Assignment-1

1)a) What is JVM, JDK, JRE

A) JVM (Java virtual Machine)

The JVM is a caucial part of java technology It is a virtualised suntime environment that allows java applications to sun on vasious platforms without modifications. The JVM intesprets os compiles the java bytecode into machine code that can be executed by host opesating system

JDK (Java Development KH)

JDK is a softwase development kit used by develo pess to cseate java applications. It contains tools such as the java compiles (Javac), libsaries, debugging tools and utilities necessasy fos java development.

JRE (Java suntime environment)

It is sequised to sun java applications on a computes. It includes the JVM, neccessasy libsaries and suntime components. when you want to sun a Java psogsam on youss system. you typically need to have the JPE installed

b) Difference between procedure oriented language and object-oriented program language

A) Procedure oriented	object oriented
① This makes use of a step-by-step approach for breaking down a task into a collection of routines and variables by following instructions	① This uses objects and classes for creating models based on the real world environment
② It is less secure	② It is more secure comparted to it
③ It is structure oriented	③ It is object-oriented
④ It prioritizes function over data	④ It priorities data over function
⑤ It does not provide any inheritance	⑤ It provides inheritance in 3 modes: protected private, public
⑥ examples: C, pascal COBOL etc	⑥ examples: Java, python

(2) a) what feature of Java makes it platform independent and portable?

A) Java is a programming language and a platform. Java is high level, robust, secured and object-oriented language

Platform:
Any hardware or software environment in which a program runs is known as a platform

Features of Java

1. simple
2. object-oriented
3. platform independent
4. secured
5. Robust
6. Architecture Neutral
7. portable
8. High performance
9. Distributed
10. Multi-threaded

Platform Independent
A platform is a hardware or software environment in which a program runs. Java provides software based platform
It has two components
1. Run time environment
2. API (Application programming interface)

Java code can be run on multiple platforms eg: windows, Linux, macOS etc.
Java code is compiled by the compiler and converted into byte codes.
This byte code is platform Independent

Portable
portable means able to be easily carried or moved, therefore we may carray Java bytecode to any platform there is a possibility to use the same software in different environments
∴ These are the features which makes java portable and platform independent

b) Is Java a Robust language? Justify your answer.

A) yes, Java is a robust language.
Robust means strong. Java uses strong memory management. There are lack of pointers that avoids security problem. There is automatic garbage collection in java. there is expection handling. All these makes java robust

3) a) Differentiate between class & object

A) object means a real-world entity. Such as pen, chair etc. object-oriented programming is a methodology or paradigm to design a program using class & objects.

It simplifies the software development & maintaience by providing some concepts:

* object
* class
* Inheritance
* polymorphism
* Abstraction
* Encapsulation

class	object
① A class is a blue print or template for creating objects	① An object is an instance or realization of a class
② A class is a static concept it remains same throughout the program's execution.	② objects are dynamic & exist in memory during program execution
③ A class doesn't consume memory space on its own	③ each object consumes memory space to store
④ If you think of a "car" as a class, it defines what a car is in terms of its attributes (color, model etc) and behaviours (start, stop etc)	④ If you create an object "my car" from the "car" class it represents a specific car, with its own color, model etc

b) Demonstrate constructor overloading concept

A) constructor overloading concept is a concept in object-oriented programming that allows a class to have multiple constructor with different parameters lists. These overloaded constructors provide flexibility when creating objects of a class, as they can accept different combinations of input values

→ In a class you can define multiple constructors with different parameter lists

→ constructor overloading allows you to create constructors that accept different types of parameters

→ overloaded constructors are used to initialize the objects state when it is created.

→ It provides flexibity when working with objects of a class.

4) Explain the basic concepts of object-oriented Programming.

A) Object-oriented programming is a methodology or a paradigm to design a program using classes and objects. It simplifies the software development and maintainance by providing some concepts.

1. object
2. class
3. polymorphism
4. Inheritance
5. Abstraction
6. encapsulation

1. object:
Any entity that has state and behaviour is known as object ex: chair, pen, table etc. It can be physical.

