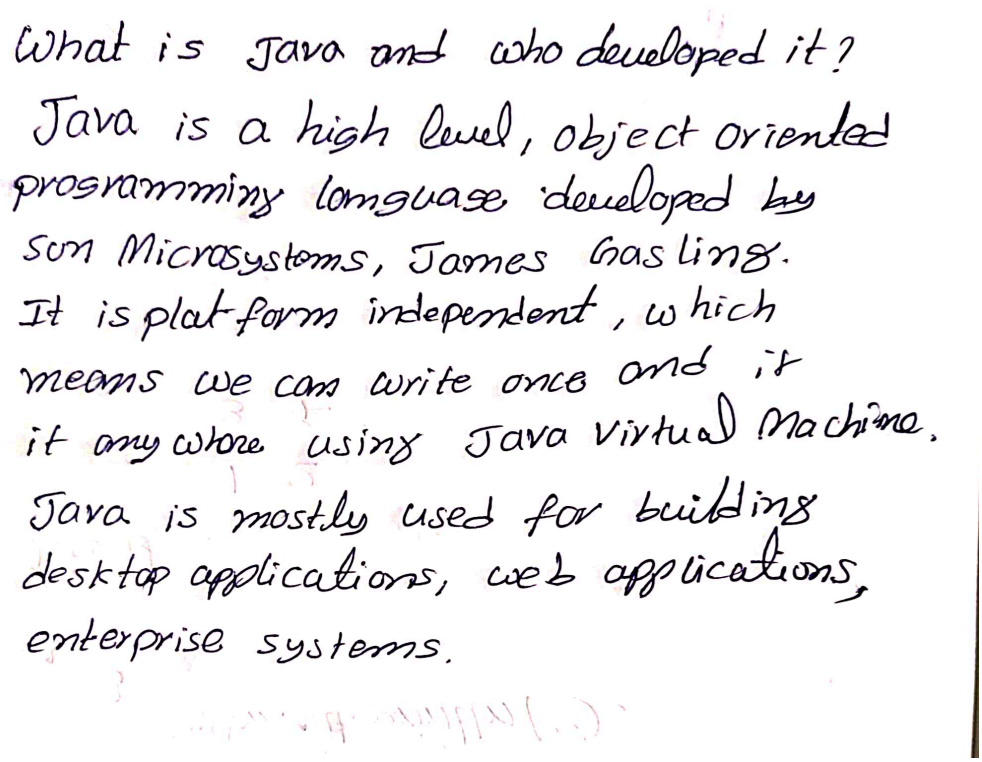
Java Fundamentals - Exercises

Java Introduction & Features

- - What is Java and who developed it?



- - List any 5 features of Java.

→ List any 5 features of Java.

i) Object Oriented Programming

Java is not a purely object oriented because it having primitive data types and static members.

ii) Platform independent :-

When we compile a java code it converts into byte code with extension of .class file. Java Virtual Machine is responsible for platform independent.

Once you create java file. It will execute in any type of operating system which is having JVM.

III) Secure and Robust :-

- No explicit pointer.
- Java program runs inside a virtual machine sandbox.
- Bytecode verifier checks code for illegal code.
- It having Exception handling. It helps to deal with exceptions.
- It having a strong Memory manasment system.

