Programming Fundamentals

---

1. What is a program?

A program can be defined as the set of instructions given to the computer.

---

2. What is a compiler?

A compiler is a software or a tanslator which converts the HLL(Source Code) into the MLL(Executable Code) simultaneously.

---

3. What is an interpreter?

A interpreter is a software or a tanslator which converts the HLL(Source Code) into the MLL(Executable Code) line by line.

---

4. Who is the founder of java and what were the initial names of Java?

The credit for the innovation of java goes to the **Green team** of Sun Microsystem led by James Gosling. The initial names that were tried for java were:

- **C++++--**
- Green
- Oak
- Java : it is the name of the cafe

---

5. Why Java is called as the portable programming language?

Java is called as the portable language because java program which is written and compiled in one Operating System can be executed any other operating system as well.

---

6. What is object?

An object is defined as the real world entity which is also referred to as the instance of a class.

---

7. What is a class?

A class is referred to as the blueprint of an object.

---

8. What is Object Orientation?

An Object Orientation is defined as the perspective towards a real world entity.
In Object Orientation we follow 5 important principles:

- The world is a collection of Objects
- Every object is useful and no object is useless
- Every Object is under constant interactions
- Every object belongs to a type and the type in Object Orientation is called as **class**.
- Every Object has something and every Object does something

---

9. What are the major pillars of Object Orientation?

1. **Encapsulation:**
   It refers to the feature of providing the security to the most important component of an object
2. **Inheritance:**
   It refers to the process of representing the data inform of parent child relationship, in other words it is the hierarchey of classes.
3. **Polymorphism:** It represents 1:M relationship.
   Polymorphism is divided into 2 types:
   1. Compile time Polymorphism
   2. Runtime Polymorphism

4. **Abstraction:** It is the process of hiding the implementation in the Parent class and enforcing the Child class to provide the implementation

---

10. What is Encapsulation? What are the rules to be followed while encapsulating a program?

It refers to the feature of providing the security to the most important component of an object.
The rules to be followed while encapsulating the program are:

- Use private members on the properties
- Use Setter & Getters

---

11. Write a program to Encapsulate a class in java.

```
class Tiger{
    private String name;//Have access only within the Tiger Class
    private String color;//Have access only within the Tiger Class
    private int age;//Have access only within the Tiger Class
    private String country;//Have access only within the Tiger Class

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getColor() {
        return color;
    }
    public void setColor(String color) {
        this.color = color;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    public String getCountry() {
        return country;
    }
    public void setCountry(String country) {
        this.country = country;
    }


}
public class Launch1 {
    public static void main(String[] args) {
        Tiger t1 = new Tiger();
        //t1.name="ramu";//Cannot access the private members outside the
class
        t1.setName("ramu");
        t1.setAge(7);
        t1.setColor("orange");
        t1.setCountry("india");
        System.out.println(t1.getName());
        System.out.println(t1.getAge());
        System.out.println(t1.getColor());
        System.out.println(t1.getCountry());
    }

}
```

---

11. What is Inheritance? What are the advantages of using inheritance?

It refers to the process of representing the data inform of parent child relationship, in other words it is the hierarchey of classes.
Advantages of Inheritance is:

- It promotes reusability of code
- Since it reuses the code, it consumes lesser coding time
- It brings profit for the organization

Inheritance can be achieved by making use of **extends** keyword which promotes **is-a** relationship

---

12. What are the different types of methods in inheritance?

In inheritance we have 3 different types of methods:

1. **Inherited Method:** These are the methods that are inherited from the parent class and used as it is in the child class.

2. **Overridden Method:** These are the methods that are inherited from the Parent class but are modified as per the need in the child class.

3. **Specialized Method:** These are the methods that are not inherited from the parentb class but are present uniquely in the child class.