2. class:
collection of objects is called class. It is a logical entity. It is a blueprint or structure of an object

3. Inheritance:
when one object aquires all the properties and behaviours of parent object. i.e known as inheritance It provides code reusability it is used to achieve run-time polymorphism

④ polymorphism:
when one task is performed by different ways i.e known as polymorphism for example to draw of shaps like circles, rectangles & triangles

⑤ Abstraction:
Hiding intesnal details and showing functionality is known as abstraction for example phone call we don't know the intesnal processing. In Java we use abstract class & interface to achieve abstraction

⑥ encapsulation:
Binding code and data together into a single unit is known as encapsulation. A Java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here

5a) What is type casting? List and explain types of type casting with suitable examples

A) Type casting:

Type casting a method or process that converts the data type into another data type in both ways manually & automatically.

Type casting is of two types

(1) Narrowing type casting
(2) widening type casting

(1) Narrowing type casting:

Converting a higher data type into a lower one is called Narrowing type casting. It is done manually by programmer It is known as explicit conversion or casting up.

double -> float -> long -> int -> char -> short -> byte

ex: double double value = 3.14915

int intvalue = (int) double value;
// explict casting from double to int

(2) Widening type casting:

converting a lower data type into higher one is called widening type casting. also known as implicit conversion or casting down

byte -> short -> char -> int -> long -> float -> double

ex: int intvalue = 42;
double double value = intvalue // implicit casting from int to double

b) List the primitive data types available in Java

A) Data types specify the data, diff sizes and values that can be stored.

① primitive data types ② Non-primitive type

These are 8 types of primitive data types

(1) Boolean data types:

The boolean data type is used to store only two possible values. True & false used to check true/false conditions ex: Boolean one = false

(2) Byte data type

It is used to save memory in large arrays where memory saving is most required. It can also be used in place of int data type

ex: byte a=10; byte b=-20

(3) Short data type:

The short data type can also be used to save memory just like byte data type. A short data type is 2 times smaller than an integer.

ex: Short s=10000; short x=-50000

(4) Int data type:

The int is used as a default data type for integral values unless if there is no problem about memory

ex: int a = 100000 int b = -200000

(5) long data type:

The long is used when you need a range of values more than those provided by int

ex: long a = 100000 ; long b = -200000

(6) float data type:

It is recommended to use a float (instead of double) if you need to save memory in large assays of floating point numbers

ex: float f1 = 343.5f

(7) Double data type:

The double data type is used for decimal values just like float

ex: double d1 = 12.3

(8) char data type:

The char data type is used to store characters

ex: char letter A = 'A'

Data type	Default value	Default Size
Boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte
Short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

6) Define constructor & illstrate types of constructor with examples.

A) Constructor:-

A constructor is a block of code similar to the method. It is called when an instance of a class is created At the time of calling a constructor memory for the object is allocated in memory. every time an object is created using the new() keyword, atleast one constructor is called.

There are two types of constructors in Java

→ No arg constructor

→ parameterized constructor

① No arg (default) constructor

A constructor is called "default constructor" when it doesn't have any parameter

Syntax

```
<class-name>(){}
```

ex:-

```
class Bike1{
Bike1(){System.out.println("Bike created");}
public static void main(String args[]){
Bike1 b=new Bike1();
}
}
```

Output

Bike created

② parameterized constructor:—

A constructor which has a specific number of parameters is called a parameterized constructor

Ex:

```
class Student4{
    int id;
    String name;
    Student4(int i, String n){
     id=i;
     name=n;
     }
     void display(){System.out.println(id+" "+name);}
     public static void main(String args[]){
     Student4 s1 = new Student4(111, "Karan");
     Student5 s2 = new Student4(222, "Aryan");
      s1.display();
      s2.display();
      }
}
```

Output

```
111 Karan
222 Aryan
```

7) Explain about control statements in Java

A) Java compiler executes the code from top to bottom. The statements in the code are executed according to the order in which they appear. However, Java provides statements that can be used to control the flow of data such statements are called control flow statements.

