower case and vice versa, but all other characters remain the same.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | String entered by the user |
| 2. | i | int | Looping variable |
| 3. | converted | String | To store the converted string |
| 4. | c | char | For extraction of characters from the string |

```java
import java.util.Scanner;

class CaseConverter {
  public static void main() {
    // Create a Scanner object to read input
    Scanner scanner = new Scanner(System.in);

    // Prompt the user to enter a string
    System.out.print("Enter a string: ");

    // Read the string
    String input = scanner.nextLine();

    // Initialize a new string to hold the converted characters
    String converted = "";

    // Iterate through the characters in the input string
    for (int i = 0; i < input.length(); i++) {
      char c = input.charAt(i);

      // Check if the character is an uppercase letter
      if (c >= 'A' && c <= 'Z') {
        // Convert the character to lowercase
        c = (char)(c + 32);
      }
      // Check if the character is a lowercase letter
      else if (c >= 'a' && c <= 'z') {
        // Convert the character to uppercase
        c = (char)(c - 32);
      }

      // Add the converted character to the output string
      converted += c;
    }

    // Print the converted string
    System.out.println("Converted string: " + converted);
  }
}
```

QUESTION 3:

Define a class to input a word from the user and check whether it is a Palindrome word or not.
(Hint: A string is said to be palindrome if it remains the same on reading from both ends)

METHOD 1

```java
import java.util.Scanner;

public class Palindrome {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter a word: ");
    String word = sc.nextLine();

    // Reverse the word
    String reverse = "";
    for (int i = word.length() - 1; i >= 0; i--) {
      reverse += word.charAt(i);
    }

    // Check if the word is a palindrome
    if (word.equals(reverse)) {
      System.out.println(word + " is a palindrome");
    } else {
      System.out.println(word + " is not a palindrome");
    }
  }
}
```

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | word | String | String entered by the user |
| 2. | i | int | Looping variable |
| 3. | reverse | String | To store the reverse of the entered string |

METHOD 2

```java
import java.util.Scanner;

class PalindromeChecker {
  public static void main() {
    // Create a Scanner object to read input
    Scanner scanner = new Scanner(System.in);

    // Prompt the user to enter a word
    System.out.print("Enter a word: ");

    // Read the word
    String word = scanner.nextLine();

    // Initialize a flag to true
    boolean isPalindrome = true;

    // Iterate through the characters in the word
    for (int i = 0; i < word.length(); i++) {
      // Check if the character at the current index is equal to the
      //character at the opposite end of the word
      if (word.charAt(i) != word.charAt(word.length() - 1 - i)) {
        // Set the flag to false
        isPalindrome = false;
        break;
      }
    }

    // Print the result
    if (isPalindrome) {
      System.out.println(word + " is a palindrome");
    } else {
      System.out.println(word + " is not a palindrome");
    }
  }
}
```

**Variable Description Table:**

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | word | String | String entered by the user |
| 2. | i | int | Looping variable |
| 3. | isPalindrome | boolean | Flag to check whether the string is palindrome or not |
| 4. | c | char | For extraction of characters from the string |

QUESTION 4:

Define a class to input a word and print its piglatin form.
(Hint : A Pig Latin is an encrypted word in English, which is generated by doing the following alterations:

The first vowel occurring in the input word is placed at the start of the new word along with the remaining alphabet of it. The alphabet is present before the first vowel is shifted, at the end of the new word it is followed by “ay”. )

Sample Input: “paris”
Output: arispay

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | word | String | String entered by the user |
| 2. | i | int | Looping variable |
| 3. | vowelIndex | int | To find the first vowel of the string |
| 4. | c | char | character extracted from the string |

```java
import java.util.Scanner;

public class PigLatin {
  public static void main() {
    Scanner sc = new Scanner(System.in);

    // Prompt the user to enter a word
    System.out.print("Enter a word: ");
    String word = sc.nextLine();

    // Find the first vowel in the word
    int vowelIndex = -1;
    for (int i = 0; i < word.length(); i++)
    {
      char c = word.charAt(i);
      if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u'
      || c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U')
      {
        vowelIndex = i;
        break;
      }
    }

    // If there is no vowel in the word, just add "ay" to the end
    if (vowelIndex == -1) {
      System.out.println(word + "ay");
    } else {
      // Otherwise, move the first vowel to the start and add "ay" to the end
      System.out.println(word.substring(vowelIndex) + word.substring(0, vowelIndex) + "ay");
    }
  }
}
```

QUESTION 5:

Define a class to input a non-palindromic word and print the word in palindromic form.
Sample Input: leaf

Output: leaffael

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | Word entered by the user |
| 2. | i | int | Looping variable |
| 3. | palindrome | String | To store the palindrome string |

```java
import java.util.Scanner;

class Palin {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a word:");
    String input = sc.nextLine();

    String palindrome = input;
    for (int i = input.length() - 1; i >= 0; i--) {
      palindrome += input.charAt(i);
    }
    System.out.println("Palindromic word: " + palindrome);
  }
}
```

QUESTION 6:

Define a class to input a sentence and count the number of words, alphabets and digits are there in the sentence.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | Sentence entered by the user |
| 2. | i | int | Looping variable |
| 3. | wordCount | int | to count the number of words in the sentence |
| 4. | alphabetCount | int | to count the number of alphabets in the sentence |
| 5. | digitCount | int | to count the number of digits in the sentence |
| 6. | c | char | To store the extracted character from the sentence |

```java
import java.util.Scanner;

public class Count {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a sentence:");
    String input = sc.nextLine();

    int wordCount = 0; // variable to count the number of words
    //in the sentence
    int alphabetCount = 0; // variable to count the number of
    //alphabets in the sentence
    int digitCount = 0; // variable to count the number of digits
    //in the sentence

    // Iterate through each character in the sentence
    for (int i = 0; i < input.length(); i++) {
      char c = input.charAt(i);
      if (Character.isWhitespace(c)) {
        // If the character is a space, increment the word count
        wordCount++;
      } else if (Character.isLetter(c)) {
        // If the character is a letter, increment the alphabet count
        alphabetCount++;
      } else if (Character.isDigit(c)) {
        // If the character is a digit, increment the digit count
        digitCount++;
      }
    }

    // Add 1 to the word count to account for the last word in the
    //sentence
    wordCount++;

    System.out.println("Number of words: " + wordCount);
    System.out.println("Number of alphabets: " + alphabetCount);
    System.out.println("Number of digits: " + digitCount);
  }
}
```

QUESTION 7:

Define a class to input a word and print the ASCII value of each character of the word.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | Sentence entered by the user |
| 2. | i | int | Looping variable |
| 3. | c | char | To store the extracted character from the sentence |
| 4. | asciiValue | int | To store the ASCII value of the character |

```java
import java.util.Scanner;

public class Ascii {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a word:");
    String input = sc.nextLine();

    // Iterate through each character in the word
    for (int i = 0; i < input.length(); i++) {
      char c = input.charAt(i);
      // Convert the character to its ASCII value
      int asciiValue = (int) c;
      System.out.println("ASCII value of '" + c + "': " + asciiValue);
    }
  }
}
```

QUESTION 8:

Define a class to accept a string in lowercase and change the first letter of each word to uppercase. Display the new string.