---

13. Write a simple program to show case the usage of inheritance

```
class Parent{
    //Inherited
    void eat() {
        System.out.println("eats food");
    }
    void sleep() {
        System.out.println("parent sleeps at 9pm");
    }
    //Inherited
    void run() {
        System.out.println("runs for 10km");
    }
}

class Child extends Parent{
    //Overriding---Overridden Method
    void sleep() {
        System.out.println("child sleeps at 1am");
    }
    void swims() {
        System.out.println("child swims");
    }
}
public class Launch2 {
    public static void main(String[] args) {
        Child c1 = new Child();
        c1.eat();
        c1.sleep();
        c1.run();
        c1.swims();
    }
}
```

---

14. What are loops in java? What are the different types of loops?

A loop basically is a feature in java to run a sequence of statements continuously untill a specific condition is met.
There are basically 4 types of loops in java:

1. **for loop**
2. **while loop**
3. **do while loop**
4. **for each loop**

---

15. What are conditional statement in java? What are the different types of conditional statements?

A conditional statement is used for executing a set of statement based on a particular condition.
There are basically 4 types of condtional statements in java:

1. **if condition**
2. **else if condition**
3. **else condition**
4. **switch cases**

---

16. What is a data-type? What are its types?

A data types is defined as the type to which a particular data belongs to.
We have 2 types of data-types:

1. **Primitive Data-Types :** byte, short, int, long, char, boolean, float, double, etc.(literals)
2. **Non-Primitive Data-Types :** Wrapper Classes(Byte, Short, Integer, Long, Float, Character, Double, Boolean) (Other objects)

---

17. What is type casting? Mention its types.

A type casting is the process of converting one data-types into another data-type.

- **Implicit Type Casting :** It is the process of converting one types of data into another type of data internally by the compiler.

```
public static void main(String[] args) {
        byte a = 10;
        int b;
        b=a;//implicit type casting
        System.out.println(b);
}
```

- **Explicit Type Casting :** It is the process of converting one types of data into another type of data externally by the programmer.

```
public static void main(String[] args) {
        int a = 10;
        byte b;
        b=(byte)a;//explicit type casting
        System.out.println(b);
    }
```

---

18. Pattern Program - 1

```
/*
 * 1 2 3 4 5
 * 1 2 3 4 5
 * 1 2 3 4 5
 * 1 2 3 4 5
 * 1 2 3 4 5
 *
 * Algorithm
 * --> It is in form of row x columns
 * --> we have 2 entities i.e row and column
 * --> we must have 2 loops to complete as we have 2 entities
 * --> row is represent by i
 * --> column(number) is represent by j
 */


import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5

        for(int i=1;i<=n;i++) {
            for(int j=1;j<=n;j++) {
                System.out.print(j+" ");
            }
            System.out.println();
        }

    }
}
```

---

19. Pattern Program - 2

```
/*
 * 1 1 1 1 1
 * 2 2 2 2 2
 * 3 3 3 3 3
 * 4 4 4 4 4
 * 5 5 5 5 5
 *
 *
 * --> It is in form of row x columns
 * --> we have 2 entities i.e row and column
 * --> we must have 2 loops to complete as we have 2 entities
 * --> row is represent by i
 * --> column(number) is represent by j
 */


import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5

        for(int i=1;i<=n;i++) {
            for(int j=1;j<=n;j++) {
                System.out.print(i+" ");
            }
            System.out.println();
        }

    }
}
```

20. Pattern Program - 3

```
/*
 * 1    2   3   4   5
 * 6    7   8   9   10
 * 11   12  13  14  15
 * 16   17  18  19  20
 * 21   22  23  24  25
 *
 * --> It is in form of row x columns
 * --> we have 2 entities i.e row and column
 * --> we must have 2 loops to complete as we have 2 entities
 * --> row is represent by i
 * --> column(number) is represent by j
 * --> create a variable and initialize it count=1
 * --> print the count and increment it after printing it each time
 */


import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        int count=1;
        for(int i=1;i<=n;i++) {
            for(int j=1;j<=n;j++) {
                System.out.print(count+"    ");
                count++;
            }
            System.out.println();
        }
    }
}
```

---

21. Pattern Program - 4

```
/*
 * 01   02  03  04  05
 * 06   07  08  09  10
 * 11   12  13  14  15
 * 16   17  18  19  20
 * 21   22  23  24  25
 *
 * --> It is in form of row x columns
 * --> we have 2 entities i.e row and column
 * --> we must have 2 loops to complete as we have 2 entities
 * --> row is represent by i
 * --> column(number) is represent by j
 * --> create a variable and initialize it count=1
 * --> print the count and increment it after printing it each time
 * --> apply a condition to check if the number is less than 10
 *          --> if number is less than 10 then attach 0 to it
 *          --> else print the number as it is
 */


import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        int count=1;
        for(int i=1;i<=n;i++) {
            for(int j=1;j<=n;j++) {
                if(count<10) {
                    System.out.print("0");
                    System.out.print(count+"    ");
                    count++;
                }
                else {
                    System.out.print(count+"    ");
                    count++;
                }
            }
            System.out.println();
        }

    }
}
```