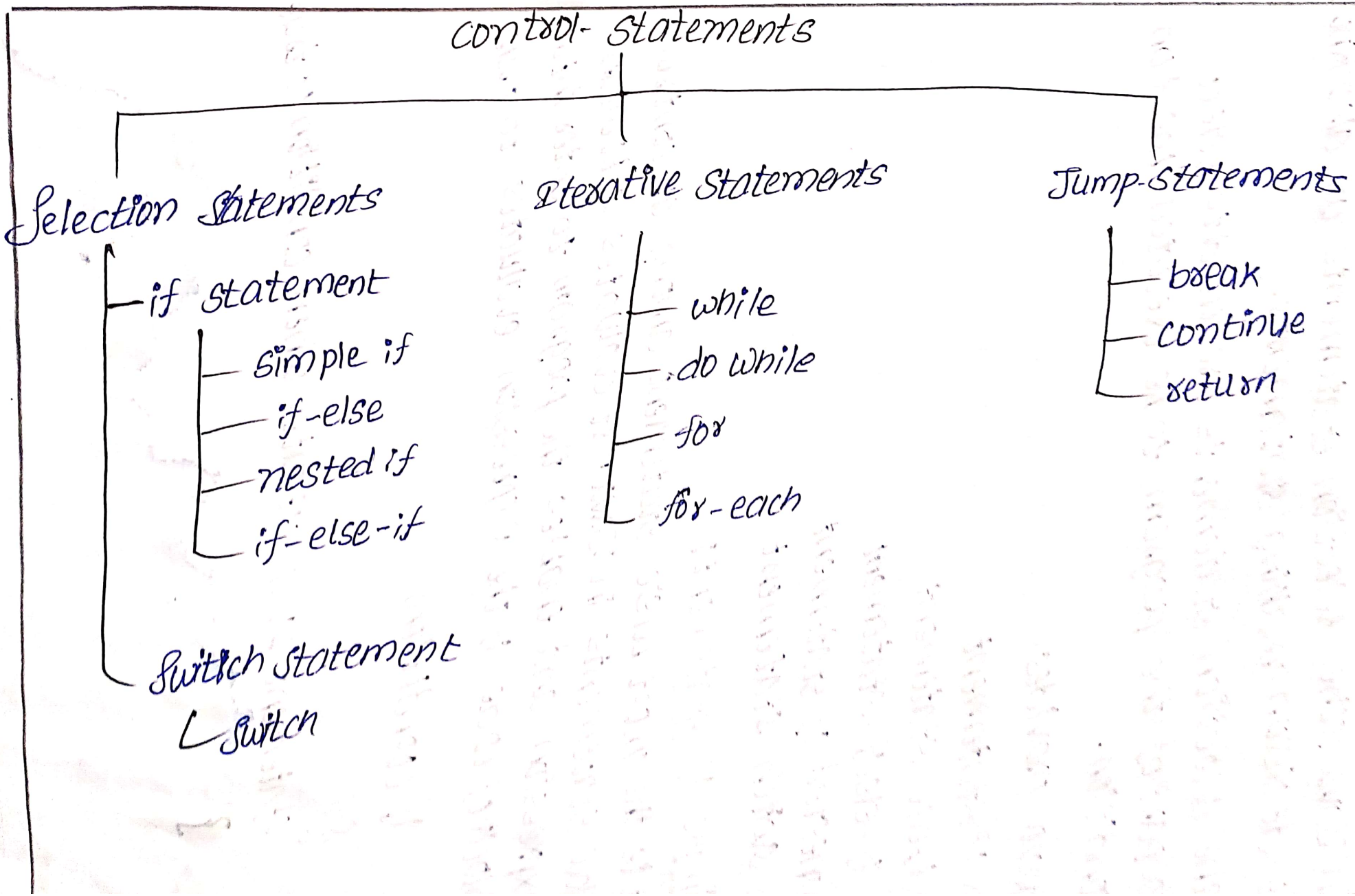
These are 3 types of control flow statements