Sample Input: hard work is the key to success
Sample Output: Hard Work Is The Key To Success

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | String entered by the user |
| 2. | output | String | to store the modified string |
| 3. | i | int | Looping variable |
| 4. | c | char | To store the extracted character from the string |
| 5. | newWord | boolean | to keep track of whether we are at the beginning of a new word |

```java
import java.util.Scanner;

public class CapitalizeWords {
  public static void main() {
    // Create a Scanner object to read input from the user
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a string in lowercase:");
    String input = sc.nextLine();

    String output = ""; // variable to store the modified string
    boolean newWord = true; // variable to keep track of whether
    //we are at the beginning of a new word

    // Iterate through each character in the input string
    for (int i = 0; i < input.length(); i++) {
      char c = input.charAt(i);
      if (newWord) {
        // If we are at the beginning of a new word, convert the
        //character to uppercase
        c = Character.toUpperCase(c);
        newWord = false; // set newWord to false until we reach a space
      }
      if (c == ' ') {
        // If we reach a space, set newWord to true to indicate
        //that the next character is the beginning of a new word
        newWord = true;
      }
      output += c; // add the character to the output string
    }

    System.out.println("Modified string: " + output);
  }
}
```

QUESTION 9:

Define a class to accept a string. Convert the string to uppercase Count and output the number of double letter sequences that exist in the string.
Sample Input: She was feeding the little rabbit with an apple

Output: 4

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | String entered by the user |
| 2. | count | int | to count the number of double letters |
| 3. | i | int | Looping variable |

```java
import java.util.Scanner;

public class CountDoubleLetters {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a string:");
    String input = sc.nextLine();

    // Convert the string to uppercase
    input = input.toUpperCase();

    int count = 0; // variable to count the number of double letters

    // Iterate through each character in the string
    for (int i = 0; i < input.length() - 1; i++) {
      // Check if the current character is the same as the next character
      if (input.charAt(i) == input.charAt(i + 1)) {
        count++;
      }
    }

    System.out.println("Number of double letters: " + count);
  }
}
```

QUESTION 10:

Define a class to accept a character from the user. Find its ASCII code. Now reverse this ASCII code and convert the reversed ASCII code to the character. Display the entered character and the character of the reversed ASCII code.

Sample Input:
Enter a character:
A
Sample Output:
Original character: A
Character of reversed ASCII code: 8
(Working: ASCII code of A is 65
Reverse of 65 is 56
Character of 56 is 8)

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | String entered by the user |
| 2. | asciiCode | int | to find the ASCII code of the character |
| 3. | asciiCodeString | String | convert the ASCII code to a string |
| 4. | reversedAscii | String | to store the reversed ASCII code |
| 5. | i | int | Looping variable |
| 6. | reversedAsciiInt | int | convert the reversed ASCII code to an int |
| 7. | reversedChar | char | convert the int to a character |

```java
import java.util.Scanner;

public class ReverseAscii {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a character:");
    char input = sc.nextLine().charAt(0);

    int asciiCode = (int) input; // find the ASCII code of
    //the character
    String asciiCodeString = Integer.toString(asciiCode);
    //convert the ASCII code to a string
    String reversedAscii = ""; // variable to store the
    //reversed ASCII code
    // Iterate through the ASCII code string in reverse and add
    //each character to the reversed ASCII string
    for (int i = asciiCodeString.length() - 1; i >= 0; i--) {
      reversedAscii += asciiCodeString.charAt(i);
    }
    int reversedAsciiInt = Integer.parseInt(reversedAscii);
    // convert the reversed ASCII code to an int
    char reversedChar = (char) reversedAsciiInt;
    // convert the int to a character

    System.out.println("Original character: " + input);
    System.out.println("Character of reversed ASCII code: " + reversedChar);
  }
}
```

QUESTION 11:

Define a class to accept the names of 30 students in an array and display all the names that start with “PR”.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | names[ ] | String array | to store the names of the students |
| 2. | i | int | Looping variable |

```java
import java.util.Scanner;

class NamesWithPR {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    String names[] = new String[30];
    // array to store the names of the students

    // Read in the names of the students
    System.out.println("Enter the names of 30 students:");
    for (int i = 0; i < names.length; i++) {
      names[i] = sc.nextLine();
    }

    // Print the names that start with "PR"
    System.out.println("Names starting with PR:");
    for (int i = 0; i < names.length; i++) {
      if (names[i].startsWith("PR")) {
        System.out.println(names[i]);
      }
    }
  }
}
```

QUESTION 12:

Define a class to accept names of 20 books in a string array. Now enter the name of the book and check whether it is present in the given array or not using Linear Search technique. If found display “Book exists” otherwise display “Book not present”.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | books[ ] | String array | to store the names of the books |
| 2. | i | int | Looping variable |
| 3. | found | boolean | flag to indicate if the book was found |

```java
import java.util.Scanner;

class LinearSearchBooks {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    String books[] = new String[20];
    // array to store the names of the books

    // Read in the names of the books
    System.out.println("Enter the names of 20 books:");
    for (int i = 0; i < books.length; i++) {
      books[i] = sc.nextLine();
    }

    System.out.println("Enter the name of the book to search for:");
    String bookToSearch = sc.nextLine();

    // Linear search for the book in the books array
    boolean found = false; // flag to indicate if the book was found
    for (int i = 0; i < books.length; i++) {
      if (books[i].equals(bookToSearch)) {
        found = true;
        break;
      }
    }

    // Print the result of the search
    if (found) {
      System.out.println("Book exists");
    } else {
      System.out.println("Book not present");
    }
  }
}
```

QUESTION 13:

Define a class to input a sentence and count the number of vowels and consonants in it.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | input | String | String entered by the user |
| 2. | vowelCount | int | to count the number of vowels |
| 3. | consonantCount | int | to count the number of consonants |
| 4. | c | char | To store the extracted character |
| 5. | i | int | Looping variable |

```java
import java.util.Scanner;

 class CountVowelsConsonants {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a sentence:");
    String input = sc.nextLine();

    int vowelCount = 0; // variable to count the number of vowels
    int consonantCount = 0; // variable to count the number of consonants

    // Iterate through each character in the sentence
    for (int i = 0; i < input.length(); i++) {
      char c = input.charAt(i);
      // Check if the character is a vowel (a, e, i, o, u)
      if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {
        vowelCount++;
      } else if (Character.isLetter(c)) {
        // If the character is not a vowel but is a letter, it
        //must be a consonant
        consonantCount++;
      }
    }

    System.out.println("Number of vowels: " + vowelCount);
    System.out.println("Number of consonants: " + consonantCount);
  }
}
```

QUESTION 14:

Define a class to initialize the seven Wonders of the World along with their locations in two different arrays. Search for a name of the country input by the user. If found, display the name of the country along with its Wonder, otherwise display “Sorry not found!”.

Seven Wonders:
CHICHEN ITZA, CHRIST THE REDEEMER, TAJ MAHAL, GREAT WALL OF CHINA, MACHU PICCHU, PETRA, COLOSSEUM
Locations:
MEXICO, BRAZIL, INDIA, CHINA, PERU, JORDAN, ITALY
Examples:
Country name: INDIA
Output: TAJ MAHAL
Country name: USA
Output: Sorry Not found!