---

22. Pattern Program - 5

```
/*
 * 1 0 1 0 1
 * 1 0 1 0 1
 * 1 0 1 0 1
 * 1 0 1 0 1
 * 1 0 1 0 1
 */

import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        for(int i=1;i<=n;i++) {
            for(int j=1;j<=n;j++) {
                if(j%2==0) {
                    System.out.print("0 ");
                }
                else {
                    System.out.print("1 ");
                }
            }
            System.out.println();
        }

    }
}
```

23. Pattern Program - 6

```
/*
 * 1            --> row no: 1
 * 1 2          --> row no: 2
 * 1 2 3        --> row no: 3
 * 1 2 3 4      --> row no: 4
 * 1 2 3 4 5    --> row no: 5
 */

import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        for(int i=1;i<=n;i++) {
            for(int j=1;j<=i;j++) {
                System.out.print(j+" ");
            }
            System.out.println();
        }

    }
}
```

24. Pattern Program - 7

```


/*
 * 1 # # # # #
 * 1 2 # # # # #
 * 1 2 3 # # # # #
 * 1 2 3 4 # # # # #
 * 1 2 3 4 5 # # # # #
 *
 * 1                # # # # #
 * 1 2              # # # # #
 * 1 2 3            # # # # #
 * 1 2 3 4          # # # # #
 * 1 2 3 4 5        # # # # #
 */

import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        for(int i=1;i<=n;i++) {
            for(int j=1;j<=i;j++) {
                System.out.print(j+" ");
            }
            for(int j=1;j<=n;j++) {
                System.out.print("# ");
            }
            System.out.println();
        }

    }
}
```

---

25. Pattern Program - 8

```
/*
 * 5 4 3 2 1 # # # # #
 * 5 4 3 2 # # # # #
 * 5 4 3 # # # # #
 * 5 4 # # # # #
 * 5 # # # # #
 *
 * 5 4 3 2 1        # # # # #
 * 5 4 3 2          # # # # #
 * 5 4 3            # # # # #
 * 5 4              # # # # #
 * 5                # # # # #
 */

import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        for(int i=1;i<=n;i++) {
            for(int j=n;j>=i;j--) {
                System.out.print(j+" ");
            }
            for(int j=1;j<=n;j++) {
                System.out.print("# ");
            }
            System.out.println();
        }

    }
}
```

---

26. Pattern Program - 9

```
/*
 * 1    6    11   16   21
 * 2    7    12   17   22
 * 3    8    13   18   23
 * 4    9    14   19   24
 * 5    10   15   20   25

 */

import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        int count;
        for(int i=1;i<=n;i++) {
            count=i;
            for(int j=1;j<=n;j++) {
                System.out.print(count+"    ");
                count=count+n;
            }
            System.out.println();
        }

    }
}
```

---

27. Pattern Program - 10

```
/*
 * # # # # # 1
 * # # # # 2 2 2
 * # # # 3 3 3 3 3
 * # # 4 4 4 4 4 4 4
 * # 5 5 5 5 5 5 5 5 5
 *
 *
 * # # # # #            1                ---> 1 ---> 2*i-1 => 2*1-1=1
 * # # # #            2 2 2              ---> 3 ---> 2*i-1 => 2*2-1=3
 * # # #            3 3 3 3 3            ---> 5 ---> 2*i-1 => 2*3-1=5
 * # #            4 4 4 4 4 4 4         ---> 7 ---> 2*i-1 => 2*4-1=7
 * #            5 5 5 5 5 5 5 5 5       ---> 9 ---> 2*i-1 => 2*5-1=9
 */

import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the value of n : ");
        int n = scan.nextInt();// 5X5
        for(int i=1;i<=n;i++) {
            for(int j=n;j>=i;j--) {
                System.out.print("# ");
            }

            for(int j=1;j<=2*i-1;j++) {
                System.out.print(i+" ");
            }
            System.out.println();
        }

    }
}
```