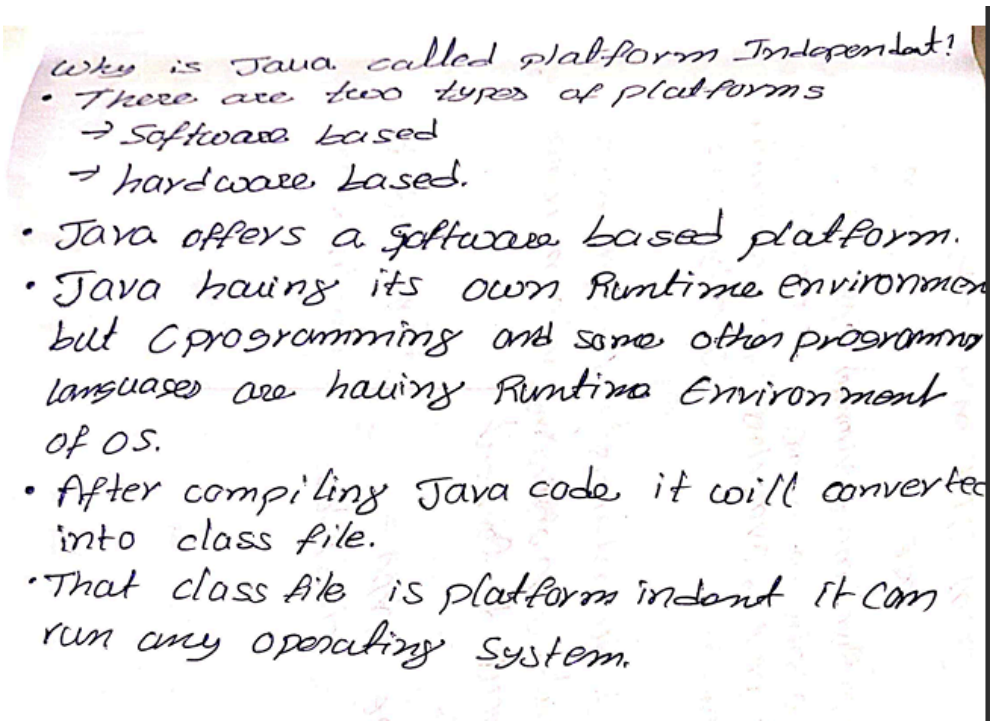
IV) High performance :-

Java is faster than other interpreted languages like python.
Java is converted into bytecode, for Interpretion and JIT compilation process.

V) Multithreading :-

Multithreading means just like multi tasking. Java allows multiple threads to run at the same time, which improves the CPU utilization.

- - Why is Java called platform-independent?



JVM, JRE, JDK - Architecture

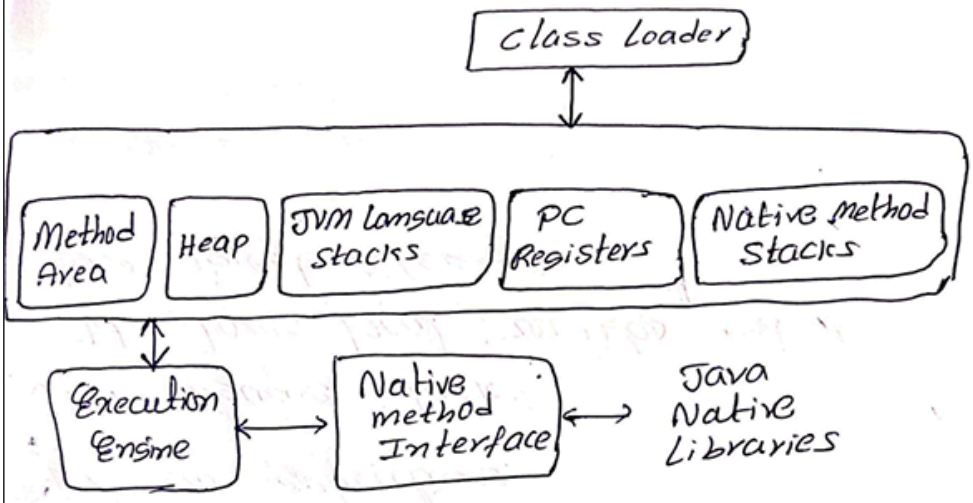
- - Differentiate between JVM, JRE, and JDK.

Differentiate between JVM, JRE and JDK.

JDK	JRE	JVM
Java development kit is a software development kit used to develop application.	Java Runtime Environment is a software package that provides JVM, class libraries.	Java virtual Machine is provides an environment for the execution of Java byte code.
JDK is used by developers to write, compile and debug java programs	Used by end-users to execute java applications without development tools.	Converts byte code into machine code.
JDK = JRE + Development tools	JRE = JVM + class files	JVM = provides a run time environment

- Draw and explain the architecture of JVM.