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | wonders[ ] | String array | to store the names of the wonders |
| 2. | locations[ ] | String array | to store the names of the locations |
| 3. | country | String | the name of a country |
| 4. | i | int | Looping variable |
| 5. | found | boolean | flag to indicate if the country was found |

```java
import java.util.Scanner;

public class WondersSearch {
  public static void main() {
    String wonders[] = {"CHICHEN ITZA",
    "CHRIST THE REDEEMER", "TAJ MAHAL", "GREAT WALL OF CHINA",
    "MACHU PICCHU", "PETRA", "COLOSSEUM"};
    String locations[] = {"MEXICO", "BRAZIL", "INDIA",
    "CHINA", "PERU", "JORDAN", "ITALY"};

    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the name of a country:");
    String country = sc.nextLine();

    // Search for the country in the locations array
    boolean found = false;
    // flag to indicate if thecountry was found
    for (int i = 0; i < locations.length; i++) {
      if (locations[i].equalsIgnoreCase(country)) {
        // If the country is found, print the country
        //name and the corresponding Wonder
        System.out.println(country + ": " + wonders[i]);
        found = true;
        break;
      }
    }

    // If the country was not found, print "Sorry not found!"
    if (!found) {
      System.out.println("Sorry not found!");
    }
  }
}
```

QUESTION 15:

Write a program to input the names of 30 students of class X in one dimensional array. Arrange the names in the alphabetical ascending order using the Bubble sort technique. Display the sorted array of names.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | names[ ] | String array | to store the names of the students |
| 2. | temp | String | to store the name temporary while swapping |
| 3. | country | String | the name of a country |
| 4. | i,j | int | Looping variables |

```java
import java.util.*;
class StudentSorter {
  public static void main() {
    // Create a Scanner object to read input
    Scanner input = new Scanner(System.in);

    // Declare and initialize the array to hold the names of the students
    String names[] = new String[30];

    // Input the names of the students
    System.out.println("Enter the names of the students:");
    for (int i = 0; i < names.length; i++) {
      names[i] = input.nextLine();
    }

    // Use bubble sort to sort the names in alphabetical order
    for (int i = 0; i < names.length - 1; i++) {
      for (int j = 0; j < names.length - i - 1; j++) {
        if (names[j].compareTo(names[j + 1]) > 0) {
          // Swap names[j] and names[j + 1] if names[j]
          // comes after names[j + 1] in alphabetical order
          String temp = names[j];
          names[j] = names[j + 1];
          names[j + 1] = temp;
        }
      }
    }

    // Display the sorted names
    System.out.println("Sorted names:");
    for (int i = 0; i < names.length; i++) {
      System.out.println(names[i]);
    }
  }
}
```

# USER DEFINED FUNCTIONS PROGRAMS

- Questions related to this concept can appear in both Section A and Section B of exam paper.
- This topic covers a range of concepts in programming, including function overloading, number programs, and string programs.
- Function overloading, in particular, is considered an easy question due to its straightforward concept and simple variable description table.
- Carefully read and understand the question before beginning to work on it. Determine if a main function is required for the question. If a main function is necessary, include it in your solution. If a main function is not specified or required, do not include this in your solution.

QUESTION 1:

Design a class to overload a function series() as follows:

(a) void series(int x, int n): to display the sum of the series given below:
$x^1 + x^2 + x^3 + \ldots + x^n$ terms

(b) void series(int p): to display the following series:
0, 7, 26, 63, … p terms.

(c) void series(): to display the sum of the series given below:
1/2 + 1/3 + 1/4 + … + 1/10.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | sum | int / double | Sum of the series / term of the series |
| 2. | i | int | Looping Variable |

```
class Series
 {
  // Method to display the sum of the series
  public void series(int x, int n)
 {
    double sum = 0.0;
    for (int i = 1; i <= n; i++)
   {
      sum += Math.pow(x,i);
    }
    System.out.println("Sum of the series: " + sum);
  }

  // Method to display the series 0, 7, 26, 63, … p terms
  public void series(int p)
  {
    int sum = 0;
    for (int i = 1; i <= p; i++)
   {
      sum += (i * i * i) - 1;
   }
     System.out.print("Sum of the series: "+sum );
    }


  // Method to display the sum of the series 1/2 + 1/3 + 1/4 + … + 1/10
  public void series() {
    double sum = 0.0;
    for (int i = 2; i <= 10; i++)
   {
      sum += (1.0 / i);
    }
    System.out.println("Sum of the series: " + sum);
  }
}
```

QUESTION 2:

Design a class to overload a function check() as follows:

i) void check(String str, char ch) – to find and print the frequency of a character in a string.
Example :Input: Str = “success” ch = ‘s’
Output: number of s present is=3

ii) void check (String s1) – to display only the vowels from string s1 , after converting it to lower case.
Example :Input:S1= “computer”
Output: o u e

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | count | int | Counter |
| 2. | i | int | Looping Variable |
| 3. | ch | char | Extracted character |
| 4. | s1 | String | Entered String |

```java
class Check
{
  // Method to find and print the frequency of a character in a string
  public void check(String str, char ch) {
    int count = 0;
    for (int i = 0; i < str.length(); i++) {
      if (str.charAt(i) == ch) {
        count++;
      }
    }
    System.out.println("Number of '" + ch + "' present: " + count);
  }

  // Method to display only the vowels from a string after converting it to lower case
  public void check(String s1) {
    s1 = s1.toLowerCase();
    for (int i = 0; i < s1.length(); i++) {
      char ch = s1.charAt(i);
      if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
        System.out.print(ch + " ");
      }
    }
    System.out.println();
  }
}
```

QUESTION 3:

Write a program to input a number. Use a function int Armstrong(int n) to accept the number. The function returns 1, if the number is Armstrong, otherwise zero(0).

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num | int | Number entered by the user |
| 2. | cubeSum | int | Sum of cubes of the individual digits |
| 3. | digit | int | Digit of the number |
| 4. | r | int | To store the return value of the function |

```java
import java.util.Scanner;
public class Arms
{
    public int Armstrong(int n)
    {
     int num = n, cubeSum = 0;
        while (num > 0)
        {
            int digit = num % 10;
            cubeSum = cubeSum + (digit * digit * digit);
            num /= 10;
        }
         if (cubeSum == n)
            return 1;
        else
            return 0;
    }
    public static void main()
    {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter Number: ");
        int num = in.nextInt();
        Arms obj = new Arms();
        int r = obj.Armstrong(num);
        if (r == 1)
            System.out.println(num + " is an Armstrong number");
        else
            System.out.println(num + " is not an Armstrong number");
    }

}
```

QUESTION 4:

Write a program using a function called area() to compute area of the following:

(a) Area of circle = (22/7) * r * r
(b) Area of square= side * side
(c) Area of rectangle = length * breadth

Display the menu to display the area as per the user's choice.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | choice | int | User's choice |
| 2. | radius | double | Radius of the circle |
| 3. | side | int | Side of the square |
| 4. | length | double | Length of the rectangle |
| 5. | breadth | double | breadth of the rectangle |

```java
import java.util.*;

 class AreaCalculator {
  public static void main(String[] args) {
    Scanner input = new Scanner(System.in);
    AreaCalculator ob= new AreaCalculator();
    // Display the menu
    System.out.println("1. Area of circle");
    System.out.println("2. Area of square");
    System.out.println("3. Area of rectangle");
    System.out.print("Enter your choice: ");
    int choice = input.nextInt();

    switch (choice) {
      case 1:
        // Calculate the area of a circle
        System.out.print("Enter the radius: ");
        double radius = input.nextDouble();
        System.out.println("Area of circle: " + ob.area(radius));
        break;
      case 2:
        // Calculate the area of a square
        System.out.print("Enter the side length: ");
        int side = input.nextInt();
        System.out.println("Area of square: " + ob.area(side));
        break;
      case 3:
        // Calculate the area of a rectangle
        System.out.print("Enter the length: ");
        double length = input.nextDouble();
        System.out.print("Enter the breadth: ");
        double breadth = input.nextDouble();
        System.out.println("Area of rectangle: " + ob.area(length, breadth));
        break;
      default:
        System.out.println("Invalid choice.");
    }
  }

  // Method to calculate the area of a circle
  double area(double radius) {
    return (22.0 / 7.0) * radius * radius;
  }

  // Method to calculate the area of a square
   int area(int side) {
    return side * side;
  }

  // Method to calculate the area of a rectangle
double area(double length, double breadth) {
    return length * breadth;
  }
}
```

QUESTION 5:

Define a class using a method Palin( ), to check whether a string is a Palindrome or not.