---

27. Pattern Program - 10

```
/*
 * Program to create a simple calculator application using switch cases
 */

import java.util.Scanner;
public class Code3 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("User please enter the operation to be
performed : \n+"
                + "1. Addition\n"
                + "2. Subtraction\n"
                + "3. Multiplication\n"
                + "4. Division");
        int op = scan.nextInt();
        int val;
        switch (op) {
        case 1: {
            System.out.println("Addition Operartion is invoked.");
            System.out.println("Enter the value no: 1: ");
            int num1 = scan.nextInt();
            System.out.println("Enter the value no: 2: ");
            int num2 = scan.nextInt();
            val = num1 + num2;
```

```
                System.out.println("The result of addition is : "+val);
                break;
            }
            case 2: {
                System.out.println("Subtraction Operartion is invoked.");
                System.out.println("Enter the value no: 1: ");
                int num1 = scan.nextInt();
                System.out.println("Enter the value no: 2: ");
                int num2 = scan.nextInt();
                val = num1 - num2;
                System.out.println("The result of subtraction is : "+val);
                break;
            }
            case 3: {
                System.out.println("Multiplication Operartion is invoked.");
                System.out.println("Enter the value no: 1: ");
                int num1 = scan.nextInt();
                System.out.println("Enter the value no: 2: ");
                int num2 = scan.nextInt();
                val = num1 * num2;
                System.out.println("The result of multiplication is : "+val);
                break;
            }
            case 4: {
                System.out.println("Division Operartion is invoked.");
                System.out.println("Enter the value no: 1: ");
                int num1 = scan.nextInt();
                System.out.println("Enter the value no: 2: ");
                int num2 = scan.nextInt();
                val = num1 / num2;
                System.out.println("The result of division is : "+val);
                break;
            }
            default:
                System.out.println("wrong input selected");
            }
        }
    }
```

---

28. What are String in Java?

A String is defined as the series of characters which is enclosed within the double quotes(" ").

---

29. What are the different types of Strings in Java?

In java strings are classified into 2 types:

1. Mutable Strings
2. Immutable Strings

---

30. What is a Immutable strings in java? How do we represent them?

These are the String which will not change once it is created. It is represent using **java.lang.String class**

---

31. What is a Mutable strings in java? How do we represent them?

These are the String which can change once it is created. It is represent using :

1. **java.lang.StringBuffer class**
2. **java.lang.StringBuilder class**

---

32. Palindrome Program

```
import java.util.Scanner;

/*
 * Given the following
 * input    : madam
 * Output   : Given String is a palindrome

 note:
 next()--> used for collecting single word
 nextLine()--> used for collecting a statement

 */

public class Code4 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter the String: ");
        String str = scan.next();
        String str1="";

        /*
         i
         0  1   2   3   4
    str  m  a   d   a   m

    str1 m  a   d   a   m
         */

        for(int i=str.length()-1;i>=0;i--) {
            str1=str1+str.charAt(i);
        }

        if(str.equals(str1)==true) {
            System.out.println("Given String is a palindrome");
        }
        else {
            System.out.println("Given String is not a palindrome");
        }

    }
}
```

33. Count Vowels

```
import java.util.Scanner;

/*
 * input     : HElLO GooD Morning
 * Output    : Vowel Count = 6
 */
public class Code5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Please enter the String: ");
        String str = sc.nextLine();
        int vowel_cnt=0;

        for(int i=0;i<str.length();i++) {

if(str.charAt(i)=='a'||str.charAt(i)=='e'||str.charAt(i)=='i'||str.charAt(
i)=='o'
                || str.charAt(i)=='u') {
                vowel_cnt++;
            }
            else
if(str.charAt(i)=='A'||str.charAt(i)=='E'||str.charAt(i)=='I'||str.charAt(
i)=='O'
                    || str.charAt(i)=='U') {
                vowel_cnt++;
            }
            else {

            }
        }
        System.out.println("Vowel Count is = "+vowel_cnt);
    }
}
```