① Decision making statements
- * If statements
- * switch statements

② Loop statements
- * do while loop
- * while loop
- * for loop
- * for-each loop

③ Jump statements
- * break statement
- * continue statement

Decision making statements:-
Decision making statements decide which statement to execute first and when. Decision making statements evaluate Boolean expression and control the program, depending upon result of condition provided



Selection control statements

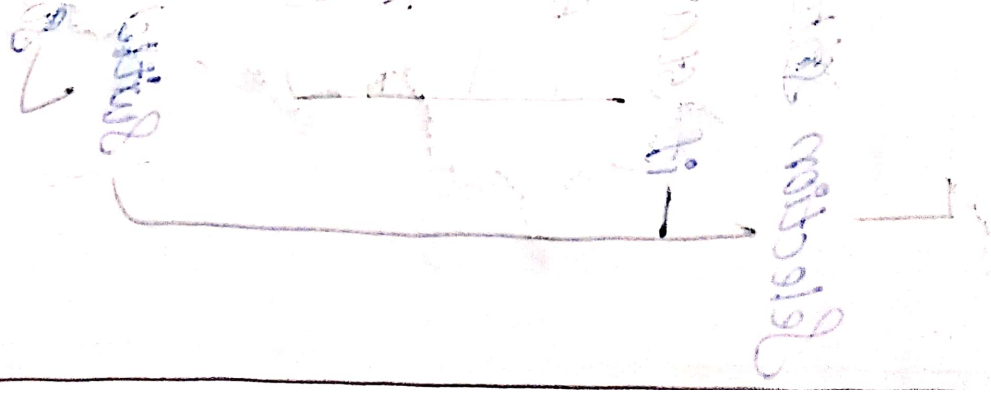
- In Java, the selection statements are also known as decision making statements.
- The selection statements are used to select a part of the program to be executed based on condition.
- Java provides.

→ if statement
→ if else statement
→ if-else if statement
→ nested if statement
→ switch statement

Simple if statement:-

It is the most basic statement among all control flow statements in Java. It evaluates a Boolean expression and enables the program to enter into block of code if the expression evaluates to true

```
if (condition)
{
statement 1; // executes when condition is true
}
```



ex for if

```
Public class ifexample {
Public static void main (String[]args)
{
 int age=20;
  if(age>18)
  {
  System.out.print("Age is greater than 18");
    }
  }
  }
```

if-else-Statement

The if-else statement is an extension to the if statement, which uses another block of code, i.e else block. The else block is executed if the condition of if-block is evaluated as false

```
if (condition)
{
Statement 1; // executes when it is true
}
else
{
Statement 2; // executes when it is false
}
```

```
Public class ifelse example
{
Public static void main (string[]args)
{
 int number =13;
    if(number %.2==0)
{
  System.out.println("even number");
  }
else{
    system.out.println("odd number");
     }
   }
}
```

Output

odd number

if-else-if

The if-else-if statement contains the if statement followed by mutiple else-if statements. In other words we can say that it is the chain of else statements that create a decision tree where the program may enter in the block of code where the condition is true

```
if (condition1)
{
Statement 1; // executes condition 1 is true
}
else if (condition2)
{
Statement 2; executes condition 2 is true
}
else
{
statement 2; executes all conditions are false
}
```

example program

```
Public class ifelseif example{
public static void main (string [ ]args){
    int marks =65;
    if(marks <50){
      system.out.println("fail");
  }
  else if (marks >=50 && marks<60){
      system.out.println(" D grade");
  }
  else if (marks >60 && marks<70){
      system.out.println("c grade");
  }
```

```
else if (marks >=70 && marks<80){
    System.out.println("B grade");
}
else if (marks >= 80 && marks<=90){
    System.out.println("A grade");
} else if (marks >=90 && marks<=100){
    System.out.println("A+ grade");
}else{
    System.out.println("Invalid");
}
}
```