A Palindrome is a string that reads the same from the left to right and vice versa.

Example : MADAM, ARORA, NITIN etc.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | s | String | Entered String |
| 2. | str | String | String in Upper case |
| 3. | len | int | Length of the string |
| 4. | ch | char | Extracted character |
| 5. | rev | String | Reverse of the string |

```java
public class Palindrome

{
    public void Palin(String s) {

        String str = s.toUpperCase();
        int len = str.length();
        String rev="";
        for (int i = 0; i <  len ; i++)
        {
            char ch=str.charAt(i);
            rev=ch+rev;
        }

            if (str.equals(rev))
            System.out.println("It is a palindrome string.");
        else
            System.out.println("It is not a palindrome string.");
        }
  }
```

QUESTION 6:

Write a program in Java to accept a String from the user. Pass the String to a function Display(String str) which displays the consonants present in the String.

Sample Input: computer
Sample Output:
c
m
p
t
r

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | str | String | Entered String |
| 2. | i | int | Looping Variable |
| 3. | c | char | Extracted character |

```java
import java.util.Scanner;

public class Consonant
{
    public static void Display(String str)
    {
        // loop through each character in the string
        for (int i = 0; i < str.length(); i++)
      {
            char c = str.charAt(i);
            if(Character.isLetter(c))
            // check if the character is a consonant
            if (c != 'a' && c != 'e' && c != 'i' && c != 'o' && c != 'u' && c != 'A'
                 && c != 'E' && c != 'I' && c != 'O' && c != 'U')
                System.out.println(c);
        }
    }

    public static void main()
    {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scanner.nextLine();
        Display(str);
    }
}
```

QUESTION 7:

Write a program in Java to accept a String from the user. Pass the String to a function First(String str) which displays the first character of each word.
Sample Input : Computer Applications ICSE
Sample Output:
C
A
I

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | str | String | Entered String |
| 2. | i | int | Looping Variable |

```java
import java.util.Scanner;

public class Display
 {
  public static void main()
  {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a string: ");
    String str = sc.nextLine();
    Display ob=new Display();
    // Call the function First and pass the string as an argument
    ob.First(str);
  }



  // Function to display the first character of each word in the string
  void First(String str)
  {
     str=' '+str;

    for (int i = 0; i < str.length(); i++)
    {
      // If the current character is a space, display the first\
      //character of the next word
     if (str.charAt(i) == ' ')
          System.out.println(str.charAt(i+1));

    }
    }
}
```

QUESTION 8:

Write a program to accept 10 numbers in a Single Dimensional Array. Pass the array to a function Search(int m[], int n) to search the given number ns in the list of array elements. If the number is present, then display the message ‘Number is present’ otherwise, display ‘number is not present’.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | str | String | Entered String |
| 2. | i | int | Looping Variable |
| 3. | found | boolean | Flag to check the presence of number |
| 4. | a[ ] | int | Array entered by the user |
| 5. | num | int | Number to search |

```java
import java.util.Scanner;

public class Search
{
    public void search(int m[], int n) {
        boolean found = false;
        for (int i = 0; i < m.length; i++) {
            if (m[i] == n) {
                found = true;
                break;
            }
        }
        if (found)
            System.out.println("Number is present");
        else
            System.out.println("Number is not present");
    }

    public static void main() {
        Scanner in = new Scanner(System.in);
        int a[] = new int[10];
        System.out.println("Enter 10 numbers");

        for (int i = 0; i < a.length; i++) {
            a[i] = in.nextInt();
        }

        System.out.print("Enter number to search: ");
        int num = in.nextInt();

        Search obj = new Search();
        obj.search(a, num);
    }
}
```

QUESTION 9:

Write a class Area using function overloading that computes the area of a parallelogram, a rhombus and a trapezium.

Formula:
Area of a parallelogram = base * height
Area of a rhombus = (1/2) * d1 * d2 (where, d1 and d2 are the diagonals)
Area of a trapezium = (1/2) * ( a + b) * h
(where a and b are the parallel sides, h is the perpendicular distance between the parallel sides)

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | a | int / double | Area of parallelogram / rhombus |
| 2. | ar | double | Area of a trapezium |

```java
public class Area
{
    public int area(int base, int height)
    {
        int a = base * height;
        return a;
    }

    public double area(double d1, double d2)
    {
        double a = 0.5 * d1 * d2;
        return a;
    }

    public double area(double a, double b, double h)
    {
        double ar = 0.5 * (a + b) * h;
        return ar;
    }
}
```

QUESTION 10:

Define a class with the name Peri using function overloading that computes the perimeter of a square, a rectangle and a circle.

Formula:
Perimeter of a square = 4 * s
Perimeter of a rectangle = 2 * (l + b)
Perimeter of a circle = 2 * (22/7) * r

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | p | int / double | Perimeter of square/rectangle/circle |
| 2. | l,b | double | Length and Breadth of rectangle |
| 3. | s | int | Side of square |
| 4. | r | double | Radius of circle |

```
public class Peri
{
    public int perimeter(int s)
    {
        int p = 4 * s;
        return p;
    }

    public double perimeter(double l, double b)
    {
        double p = 2 * (l + b);
        return p;
    }

    public double perimeter(double r)
    {
        double p =2 * 22.0/7.0 * r;
        return p;

}
}
```

QUESTION 11:

Design a class to overload a function compare( ) as follows:

1. void compare(int, int) – to compare two integers values and print the greater of the two integers.
2. void compare(char, char) – to compare the numeric value of two characters and print with the higher numeric value.
3. void compare(String, String) – to compare the length of the two strings and print the longer of the two.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | a,b | int | Integers to compare |
| 2. | a,b | char | Characters to compare |
| 3. | a,b | String | Strings to compare |
| 4. | l1, l2 | int | Length of the strings |
| 5. | x,y | int | Numeric values of the characters |

```java
public class Comp
{
    void compare(int a, int b)
    {
        if (a > b)
         System.out.println(a);
        else
            System.out.println(b);

    }

    void compare(char a, char b)
    {
        int x = (int)a;
        int y = (int)b;
        if (x > y)
            System.out.println(a);
        else
            System.out.println(b);
    }

    void compare(String a, String b)
    {
        int l1 = a.length();
        int l2 = b.length();
        if (l1 > l2)
            System.out.println(a);
         else
            System.out.println(b);
    }
    }
```

QUESTION 12:

Design a class to overload a function polygon() as follows:

1. void polygon(int n, char ch) – with one integer and one character type argument to draw a filled square of side n using the character stored in ch.
2. void polygon(int x, int y) – with two integer arguments that draws a filled rectangle of length x and breadth y, using the symbol '@'.
3. void polygon() – with no argument that draws a filled triangle shown below:

Example:

Input value of n=2, ch = 'O'

Output:

OO

OO

Input value of x = 2, y = 5

Output:

@@@@@

@@@@@

Output:

*

**

***

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | n | int | Number of rows for square |
| 2. | ch | char | Filling Character for square |
| 3. | i | int | Looping variables |
| 4. | x,y | int | Length and breadth of rectangle |

```java
public class Polygon
{
    public void polygon(int n, char ch)
    {
        for (int i = 1; i <= n; i++)
        {
            for (int j = 1; j <= n; j++)
            {
                System.out.print(ch);
            }
            System.out.println();
        }
    }

    public void polygon(int x, int y)
    {
        for (int i = 1; i <= x; i++)
        {
            for (int j = 1; j <= y; j++)
            {
                System.out.print('@');
            }
            System.out.println();
        }
    }

    public void polygon()
    {
        for (int i = 1; i <= 3; i++)
        {
            for (int j = 1; j <= i; j++)
            {
                System.out.print('*');
            }
            System.out.println();
        }
    }
}
```