34. Word count.

```
import java.util.Scanner;

/*
 * input    : Hello GooD Morning
 * Output   : Word Count = 3
 */
public class Code5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Please enter the String: ");
        String str = sc.nextLine();
        int word_cnt=0;

        for(int i=0;i<str.length();i++) {
            if(str.charAt(i)==' ' && str.charAt(i+1)!=' ') {
                word_cnt++;
            }
        }
        word_cnt++;
        System.out.println("Word Count is = "+word_cnt);
    }
}
```

---

35. Replace vowels.

```
import java.util.Scanner;

/*
 * input    : Hello GooD Morning
 * Output   : H@ll@ G@@D M@rn@ng
 */
public class Code5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Please enter the String: ");
        String str = sc.nextLine();
        int word_cnt=0;
        String str1="";

        for(int i=0;i<str.length();i++) {
            if(str.charAt(i)=='a' || str.charAt(i)=='e' ||
                    str.charAt(i)=='i' || str.charAt(i)=='o'
                    || str.charAt(i)=='u' || str.charAt(i)=='A' ||
str.charAt(i)=='E' ||
                    str.charAt(i)=='I' || str.charAt(i)=='O'
                    || str.charAt(i)=='U') {
                str1 = str1 + '@';
            }
            else {
                str1 = str1 + str.charAt(i);
            }
        }
        System.out.println("Word Count is = "+str1);
    }
}
```

36. Replace Vowels with special character indivitually.

```
import java.util.Scanner;

/*
 * input    : aeiouAEIOU
 * Output   : @#$%&@#$%&
 */
public class Code5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Please enter the String: ");
        String str = sc.nextLine();
        int word_cnt=0;
        String str1="";

        for(int i=0;i<str.length();i++) {
            if(str.charAt(i)=='a' || str.charAt(i)=='A') {
                str1 = str1 + '@';
            }
            else if(str.charAt(i)=='e' || str.charAt(i)=='E') {
                str1 = str1 + '#';
            }
            else if(str.charAt(i)=='i' || str.charAt(i)=='I') {
                str1 = str1 + '$';
            }
            else if(str.charAt(i)=='o' || str.charAt(i)=='O') {
                str1 = str1 + '%';
            }
            else if(str.charAt(i)=='u' || str.charAt(i)=='U') {
                str1 = str1 + '&';
            }
            else {
                str1 = str1 + str.charAt(i);
            }
        }
        System.out.println("Word Count is = "+str1);
    }
}
```

---

36. Segregation of Strings.

```
/*
 * input    : He#$%%^ll&&&&o Goo^^%$$##d Mo$%^&rn$%^&ing Wel&*(co*&%$#me
o%n bo$%^&ar#$%^d
 * Output   : HelloGoodMorningWelcomeonbaord
 *            #$%%^ll&&&&^^%$$##$%^&$%^&&*(*&%$#%$%^&#$%^
 */
public class Code5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Please enter the String: ");
        String str = sc.nextLine();
        String str1="";
        String str2="";


        for(int i=0;i<str.length();i++) {
            if((str.charAt(i)>='a' && str.charAt(i)<='z') ||
                    (str.charAt(i)>='A' && str.charAt(i)<='Z')) {
                str1 = str1 + str.charAt(i);
            }
            else {
                str2=str2+str.charAt(i);
            }
        }
        System.out.println("Character String is = "+str1);
        System.out.println("Symbols String is = "+str2);
    }
}
```

---

37. What is an Array Data Structure in Java?

An Arrays is defined as the collection Homogeneous data(similar data).

---

38. What are the advantages of array datastructure?

There are 3 important advanatges of array data structure:

1. Creation of an array is simple
2. Storing the data inside an array is simple
3. Retrieving the data from an array is simple

---

39. Program on 1-Dimension Array

```
import java.util.Scanner;

/*
 * Program to store and retrieve name of the employees
 */
public class ArraysCode1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter to number of employee: ");
        int n = sc.nextInt();//5
        /*
         * []       ----> 1-Dimension array
         * [][]     ----> 2-Dimension array
         */
        String arr[] = new String[n];

        //Storing the data
        for(int i=0;i<arr.length;i++) {
            System.out.println("Enter the name of the Employee no: "+
(i+1));
            arr[i]=sc.next();
            /*
             * arr[0]=Rohit
             * arr[1]=Gill
             * arr[2]=Virat
             * arr[3]=Surya
             * arr[4]=MSD
             */
        }
        //Fetching the data
        for(int i=0;i<arr.length;i++) {
            System.out.println("the name of the Employee no: "+(i+1)+" is
= "+arr[i]);
        }
    }
}
```