Output

C grade

Nested if-statement

if nested if statements, the if statement can contain a if or if-else statement inside another if or else-if statement.

```
if (condition1){
statement 1; // executes when 1 is true
 if (condition2){
statement 2; executes when 2 is true
 }
 else{
statement 2; //executes when 2 is false
 }
```

example program

```
Public class Javanestedif example2{
Public static void main (string[]args){
    int age=25;
   int weight=48;
     if (age>=18){
       if(weight>50){
       system.out.print in c"you are eligible to donate
                                          blood");

     }else{
          system.out.printIn(" you are not eligible to
    donate blood");
       }
    }else{
       system.out.print In(" Age must be greter than 18");
       }
  }
  }
```

output

you are not eligible to donate blood

Switch statement

In Java, switch statements are similar to if else-if statements. The switch statements contains multiple blocks of code called cases a single case is executed based on the variable which being switched. The switch statement is easier to use instead if-else-if statements. It also enhances the readability of program

* The case variables can be int, short, byte, char string type is also supported since version 7 of Java.
* cases cannot be duplicate
* Default statement is executed when any of the case doesn't match the value of expression.

example program

```
Public class switchexample{
public ~~static~~:static void main(String[]args){
        int number=20;
        switch(number)
    {
    case 10:
    System.out.println("10");
    break;
```

```
case 20: System.out.println("20");
break;
case 30: system.out.println("30");
break;
default: system.out.println("Not in 10, 20, or 30");
}
}
}
```

Iterative control statements

- In Java, the iterative statements are known as looping statements or repetitive statements
- The iterative statement are used to execute apart of the program repeatedly
- By this reduces the size of the code, reduces complexity Increases speed.