Draw and explain the architecture of JVM.



Class Loaders :-

JVM has classloader. Whenever we run java code or compile .class (byte) code is generated. class loader loads the class file.

Method Area :-

Method Area stores Runtime constant pool. Field Data and Method data.

Heap :-

Heap is the run time data area in which objects are allocated.

Heap stores object values when object is created by using a new keyword.

Heap is managed by the JVM's garbage collector, which collections involves reclaiming memory occipied by unused objects and freeing up spaces for new applications.

JVM language stack :-

It stores local variables and it stores object references.

Execution Engine :-

It contains A virtual processor.

Interpreter : Read bytecode stream than execute the instructions.

JIT (Just in Time) compile

Java Native interface :-

It is a framework which provides interface to communicate with other languages.

- What role does the JIT compiler play?

→ What role does the JIT compiler play?

JIT compiler is responsible for converting byte code into machine code. It optimising the performance of interpreted programming language.

This JIT compiler allows for faster execution.

Java Program Structure

- - Write a basic Java program to print 'Hello, Java!'.

→ Write a basic java program to print "Hello, Java!".

```
class Hello {
    public static void main(String[] args){
        System.out.println("Hello, Java!");
    }
}
```

- - Explain each part of a Java program.

→ Explain each part of Java program.

- class is a keyword is used to declare a class in Java
- public is a keyword is an access modifier that represents visibility.
- static, If we declare any method with static it no need to create an object to invoke the static method or variable.
- void is return type of the method. It means the method does not return any value.

- main() method is entry point of Java Programming.
- String[] args is used for command line argument.
- System.out.println() is used to print the statement. System is a class, out is an object and println() is a method.

- - What is the role of the 'main' method?

→ What is the role of the 'main' method.

The java main method is the entry point for executing a java program. the main method may contain methods and objects variables.

Keywords and Identifiers

- Define keywords with examples.

Keywords and Identifiers :-

→ Define keywords with examples.

Java keywords are also known as reserved words.

Keywords are predefined words by java so they cannot be used as a variable or object name or class name.

Examples for keywords :-

Data type → byte, char, double ...

Control flow → if, else, else if, switch ....

Modifiers → abstract, final, public .....

class Related → extends, class ...

Exception Handling → try, catch, throw ...