QUESTION 13:

Design a class to overload a function volume( ) as follows:

1. double volume(double r) – with radius (r) as an argument, returns the volume of sphere using the formula:
V = (4/3) * (22/7) * r * r * r

2. double volume(double h, double r) – with height(h) and radius(r) as the arguments, returns the volume of a cylinder using the formula:
V = (22/7) * r * r * h

3. double volume(double 1, double b, double h) – with length(l), breadth(b) and height(h) as the arguments, returns the volume of a cuboid using the formula:
V = l*b*h  (length * breadth * height)

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | r | double | Radius of square |
| 2. | h,r | double | Height and radius of cylinder |
| 3. | l,b,h | double | Length, breadth and height of cuboid |

```java
public class Vol
{
    double volume(double r)
    {
        double vol = (4 / 3.0) * (22 / 7.0) * r * r * r;
        return vol;
    }

    double volume(double h, double r)
    {
        double vol = (22 / 7.0) * r * r * h;
        return vol;
    }

    double volume(double l, double b, double h)
    {
        double vol = l * b * h;
        return vol;
    }
}
```

QUESTION 14:

Design a class to overload a function SumSeries() as follows:

(i) void SumSeries(int n, double x) – with one integer argument and one double argument
to find and display the sum of the series given below:
s = (x/1) – (x/2) + (x/3) – (x/4) + (x/5) … to n terms

(ii) void SumSeries() – To find and display the sum of the following series:
s = 1 + (1 X 2) + (1 X 2 X 3) + … + (1 X 2 X 3 X 4 X … 20)

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | n | int | Number of terms |
| 2. | x | double | Numerator of the terms |
| 3. | s | double | Sum of the series |
| 4. | i | int | Looping variable |
| 5. | tm | long | Term of the series |

```java
public class Overload
{
    void SumSeries(int n, double x)
    {
        double s = 0;
        for (int i = 1; i <= n; i++)
        {
            double t = x / i;
            if (i % 2 == 0)
                s -= t;
            else
                s += t;
        }
        System.out.println("Sum = " + s);
    }

    void SumSeries()
    {
        long s = 0, tm = 1;
        for (int i = 1; i <= 20; i++)
        {
            tm *= i;
            s += tm;
        }
        System.out.println("Sum=" + s);
    }
}
```

QUESTION 15:

Design a class to overload a function num_calc() as follows:

a) void num_calc(int num, char ch) with one integer argument and one character argument computes the square of integer argument if choice ch is ‘s’ otherwise finds its cube.

b) void num_calc(int a, int b, char ch) with two integer arguments and one character argument, computes the product of integer arguments if ch is ‘p’ else adds the integers.

c) void num_calc(String s1, String s2) with two string arguments, which prints whether the strings are equal or not.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | result | int | square/ cube of the number |
| 2. | ch | char | Character received in function argument |
| 3. | s1,s2 | String | Strings received in function argument |
| 4. | i | int | Looping variable |
| 5. | ta,b | int | Numbers received in function argument |

```java
public class NumCalc {
  // overload for num_calc(int num, char ch)
  public void num_calc(int num, char ch) {
    if (ch == 's')
    {
      int result = num * num;
      System.out.println("Square of " + num + ": " + result);
    } else
    {
      int result = num * num * num;
      System.out.println("Cube of " + num + ": " + result);
    }
  }
  // overload for num_calc(int a, int b, char ch)
  public void num_calc(int a, int b, char ch)
  {
    if (ch == 'p')
    {
      int result = a * b;
      System.out.println("Product of " + a + " and " + b + ": " + result);
    } else
    {
      int result = a + b;
      System.out.println("Sum of " + a + " and " + b + ": " + result);
    }
  }
  // overload for num_calc(String s1, String s2)
  public void num_calc(String s1, String s2)
  {
    if (s1.equals(s2))
      System.out.println("Strings are equal.");
    else
      System.out.println("Strings are not equal.");
    }
  }
```

# CLASS PROGRAMS

- A question related to this topic is always there in exam paper.
- The question can be asked both with or without the use of constructors, so students should be familiar with both scenarios.
- When working with class programs, only include the “main()” function if it is specifically asked in the question.
- The process of creating a “Variable Description Table” is straightforward when working with class programs as the details of the data members are usually provided in the question.

QUESTION 1:

Define a class Employee with following specifications:
Class name:
Employee
Data members/variables :
name (String type data)
basic, DA, HRA, Total, pf, netsalary (all double type)

Member functions/methods :
(i) Employee(): A constructor to store the name of the employee (name) and basic salary (basic) of your choice.
(ii) void calculation(): to calculate allowances and salaries as per the given criteria :
DA = 45% of basic
HRA= 15% of basic
pf = 8.33% of basic
Total = basic + Da + HRA
netsalary = Total – pf
(iii) void print(): to display name and other data members with messages.

Write a main function in the above class to create an object of the class and print the details of the employee by invoking the methods.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | name | String | Name of the employee |
| 2. | basic | double | Basic Salary of the employee |
| 3. | DA | double | DA calculated of the employee |
| 4. | HRA | double | HRA calculated of the employee |
| 5. | Total | double | Total salary calculated of the employee |
| 6. | pf | double | Provident Fund of the employee |
| 7. | netsalary | double | Net Salary of the employee |

```java
public class Employee {
    String name;
    double basic, Da, HRA, Total, pf, netsalary;

    // Constructor to store the name and basic salary
    Employee() {
        name = "Sheel";
        basic = 20000;
    }

    // Method to calculate allowances and salaries
    void calculation() {
        Da = basic * 0.45;
        HRA = basic * 0.15;
        pf = basic * 0.0833;
        Total = basic + Da + HRA;
        netsalary = Total - pf;
    }

    // Method to display the details of the employee
    void print( ) {
        System.out.println("Name: " + name);
        System.out.println("Basic salary: " + basic);
        System.out.println("DA: " + Da);
        System.out.println("HRA: " + HRA);
        System.out.println("PF: " + pf);
        System.out.println("Total salary: " + Total);
        System.out.println("Net salary: " + netsalary);
    }

    public static void main() {
        // Create an object of the class Employee
        Employee emp = new Employee();

        // Calculate the allowances and salaries
        emp.calculation();

        // Print the details of the employee
        emp.print();
    }
}
```

QUESTION 2:

Define a class to accept two numbers as instance variables. Use the following functions for the given purposes:

Class Name – Calculate
void inputdata() – to input both the values
void calculate() – to find sum and difference
void outputdata() – to print sum and difference of both the numbers
Use a main method to create an object of the class to call the functions.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | num1, num2 | int | Numbers entered by the user |
| 2. | sum | int | Sum of the entered numbers |
| 3. | diff | int | Difference of the entered numbers |

```java
import java.util.*;
public class Calculate {
    int num1, num2;
    int sum;
    int diff;
    // Method to input both values
    public void inputdata() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter first number: ");
        num1 = sc.nextInt();
        System.out.print("Enter second number: ");
        num2 = sc.nextInt();
    }

    // Method to find sum and difference
    public void calculate() {
        sum = num1 + num2;
        diff = num1 - num2;
    }

    // Method to print sum and difference
    public void outputdata() {
        System.out.println("Sum: " + sum);
        System.out.println("Difference: " + diff);
    }

    public static void main() {
        // Create an object of the class Calculate
        Calculate calc = new Calculate();

        // Input both values
        calc.inputdata();

        // Calculate sum and difference
        calc.calculate();

        // Print sum and difference
        calc.outputdata();
    }
}
```