---

40. Program on 2-Dimensional Array.

```
import java.util.Scanner;

/*
 *  Company     Employee
 *     0            3
 *     1            3
 *     2            3
 *     3            3
 *
 *  Write a code to collect the names of employees from each company
 *
 * 2 entities ----> 2 loops ----> 2 [] ----> 2-Dimensional Array
 */
public class ArraysCode1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter to number of companies: ");
        int m = sc.nextInt();//5

        System.out.println("Enter to number of employees in each company:
");
        int n = sc.nextInt();//5

        String arr[][] = new String[m][n];

        //Storing the data
        for(int i=0;i<arr.length;i++) //companies
        {
            System.out.println("Inside company no: "+(i+1));

            for(int j=0;j<arr[i].length;j++) {
                System.out.println("Enter the name of employee no:"+
(j+1));
                arr[i][j]=sc.next();
            }
            /*
             * arr[0][0]=Rohit
             * arr[0][1]=Gill
             * arr[0][2]=Virat
             * -----------------
             * arr[1][0]=Surya
             * arr[1][1]=MSD
             * arr[1][2]=Raina
             * -----------------
             * arr[2][0]=Rahul
             * arr[2][1]=Kaif
             * arr[2][2]=Yuvraj
             * -----------------
             * arr[3][0]=Sachin
             * arr[3][1]=Laxman
             * arr[3][2]=Balaji
             */
        }
        //Fetching the data
        for(int i=0;i<arr.length;i++) //companies
        {
            System.out.println("Inside company no: "+(i+1));

            for(int j=0;j<arr[i].length;j++) {

                System.out.println("the name of the Employee no: "+(j+1)+"
is = "+arr[i][j]);
            }
            }
        }
    }
```

---

41. Program on 3-Dimensional array

```
import java.util.Scanner;

/*
 *  Organization Company    Employee
 *      0            0           3
 *                   1           3
 *      1            0           3
 *                   1           3
 *
 *  Write a code to collect the names of employees from each company
 *
 * 2 entities ----> 2 loops ----> 2 [] ----> 2-Dimensional Array
 */
public class ArraysCode1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter to number of organizations: ");
        int a = sc.nextInt();//5

        System.out.println("Enter to number of companies in each 
organization: ");
        int m = sc.nextInt();//5

        System.out.println("Enter to number of employees in each company 
of each organization: ");
        int n = sc.nextInt();//5

        String arr[][][] = new String[a][m][n];//---> 3-D

        //Storing the data
        for(int i=0;i<arr.length;i++) //companies
        {
            System.out.println("Inside organization no: "+(i+1));

            for(int j=0;j<arr[i].length;j++) //companies
            {
                System.out.println("Inside company no: "+(j+1));

                for(int k=0;k<arr[i][j].length;k++) {
                    System.out.println("Enter the name of employee no:"+
(k+1));
                    arr[i][j][k]=sc.next();
                }
            }
            /*
             * arr[0][0][0]=Rohit
             * arr[0][0][1]=Gill
             * arr[0][0][2]=Virat
             * arr[0][1][0]=Rohit
             * arr[0][1][1]=Gill
             * arr[0][1][2]=Virat
             *
             * arr[1][0][0]=Rohit
             * arr[1][0][1]=Gill
             * arr[1][0][2]=Virat
             * arr[1][1][0]=Rohit
             * arr[1][1][1]=Gill
             * arr[1][1][2]=Virat
             */
        }
        //Fetching the data
        for(int i=0;i<arr.length;i++) //companies
        {
            System.out.println("Inside organization no: "+(i+1));

            for(int j=0;j<arr[i].length;j++) //companies
            {
                System.out.println("Inside company no: "+(j+1));

                for(int k=0;k<arr[i][j].length;k++) {

                    System.out.println("the name of the Employee no: "+
(k+1)+" is = "+arr[i][j][k]);
                }
            }
        }
    }
}
```

42. What is method overloading? / What is compile time polymorphism? / What is virtual polymorphism?

A method overloading is the illusion in the minds of the programmer that one single method will be performing multiple tasks. But, inreality it is the independent methods which will be performing the independent tasks.
In method overloading the methods can have same name, same number of parameters, same type of parameter but it cannot have the parameters in same order.