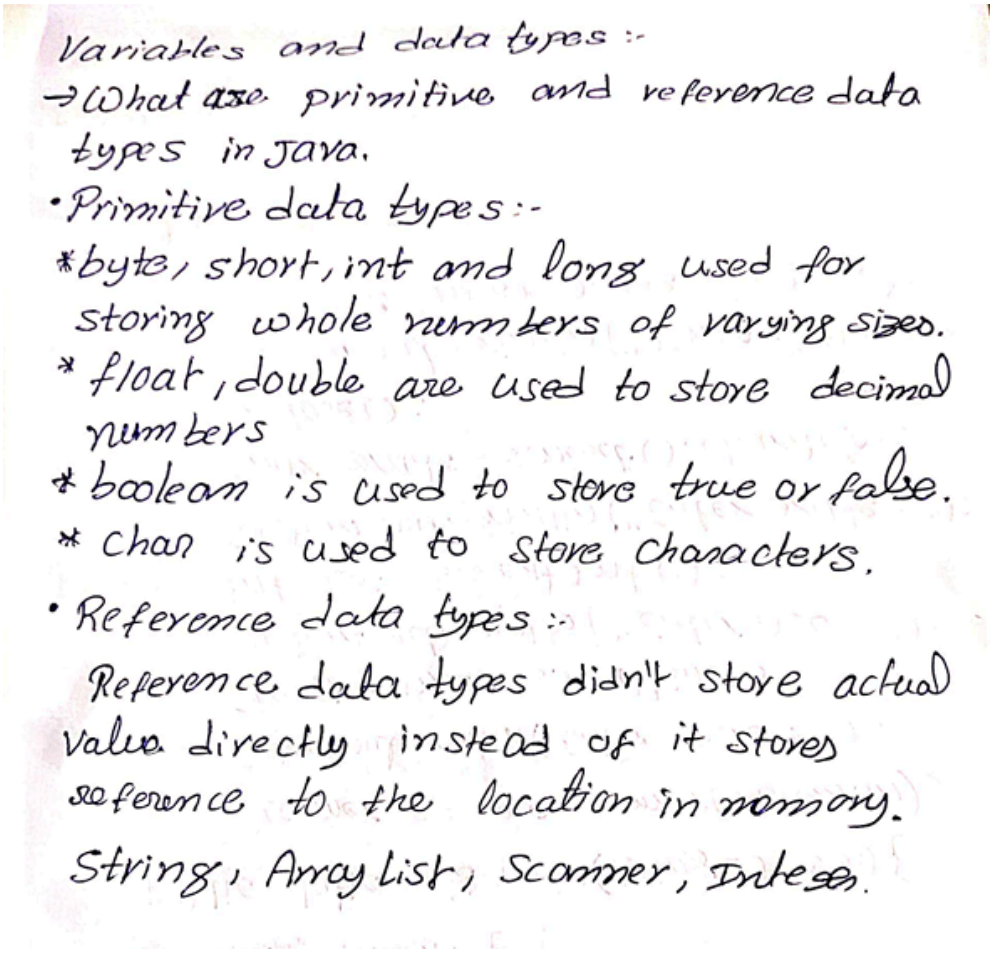
- What are identifiers and what rules apply to them?

→ What are identifiers and what are rules apply to them.

- An Identifier must starts with a letter or in specid characters either $ or _ is allowed.
- An identifier cannot start with digits
- After the first letter it contain any character.
- No reserved keywords are used as a identifier.
- No blank spaces.
- Case Sensitive.