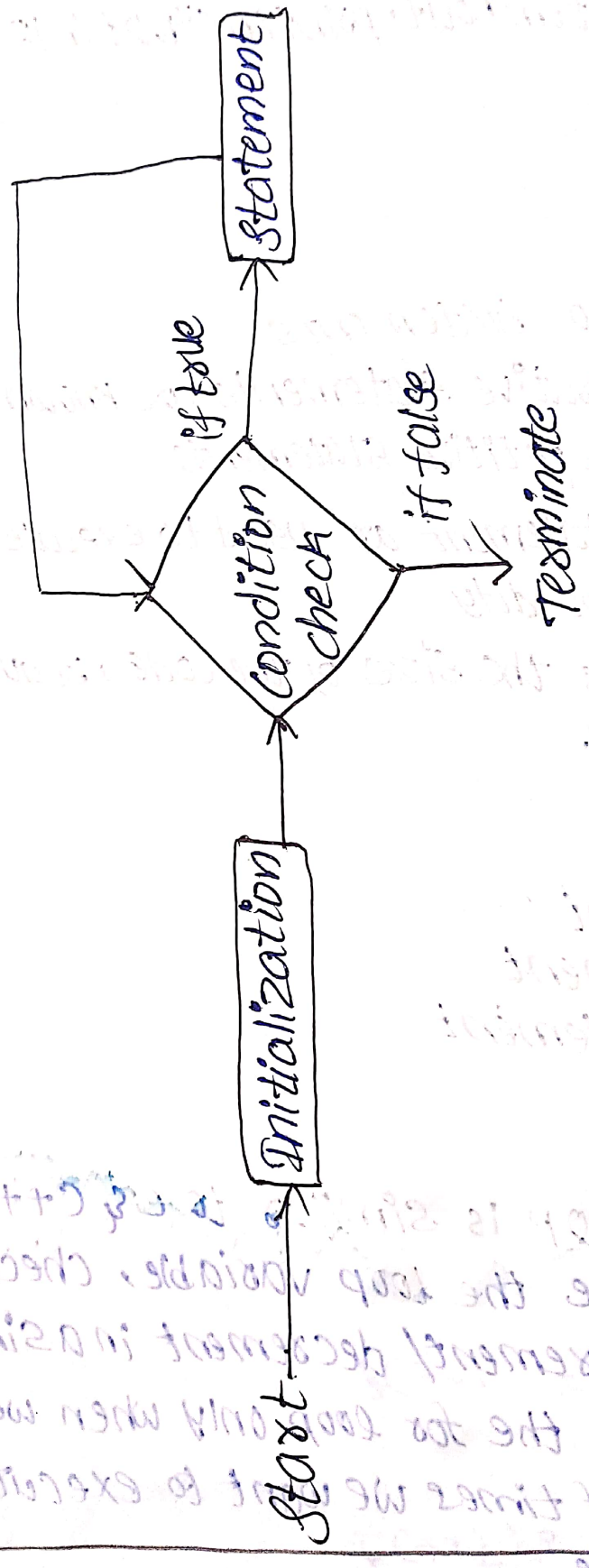
Java provides

→ for statement
→ while statement
→ do-while statement

for loop

In Java, for loop is similar to C & C++ It enables us to initialize the loop variable, check the condition & increment/ decrement in a single line of code. we use the for loop only when we exactly know the no. of times we want to execute the block of code.

for(initialization , condition, incoment/decrement)
{
statement
}



```
Public class calculation {
Public static void main(String[]args){
 int sum=0;
for (int j=1; j<=10; j++){
sum = sum+j;
}
System.out.println("The sum of first 10 natural
                    numbers is" +sum);

 }
 }
```

Output

The sum of first 10 natural numbers is 55

Nested for loop

If we have a for loop inside the another loop. it is known as nested for loop

Example program

```
Public class Nested for Example{
public static void main(String[] args)
{
// loop of i
for (int i=1; i<=3; i++){
// loop of j
for (int j=1; j<=3; j++){
```

```
System.out.println(i+" "+j)
} //end of i
} //end of j
}
}
```

for each loop

Java provides an enhanced for loop to traverse the data structures like array or collection. In the for each loop, we don't need to update the loop variable.

```
for(data-type var: array-name/collection-name)
 {
//statements
 }
```

example program

```
Public class calculation{
Public static void main(String[] args){
String[] names ={"Java","C","C++","python","javascript"
System.out.println("printing the content of array
                                    names:\n");

for(String name: names){
System.out.println(names);
}
}
}
```

Output

printing the content of the array name:

Java
C
C++
python
Javascript

while loop

If we don't know the no. of iterations we use while loop.

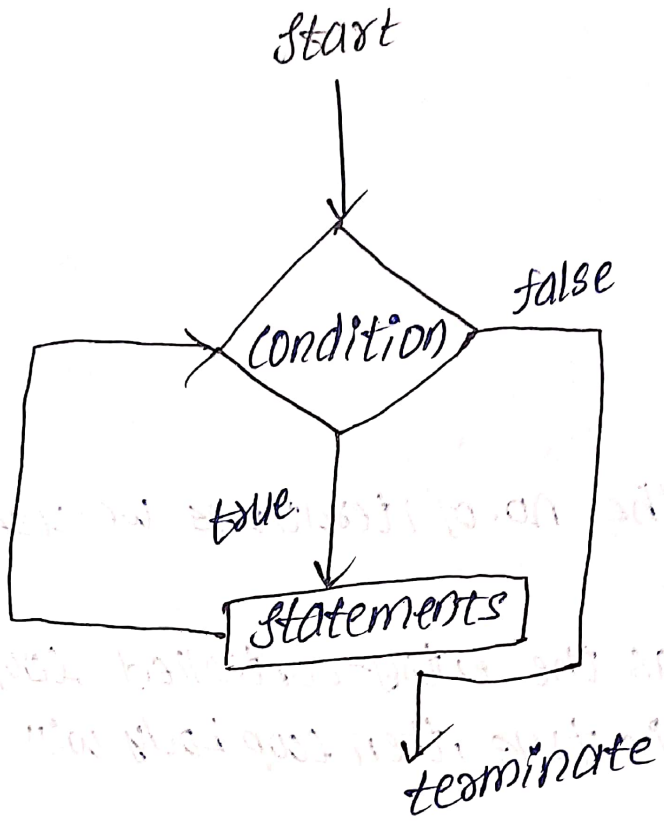
It is also known as the entry-controlled loop if the condition is true, then loop body will executed

```
while (condition){
// looping statements
}
```

```
public class Calculation{
public static void main(String[]args){
int i=0;
System.out.println("printing the list of first 10
                    even numbers \n");

while (i<=10){
System.out.print(i);
 i=i+2;
}
```