QUESTION 3:

Define a class Triplet with the following specifications:

Data Members – int a, b, c
Member Methods:
void getdata() – to accept three numbers
void findprint() – to check and display whether the numbers are Pythagorean Triplets or not.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | a,b,c | int | Numbers entered by the user |

```java
import java.util.Scanner;
public class Triplet {
    int a, b, c;

    // Method to accept three numbers
    public void getdata() {
        Scanner sc = new Scanner(System.in);
        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();
    }

    // Method to check and display whether the numbers are
    //Pythagorean Triplets or not
    public void findprint() {
        if (a*a + b*b == c*c || a*a + c*c == b*b || b*b + c*c == a*a) {
            System.out.println("The numbers are Pythagorean Triplets.");
        } else {
            System.out.println("The numbers are not Pythagorean Triplets.");
        }
    }
}
```

QUESTION 4:

Define a class Stock for a bookseller to maintain record of books belonging to the various publishers. The specifications are given below:
Class name – Stock
Data Members:
String title – Contains title of the book
String author – Contains author name
String pub – Contains publisher's name
int noc – Number of copies
Member Methods:
void getdata() – To accept title, author, publisher's name and the number of copies.
void purchase(int t, String a, String p, int n) – To check the existence of the book in the stock by comparing titlel, author's and publisher's name. Also check whether noc >n or not. If yes, maintain the balance as noc-n, otherwise display book is not available or stock is under flowing.

Write a program to perform the task given above.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | title | String | title of the book |
| 2. | author | String | author name of the book |
| 3. | pub | String | publisher's name |
| 4. | noc | int | Number of copies |
| 5. | n | int | Number of copies to purchase |

```java
import java.util.*;
public class Stock {
    String title;
    String author;
    String pub;
    int noc;

    // Method to accept title, author, publisher's name and the number of copies
    public void getdata() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter title: ");
        title = sc.nextLine();
        System.out.print("Enter author: ");
        author = sc.nextLine();
        System.out.print("Enter publisher's name: ");
        pub = sc.nextLine();
        System.out.print("Enter number of copies: ");
        noc = sc.nextInt();
    }

    // Method to check the existence of the book in the stock and maintain the balance
    public void purchase(String t, String a, String p, int n) {
        if (title.equalsIgnoreCase(t) &&
            author.equalsIgnoreCase(a) &&
            pub.equalsIgnoreCase(p) && noc > n) {
            noc -= n;
            System.out.println("Book purchased successfully. Remaining copies: " + noc);
        } else {
            System.out.println("Book is not available or stock is under flowing.");
        }
    }
}
```

QUESTION 5:

Define a class Employee having the following description:

| Data Members | Purpose |
|---|---|
| int pan: | To store personal account number |
| String name: | To store name |
| double taxincome : | To store annual taxable income |
| double tax: | To store tax that is calculated |

Member functions
void input() : Store the pan number, name, taxable income
void cal() : Calculate tax on taxable income
void display() : Output details of an employee

Calculate tax based on the given conditions and display the output as per the given format.

| Total Annual Taxable Income | Tax Rate |
|---|---|
| Up to Rs. 2,50,000 | No tax |
| From Rs. 2,50,001 to Rs. 5,00,000 | 10% of the income exceeding Rs. 2,50,000 |
| From Rs, 5,00,001 to Rs. 10,00,000 | Rs. 30,000 + 20% of income exceeding Rs.5,00,000 |
| Above Rs. 10,00,000 | Rs. 50,000 + 30% of income exceeding Rs.10,00,000 |

Output:

| Pan Number | Name | Tax-Income | Tax |
|---|---|---|---|
| ..... | .... | ..... | ... |
| ..... | .... | ..... | ... |

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | pan | int | To store personal account number |
| 2. | name | String | To store name |
| 3. | taxincome | double | To store annual taxable income |
| 4. | tax | double | To store tax that is calculated |

```java
import java.util.*;
class Employee {
    int pan;
    String name;
    double taxincome;
    double tax;

    void input() {
        // Store the pan number, name, taxable income
         Scanner in = new Scanner(System.in);
        System.out.print("Enter Employee name: ");
        name = in.nextLine();
        System.out.print("Enter PAN num: ");
        pan = in.nextInt();
        System.out.print("Enter Taxable Income: ");
       taxincome = in.nextDouble();
          }

    void cal() {
        // Calculate tax on taxable income
        if (taxincome <= 250000) {
            tax = 0;
        } else if (taxincome <= 500000) {
            tax = 0.1 * (taxincome - 250000);
        } else if (taxincome <= 1000000) {
            tax = 30000 + 0.2 * (taxincome - 500000);
        } else {
            tax = 50000 + 0.3 * (taxincome - 1000000);
        }
    }

    void display() {
        // Output details of an employee
        System.out.println("Pan Number: " + pan);
        System.out.println("Name: " + name);
        System.out.println("Tax-Income: " + taxincome);
        System.out.println("Tax: " + tax);
    }
}
```

QUESTION 6:

Define a class Discount having the following description:

| Data Members | Purpose |
|---|---|
| int cost | to store the price of an article |
| String name | to store the customer's name |
| double dc | to store the discount |
| double amt | to store the amount to be paid |

Write a program to compute the discount according to the given conditions and display the output as per the given format.

| Member functions | Purpose |
|---|---|
| void input() | Stores the cost of the article and name of the customer |
| void cal() | Calculates the discount and amount to be paid |
| void display() | Displays the name of the customer, cost, discount and amount to be paid |

| List Price | Rate of discount |
|---|---|
| Up to ₹5,000 | No discount |
| From ₹5,001 to ₹10,000 | 10% on the list price |
| From ₹10,001 to ₹15,000 | 15% on the list price |
| Above ₹15,000 | 20% on the list price |

Output:

```
Name of the customer  Discount    Amount to be paid
.....................  ........    ..................
.....................  ........    ..................
```

```java
import java.util.*;
class Discount {
    int cost;
    String name;
    double dc;
    double amt;

    void input() {
        // Stores the cost of the article and name of the customer
         Scanner in = new Scanner(System.in);
        System.out.print("Enter Customer's name: ");
        name = in.nextLine();
         System.out.print("Enter Cost of the article: ");
        cost = in.nextInt();

            }

    void cal() {
        // Calculates the discount and amount to be paid
        if (cost <= 5000) {
            dc = 0;
        } else if (cost <= 10000) {
            dc = 0.1 * cost;
        } else if (cost <= 15000) {
            dc = 0.15 * cost;
        } else {
            dc = 0.2 * cost;
        }
        amt = cost - dc;
    }

    void display() {
        // Displays the name of the customer, cost, discount and amount to be paid
        System.out.println("Name of the customer: " + name);
        System.out.println("Discount: " + dc);
        System.out.println("Amount to be paid: " + amt);
    }
}
```

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | cost | int | to store the price of an article |
| 2. | name | String | to store the customer's name |
| 3. | dc | double | to store the discount |
| 4. | amt | double | to store the amount to be paid |

QUESTION 7:

Define a class 'Student' described as below:
Data members/instance variables: name,age,m1,m2,m3 (marks in 3 subjects), maximum, average
Member methods:

i. To accept the details of a student.
ii. To compute the average and the maximum out of three marks.
iii. To display the name, age, marks in three subjects, maximum and average. Write a main method to create an object of a class and call the above member methods.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | name | String | To store name of the student |
| 2. | age | int | To store age of the student |
| 3. | m1,m2,m3 | int | To store marks in three subjects |
| 4. | maximum | int | To store maximum marks |
| 5. | average | double | To store average marks |

```java
import java.util.Scanner;
class Student {
    String name;
    int age;
    int m1;
    int m2;
    int m3;
    int maximum;
    double average;

    void accept() {
        // Accept the details of a student
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter name: ");
        name = scanner.nextLine();
        System.out.print("Enter age: ");
        age = scanner.nextInt();
        System.out.print("Enter marks in three subjects: ");
        m1 = scanner.nextInt();
        m2 = scanner.nextInt();
        m3 = scanner.nextInt();
    }

    void compute() {
        // Compute the average and the maximum out of three marks
        maximum = Math.max(Math.max(m1, m2), m3);
        average = (m1 + m2 + m3) / 3.0;
    }

    void display() {
        // Display the name, age, marks in three subjects, maximum and average
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Marks in three subjects: " + m1 + ", " + m2 + ", " + m3);
        System.out.println("Maximum: " + maximum);
        System.out.println("Average: " + average);
    }
    public static void main() {
        Student student = new Student();
        student.accept();
        student.compute();
        student.display();
    }
}
```

QUESTION 8:

Define a class Interest with the following description:

| Data Members | Purpose |
|---|---|
| int p | to store principal (sum) |
| double r | to store rate |
| int t | to store time |
| double interest | to store the interest to be paid |
| double amt | to store the amount to be paid |

| Member functions | Purpose |
|---|---|
| void input() | Stores the principal, rate, time |
| void cal() | Calculates the interest and amount to be paid |
| void display() | Displays the principal, interest and amount to be paid |

Write a program to compute the interest according to the given conditions and display the output.

| Time | Rate of interest |
|---|---|
| For 1 year | 6.5% |
| For 2 years | 7.5% |
| For 3 years | 8.5% |
| For 4 years or more | 9.5% |

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | p | int | To store principal |
| 2. | r | double | To store rate |
| 3. | t | int | To store time |
| 4. | interest | double | To store interest to be paid |
| 5. | amt | double | To store amount to be paid |

```java
import java.util.Scanner;
public class Interest {
  int p; // principal
  double r; // rate
  int t; // time
  double interest; // interest to be paid
  double amt; // amount to be paid

  // stores the principal, rate, and time
  void input() {
    Scanner in = new Scanner(System.in);
        System.out.print("Enter principle amount: ");
        p = in.nextInt();
        System.out.print("Enter time: ");
        t = in.nextInt();

    // set the rate based on the time
    if (t == 1) {
      r = 6.5;
    } else if (t == 2) {
      r = 7.5;
    } else if (t == 3) {
      r = 8.5;
    } else {
      r = 9.5;
    }
  }

  // calculates the interest and amount to be paid
  void cal() {
    interest = (p * r * t) / 100;
    amt = p + interest;
  }

  // displays the principal, interest, and amount to be paid
  void display() {
    System.out.println("Principal: " + p);
    System.out.println("Interest: " + interest);
    System.out.println("Amount to be paid: " + amt);
  }
}
```

QUESTION 9:

Define a class Vowel with the following specifications:

| Data Members | Purpose |
|---|---|
| String s | To store the string |
| int c | To count vowels |

Write a program to compute the interest according to the given conditions and display the output.

| Member Methods | Purpose |
|---|---|
| void getstr() | to accept a string |
| void getvowel() | to count the number of vowels |
| void display() | to print the number of vowels |

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | s | String | String entered by the user |
| 2. | c | int | To store vowel count |
| 3. | ch | char | Extracted character from the string |
| 4. | i | int | Loopin variable |

```java
import java.util.*;
public class Vowel {
  String s; // string
  int c; // vowel count

  // accept a string
  void getstr() {
     Scanner in = new Scanner(System.in);
        System.out.print("Enter the string: ");
        s = in.nextLine();
          }

  // count the number of vowels
  void getvowel() {
    c = 0;
    s=s.toLowerCase();
    for (int i = 0; i < s.length(); i++) {
      char ch = s.charAt(i);
      if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
        c++;
      }
    }
  }

  // print the number of vowels
  void display() {
    System.out.println("Number of vowels: " + c);
  }
}
```

QUESTION 10:

Define a class Characters with the following specifications:

Data Members:

1. String str – To store the string
Member Methods:

1. void input (String st) – to assign st to str
2. void check() – to check and print the following:
(i) number of letters
(ii) number of digits
(iii) number of uppercase characters
(iv) number of lowercase characters
(v) number of special characters

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | str | String | To store the string |
| 2. | letters | int | To store number of letters |
| 3. | digits | int | To store number of digits |
| 4. | uppercase | int | To store number of uppercase letters |
| 5. | lowercase | int | To store number of lowercase letters |
| 6. | special | int | To store number of special letters |
| 7. | i | int | Looping variable |

```java
public class Characters {
  String str; // string

  // assign st to str
  void input(String st) {
    str = st;
  }

  // check and print the number of letters, digits, uppercase characters,
  //lowercase characters, and special characters
  void check() {
    int letters = 0;
    int digits = 0;
    int uppercase = 0;
    int lowercase = 0;
    int special = 0;

    for (int i = 0; i < str.length(); i++) {
      char ch = str.charAt(i);
      if (Character.isLetter(ch)) {
        letters++;
        if (Character.isUpperCase(ch)) {
          uppercase++;
        } else {
          lowercase++;
        }
      } else if (Character.isDigit(ch)) {
        digits++;
      } else {
        special++;
      }
    }

    System.out.println("Number of letters: " + letters);
    System.out.println("Number of digits: " + digits);
    System.out.println("Number of uppercase characters: " + uppercase);
    System.out.println("Number of lowercase characters: " + lowercase);
    System.out.println("Number of special characters: " + special);
  }
}
```

QUESTION 11:

Define a class Parking with the following specifications:

| Data Members | Purpose |
|---|---|
| int vno | To store the vehicle number |
| int hours | To store the number of hours the vehicle is parked in the parking lot |
| double bill | To store the bill amount |

| Member Methods | Purpose |
|---|---|
| void input() | To input the vno and hours |
| void calculate( ) | To compute the parking charge at the rate ₹20 for the first hour or the part thereof and ₹10 for each additional hour or part thereof. |
| void display() | To display the detail |

Write a main method to create an object of the class and call the above methods.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | str | String | To store the string |
| 2. | letters | int | To store number of letters |
| 3. | digits | int | To store number of digits |
| 4. | uppercase | int | To store number of uppercase letters |
| 5. | lowercase | int | To store number of lowercase letters |
| 6. | special | int | To store number of special letters |
| 7. | i | int | Looping variable |

```java
import java.util.*;
public class Parking {
  int vno;
  // vehicle number
  int hours;
  // number of hours the vehicle is parked in the parking lot
  double bill;
  // bill amount

  // input the vno and hours
  void input() {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the vehicle number: ");
    vno = scanner.nextInt();
    System.out.print("Enter the number of hours the vehicle is parked: ");
    hours = scanner.nextInt();
  }

  // compute the parking charge at the rate ₹3 for the first
  //hour or the part thereof and ₹1.50 for each
  //additional hour or part thereof
  void calculate() {
    bill = 20; // base charge for the first hour
    if (hours > 1) {
      bill += (hours - 1) * 10; // additional charge
      //for each additional hour
    }
  }
  // display the detail
  void display() {
    System.out.println("Vehicle number: " + vno);
    System.out.println("Number of hours parked: " + hours);
    System.out.println("Bill amount: ₹" + bill);
  }
  public static void main() {
    Parking parking = new Parking();
    parking.input();
    parking.calculate();
    parking.display();
  }
}
```