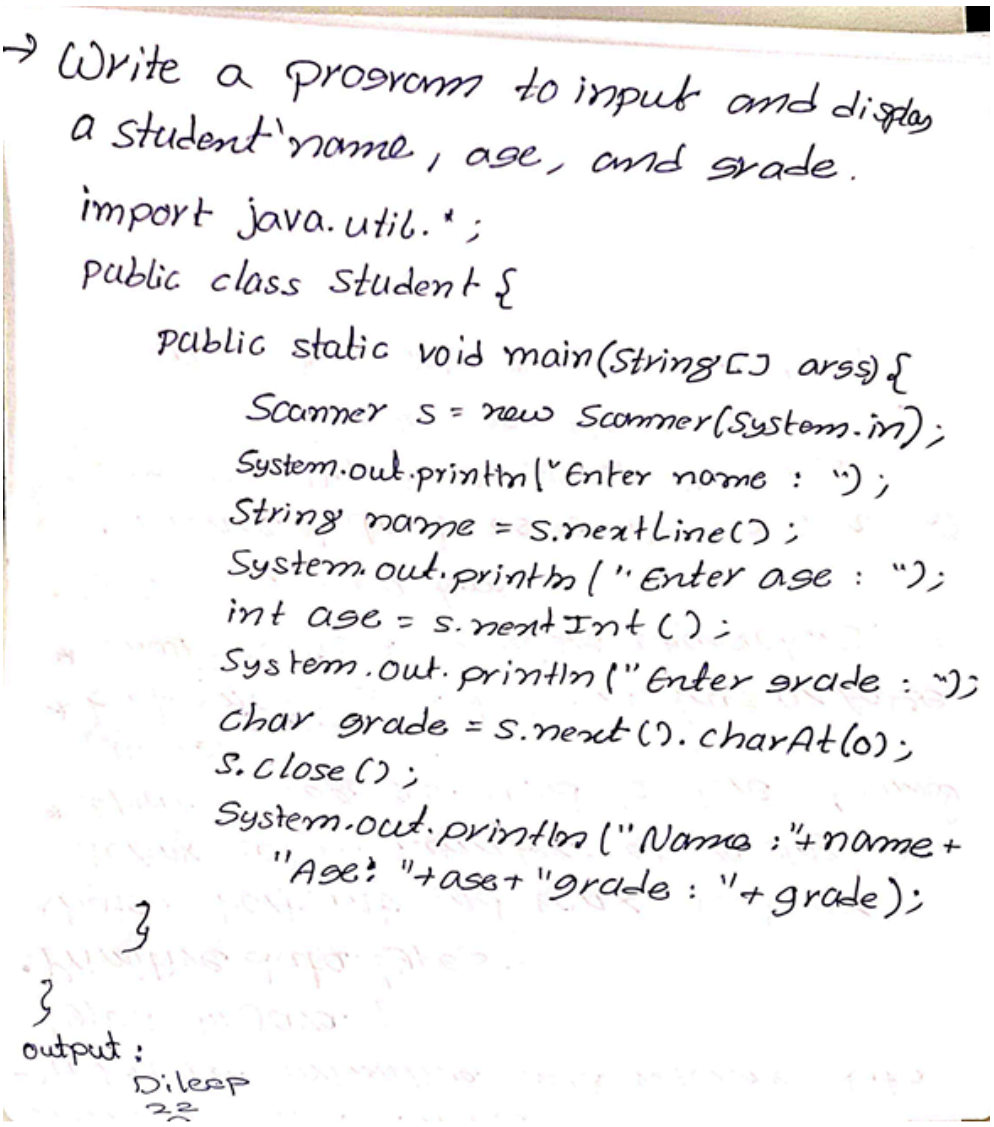
Variables and Data Types

- - What are primitive and reference data types in Java?


Variables and data types :-
→ What are primitive and reference data types in Java.
• Primitive data types :-
* byte, short, int and long used for storing whole numbers of varying sizes.
* float, double are used to store decimal numbers
* booleam is used to store true or false.
* char is used to store characters.
• Reference data types :-
Reference data types didn't store actual value directly instead of it stores reference to the location in memory.
String, Array List, Scanner, Integer.