It is also called Virtual Polymporphsim.
Virtual ----> Not Real
Polymorphism ----> 1:M

It is also called compile time polymorphism because this polymorphic approach is seen during the compilation time.

```
class Calculator{
    int add(int a,int b) {
        return a+b;
    }
    int add(int a,int b,int c) {
        return a+b;
    }
    float add(int a, float b) {
        return a+b;
    }
    float add(float a, int b) {
        return a+b;
    }
    float add(float a, float b) {
        return a+b;
    }
    double add(int a, float b, double c) {
        return a+b+c;
    }
    double add(float a, int b, double c) {
        return a+b+c;
    }
    double add(float a, double b, int c) {
        return a+b+c;
    }
}
public class Launch3 {
    public static void main(String[] args) {
        int a=10,b=20,c=30;
        float a1=10.11f,b1=20.22f,c1=30.33f;
        double a2=100.111,b2=200.222,c2=300.333;

        Calculator cal = new Calculator();
        System.out.println(cal.add(a,b));//int,int
        System.out.println(cal.add(a1, b1));//float,float
        System.out.println(cal.add(a, b1, c2));//int,float.double
    }
}
```

---

43. What is the use of static keyword? Does making a variable static makes it constant? Justify.

Static Basically is used to create the memory only once in the static segment. In static segment we have 3 important members :

1. static variable
2. static block
3. static method

No, making a variable static makes it to get the memory only once which can be accessed multiple times. The static variables is not a constant.

- Static members are VIP members
- Program execution will always start from static segment
- In-order to call the static method even before creating the object we make use of the class name
- Static variables are the instance variables with the static keyword associated with it
- Static varibales get default values assigned

```
class SampleTest{
    static int a,b,c;//static variables --->1
    int m,n,o;//non-static variables

    //static block -- can access only static variables --->2
    static
    {
        a=100;
        b=200;
        c=300;
//      m=1000;//error
//      n=2000;//error
//      o=3000;//error
    }

    //Non-static block -- can access both static and non-static variables
    {
        a=1001;
        b=2001;
        c=3001;
        m=10001;
        n=20001;
        o=30001;
    }

    //static methods -- can access only static variables
    static void display(){
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
//      System.out.println(m);//error
//      System.out.println(n);//error
//      System.out.println(o);//error
    }
    //static methods -- can access both static and non-static variables
    void display1(){
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
        System.out.println(m);
        System.out.println(n);
        System.out.println(o);
    }
}
public class Launch4 {
    public static void main(String[] args) {
        System.out.println(SampleTest.a);
        System.out.println(SampleTest.b);
        System.out.println(SampleTest.c);
        System.out.println("---------");
        SampleTest st = new SampleTest();
        st.display1();
        System.out.println("---------");
        st.display();
    }
}
```

---

45. Programming example for static.

```
import java.util.Scanner;

class Farmer{
    double principle;
    static double rate;
    double time;
    double simpleInterest;

    static {
        rate=2.5;//memory for rate will be created only once for all the
objects
    }
    void collectData() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the Principle Amount: ");
        principle=sc.nextDouble();

        System.out.println("Enter the Time Needed (In Years): ");
        time=sc.nextDouble();

    }
    void calculate() {
        simpleInterest = (principle * time * rate)/100;
    }

    void showSimpleInterest() {
        System.out.println("The Simple Interest is : "+simpleInterest);
    }
}
public class Launch5 {
    public static void main(String[] args) {
        Farmer f1 = new Farmer();
        Farmer f2 = new Farmer();
        Farmer f3 = new Farmer();

        f1.collectData();
        f2.collectData();
        f3.collectData();

        f1.calculate();
        f2.calculate();
        f3.calculate();

        System.out.println("Simple Interest for First farmer is : ");
        f1.showSimpleInterest();
        System.out.println("Simple Interest for Second farmer is : ");
        f2.showSimpleInterest();
        System.out.println("Simple Interest for Third farmer is : ");
        f3.showSimpleInterest();

    }
}
```