QUESTION 12:

Define a class ElectricityBill with the following specifications:

| Data Members | Purpose |
|---|---|
| String n | stores the name of the customer |
| int units | stores the number of units consumed |
| double bill | stores the amount to be paid |

| Member Methods | Purpose |
|---|---|
| void accept() | to input the name of the customer and number of units consumed |
| void calculate() | to calculate the bill as per the tariff table given below |
| void print() | to prints the details in the format given below |

| Number of units | Rate per unit |
|---|---|
| First 100 units | ₹4.00 |
| Next 200 units | ₹5.00 |
| Above 300 units | ₹6.00 |

Tariff Table

A surcharge of 2.5% charged if the number of units consumed is above 300 units.
Output:
Name of the customer: _____
Number of units consumed: _____
Bill amount: _____
Write a main method to create an object of the class and call the above member methods.

```java
import java.util.*;
class ElectricityBill {
  String n;  // stores the name of the customer
  int units;  // stores the number of units consumed
  double bill;  // stores the amount to be paid

  // Accepts the name of the customer and number of units consumed
  void accept() {
    Scanner input = new Scanner(System.in);
    System.out.println("Enter the name of the customer: ");
    n = input.nextLine();
    System.out.println("Enter the number of units consumed: ");
    units = input.nextInt();
  }

  // Calculates the bill as per the tariff table given
  void calculate() {
    if (units <= 100) {
      bill = units * 4.00;
    } else if (units <= 300) {
      bill = (100 * 4.00) + ((units - 100) * 5.00);
    } else {
      bill = (100 * 4.00) + (200 * 5.00) + ((units - 300) * 6.00);
      bill += (bill * 0.025);  // adds surcharge of 2.5% for units above 300
    }
  }

  // Prints the details in the format given
  void print() {
    System.out.println("Name of the customer: " + n);
    System.out.println("Number of units consumed: " + units);
    System.out.println("Bill amount: " + bill);
  }

  public static void main() {
    ElectricityBill eb = new ElectricityBill();
    eb.accept();
    eb.calculate();
    eb.print();
  }
}
```

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | n | String | To store the name of the customer |
| 2. | units | int | To store the number of units consumed |
| 3. | bill | double | To store the amount to be paid |

QUESTION 13:

Design a class Bank with the following specifications:

Data members:

name: to store the name of the depositor

acno: to input the account number

type: to store type of the account

bal: to store the balance amount in the account

Member functions:

initialize( ): to input the data members from the user

depo(int a ) : where a is the amount to be deposited and the variable bal is to be updated.

withdrawal( int a) : where a is the amount to be withdrawn after checking the balance (Minimum balance should be 1000) and the variable bal is to be updated.

print( ): to print all the details.

Write the main method to create the object of the class and call the above method.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | name | String | To store the name of the depositor |
| 2. | acno | int | To store the account number |
| 3. | type | String | To store the type of the account |
| 4. | bal | int | To store the balance amount in the account |

```java
import java.util.Scanner;
class Bank {
  String name;
  int acno;
  String type;
  int bal;

  void initialize() {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter name: ");
    name = sc.nextLine();
    System.out.print("Enter account number: ");
    acno = sc.nextInt();
    sc.nextLine(); // to consume the newline character
    System.out.print("Enter type: ");
    type = sc.nextLine();
  }
  void depo(int a)
  {
    bal += a;
  }
  void withdrawal(int a)
  {
    if (bal - a >= 1000)
      bal -= a;
    else
      System.out.println("Insufficient balance");
  }
  void print()
  {
    System.out.println("Name: " + name);
    System.out.println("Account number: " + acno);
    System.out.println("Type: " + type);
    System.out.println("Balance: " + bal);
  }
  public static void main()
  {
    Bank ob = new Bank();
    ob.initialize();
    ob.depo(1000);
    ob.withdrawal(3000);
    ob.print();
  }
}
```

QUESTION 14:

Define a class called FruitJuice with the following description:

Instance variables/data members:
int code –stores the product code number
String flavour: stores the flavour of the juice.(orange, apple, etc.)
String type: stores the type of packaging (tetra-pack, bottle, etc.)
int size: stores package size (200ml, 400ml, etc.)
int price: stores the price of the product

Member Methods:

void input(): to input and store the product code, flavour, pack type, pack size and product price

void discount() : to reduce the product price by 10%

void display(): to display the product code, flavour, pack type, pack size and product price

Write the main method to create an object of the class and call the above member methods.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | code | int | To store the product code number |
| 2. | flavour | String | To store the flavour of the juice |
| 3. | type | String | To store the type of packaging |
| 4. | size | int | To store the package size |
| 5. | price | int | price of the product |

```java
import java.util.Scanner;
class FruitJuice
{
  int code;
  String flavour;
  String type;
  int size;
  int price;
  void input()
  {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter product code: ");
    code = sc.nextInt();
    sc.nextLine(); // to consume the newline character
    System.out.print("Enter flavour: ");
    flavour = sc.nextLine();
    System.out.print("Enter pack type: ");
    type = sc.nextLine();
    System.out.print("Enter pack size: ");
    size = sc.nextInt();
    System.out.print("Enter price: ");
    price = sc.nextInt();
  }
  void discount()
  {
    price = price -(int) (price * 0.1);
  }
  void display()
  {
    System.out.println("Product code: " + code);
    System.out.println("Flavour: " + flavour);
    System.out.println("Pack type: " + type);
    System.out.println("Pack size: " + size);
    System.out.println("Price: " + price);
  }
  public static void main() {
    FruitJuice fj = new FruitJuice();
    fj.input();
    fj.discount();
    fj.display();
  }
}
```

QUESTION 15:

Define a class named Movie with the following specifications:

Instance variables/data members:

int year: to store the year of release of a movie

String title: to store the title of the movie.

float rating: to store the popularity rating of the movie.

(minimum rating = 0.0 and maximum rating = 5.0)

Member Methods:

void accept():To input and store year, title and rating.

void display():To display the title of a movie and a message based on the rating as per the table below.

| Rating | Message to be displayed |
|---|---|
| 0.0 to 4.0 | Flop |
| 4.1 to 6.4 | Semi-hit |
| 6.5 to 8.5 | Hit |
| 8.6 to 10.0 | Super Hit |

Write a main method to create an object of the class and call the above member methods.

| S.No. | Variable Name | Variable Type | Purpose |
|---|---|---|---|
| 1. | year | int | To store the year of release of a movie |
| 2. | title | String | To store the title of the movie |
| 3. | rating | float | To store the popularity rating of the movie |

```java
import java.util.Scanner;
class Movie
{
  int year;
  String title;
  float rating;

  void accept()
  {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter year: ");
    year = sc.nextInt();
    sc.nextLine(); // to consume the newline character
    System.out.print("Enter title: ");
    title = sc.nextLine();
    System.out.print("Enter rating: ");
    rating = sc.nextFloat();
  }
  void display()
  {
    System.out.println("Title: " + title);
    if (rating >= 0.0 && rating <= 4.0)
      System.out.println("Flop");
     else if (rating > 4.0 && rating <= 6.4)
      System.out.println("Semi-hit");
     else if (rating > 6.4 && rating <= 8.5)
      System.out.println("Hit");
     else if (rating > 8.5 && rating <= 10.0)
      System.out.println("Super hit");
     else
      System.out.println("Invalid rating");
    }
  public static void main()
  {
    Movie m = new Movie();
    m.accept();
    m.display();
  }
}
```