- - Write a program to input and display a student's name, age, and grade.

→ Write a program to input and display a student's name, age, and grade.



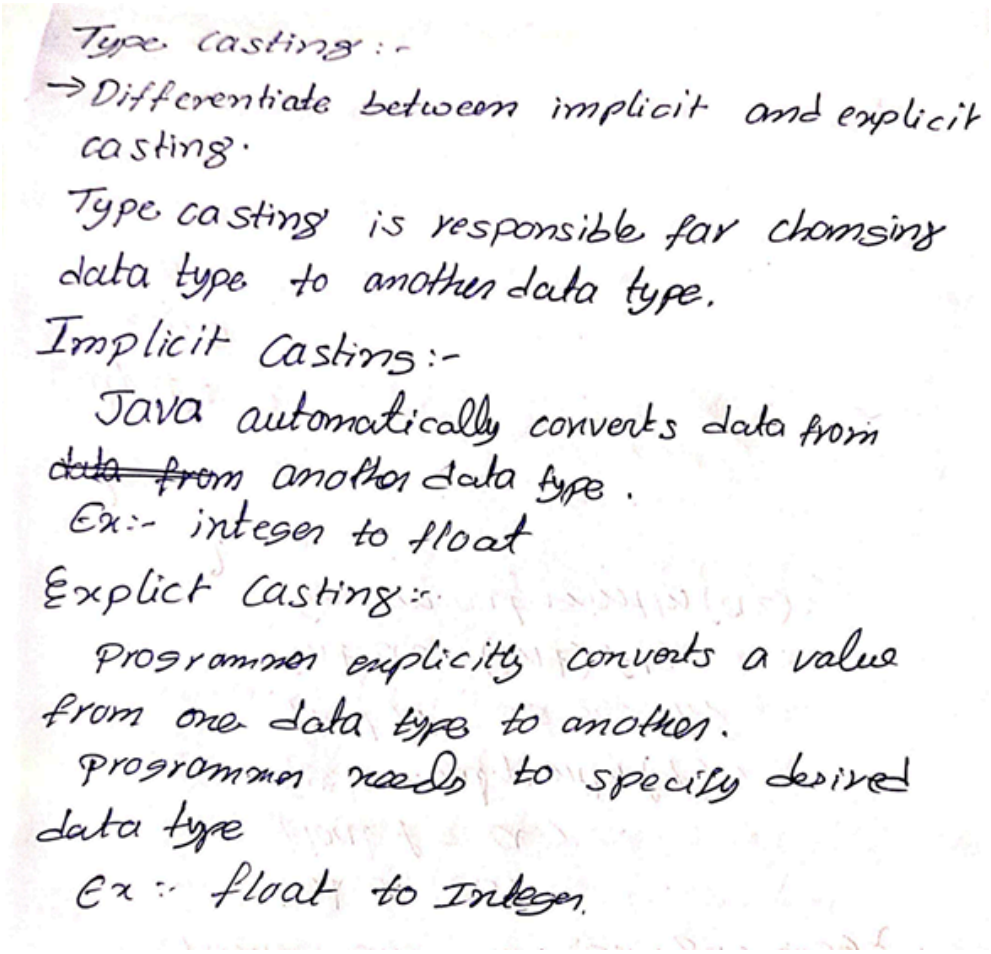
```
import java.util.*;
Public class Student {
    Public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter name : ");
        String name = s.nextLine();
        System.out.println("Enter age : ");
        int age = s.nextInt();
        System.out.println("Enter grade : ");
        char grade = s.next().charAt(0);
        S.close();
        System.out.println("Name :"+name+
            "Age: "+age+"grade : "+grade);
    }
}
```