}
}

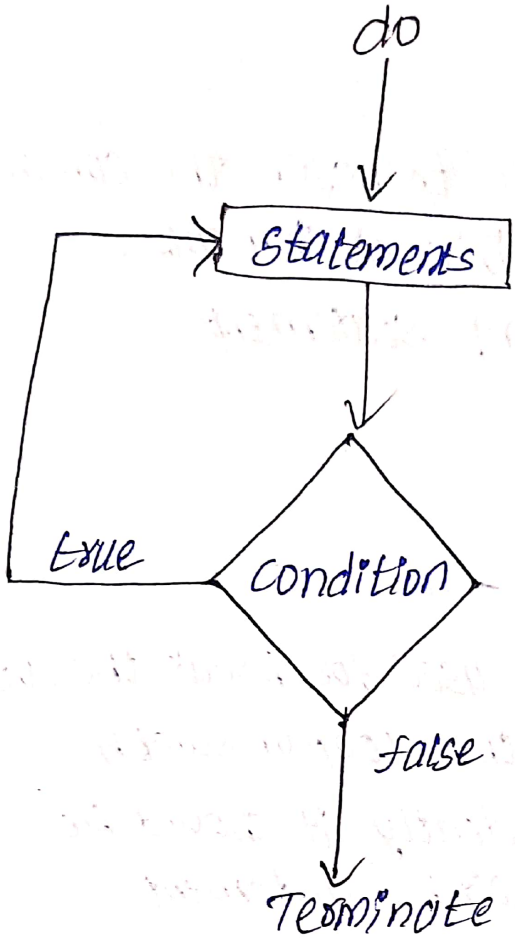


do-while loop

The do-while loop checks the condition at the end of the loop after executing the loop statements when the number of iteration is not known and we have to execute the loop at least once.

It is also known as the exit-controlled loop since the condition is not checked in advance

```
do
{
//statements
}while (condition);
```




```
Public class calculation{
Public static void main(string[]args){
int i=1;
System.out.println("printing the list of first 10
odd numbers\n");
do{
System.out.print(i);
i=i+2;
}while(i<10);
}
}
```

Jump Statements

Jump statements are used to transfer the control of the program to the specific statements.

There are two types of jump statements

→ break
→ continue

Break Statement

As the break statement is used for break the current flow of program & outside the loop or switch

It can't be used independently it should be written inside loop or switch statement

Program

```
Public class Break example{
Public static void main (string [] args){
for(int i=0; i<=10; i++){
system.out.println();
if(i==6){
break;
}
}
}
}
```

~~② Switch statements~~

③ What are access specifiers in Java? Explain with types

A) There are two types of access modifiers in Java. They are access modifiers modifiers and non access modifiers

→ The access modifiers in Java specifies the accesibility or scope of a field method constructor or class we can change the access level of fields constructors, methods and class by applying access modifiers on it.

There are four types of Java access modifiers

① private:- The access level of a private modifiers is only within the class it can't be accessed from outside the class.

② default:-
The access level of a default modifier is only within the package. If you do not specify any access level, it will be default

③ protected:
The access level of protected modifier is within the package and outside the package through child class If you do not make child class it cannot be accessed from outside package

④ public:-
The access level of a public modifiers in everywhere It can be accessed from within the class, outside the class, within the package and outside the package

* There are many non-access modifiers such as static abstract, sychronized, native etc...