output :

Dileep
22

Type Casting

- - Differentiate between implicit and explicit casting.



- - Write a program to demonstrate both types of casting.

- Write a program to demonstrate both types of casting.

```
public class Main{
    public static main (String[] args){
        int a = 65;
        float f = a;
        System.out.println (f);
        float f1 = 25.3654f;
        int a2 =(int) f1;
        System.out.println (a2);
    }
}
```

Output :

65.0

25

- What happens when narrowing conversion fails?

→ What happens when narrowing conversion fails?

When narrowing conversion fails, it typically results in run-time error due to data loss.

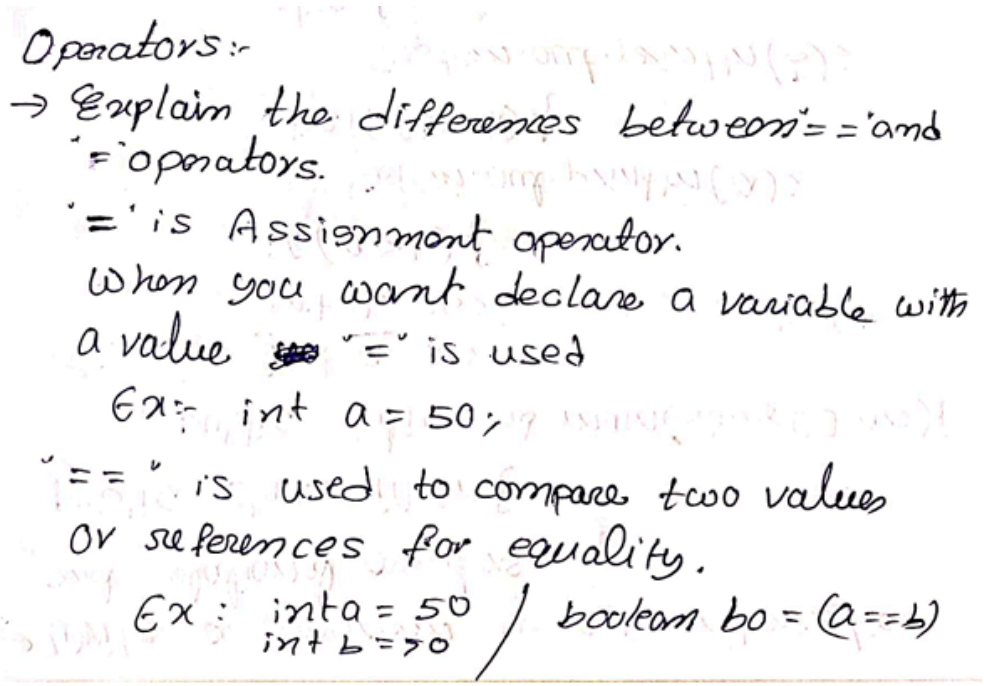
Ex:- `float f = 25.3654f;`

`int a = (int)f;`

so from above example f is assigned to a. but it losses 0.3654

Operators

- Explain the difference between == and = operators.



- - Write a program using arithmetic and relational operators.

→ Write a program using arthemetic and relational operators.

```
public class Main{
    public static void main(String[] args){
        int a = 10;
        int b = 15;
        if (a>b) {
            System.out.println(a);
        elseif (b>a) {
            System.out.println(b);
        else {
            System.out.println ("equal");
        }
    }
}
```

output:

15

- List and explain logical operators in Java.

→ List and explain logical operators in java.

Logical AND (&&):

to perform logical AND at least two conditions should be there.

if any one of the condition failed or false. The entire condition failed.

Ex :- if (x > 0 && x < 10)

Logical OR (||):

If any one of the condition true the whole condition true.

Ex :- if (x > 0 || x < 10)

Logical NOT (!):

This is a unary operator. it inverts the boolean value of its operand.

Ex :- boolean istrue = true

System.out.println (!istrue);

Control Flow (Control Statements, Loops)

- - Write a program to find the largest of three numbers using if-else.

Control Flow

→ Write a program to find the largest of three numbers using if-else.

```
Public class Largest{
    public static void main(String[] args){
        int a=2, b=4, c=3;
        if (a>b){
            if (a>c){
                System.out.println(a);
            }
            else{
                System.out.println(c);
            }
        else{
            if (b>c){
                System.out.println(b);
            }
            else{
                System.out.println(c);
            }
        }
```

- - Demonstrate the use of switch-case.

→ Demonstrate the use of switch-case.

Marks → Pass or fail

```
public class Grade {
    public static void main(String[] args){
        marks = 45;
        switch(true) {
            case (marks>30):
                System.out.println("pass");
                break;
            case (marks<30):
                System.out.println("fail");
        }
    }
}
```

output:

pass

- Write a loop that prints numbers from 10 to 1.

→ Write a loop that prints 10 to 1.

```
public class Numbers{
    public static void main(String[] args){
        for (int i=10; i>=0; i--){
            System.out.println(i);
        }
    }
}
```

Output: 10
9
8
7
6
5
4
3
2
1

Arrays

- - What are arrays in Java?

Arrays :-
→ What are array in Java?
Arrays are referenced data types.
Store Primitive and Objects.
Continuous Memory Allocation
Index based and Fixed length

It stores elements of same data types.

Declaring of an Array:-

```
int arr1[];
int[] arr2;
```

Initialization of An Array:

```
int arr[] = new int[5];
int[] arr = new int[]{1,2,3,4};
```

- - Write a program to find the sum of elements in an array.

```
import java.util.*;
public class SumArray{
  public static void main(String[] args){
      int sum=0;
      int[] numbers = {1,2,3,4,5};
      for(i=0; i<numbers.length; i++){
          sum+=numbers[i];
      }
     System.out.println(sum);
  }
}
```

Output:-

```
10
```

- - Declare and initialize a 2D array and print its elements..

→

```
public class TwoDimensionalArray{
    public static void main(String[] args){
      int[] numbers = {{1,2},{3,4}};
      for(int i=0; i<numbers.length; i++){
        for(int j=0; j<i.length; j++){
            System.out.print(numbers[j]+" ");
        }
        System.out.println("");
    }
}
```

Output :

```
1 2
3 4
```