Introduction to programming

Types of languages

- procedural
- functional
- object oriented

1) procedural :-

- specifies a series of well-structured steps and procedures to compose a program
- contains a systematic order of statements, functions and command to complete a task

2) functional

- writting a program only in pure functions i.e never modify variables, but only create new ones as an output.
- used in situations where we have to perform lots of different operations on the same set of data, like ML.
- first class functions? → Block of code we can use again and again.

3) Object-oriented

- Revolves around objects.
- code + Data = object
- Developed to make it easier to develop, debug, reuse, and maintain software.

datatype → [classes] → collection of all

Date

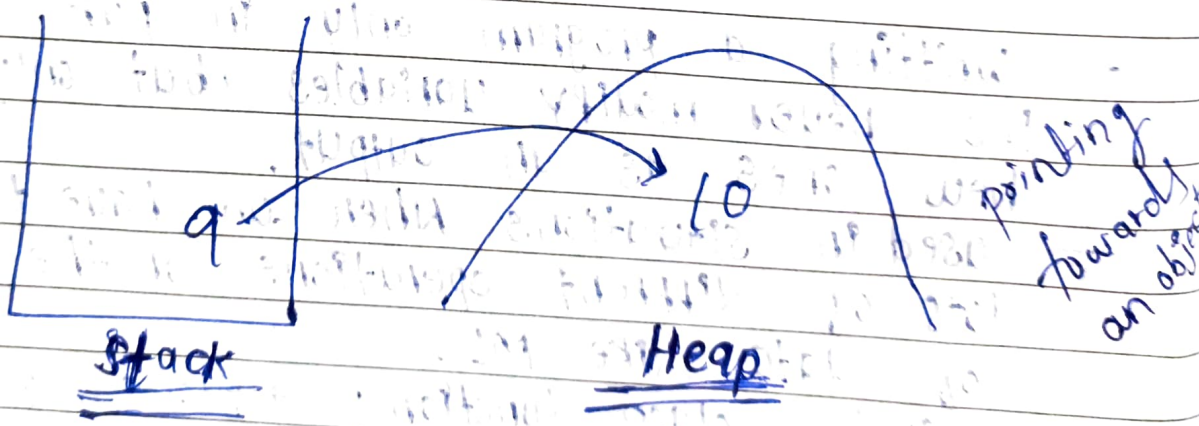
VS

Static and Dynamic language

- perform type checking at compile time
- Errors will show at Compile time.
- int a = 10
- Declare datatype before you use it.
- More control

- perform type checking at runtime
- Errors might not show till program is run
- No Need to declare datatype of variables
- Save time in writing code but might give error at runtime.

Memory management



one or more reference variable → a = 10 ← object

↓

Reference variable

a is point to 10

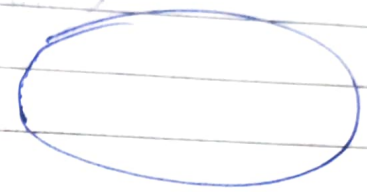
garbage collection :- object with no reference variable.

→ 10

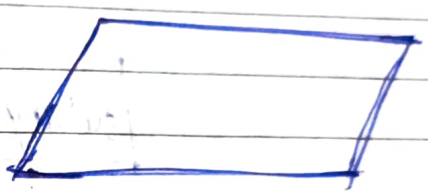
Remove.

Flowcharts

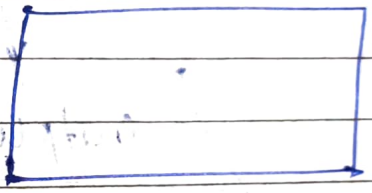
start / stop →



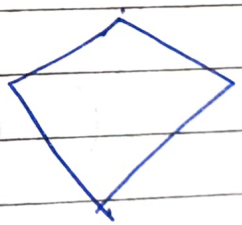
Input / output →



processing →

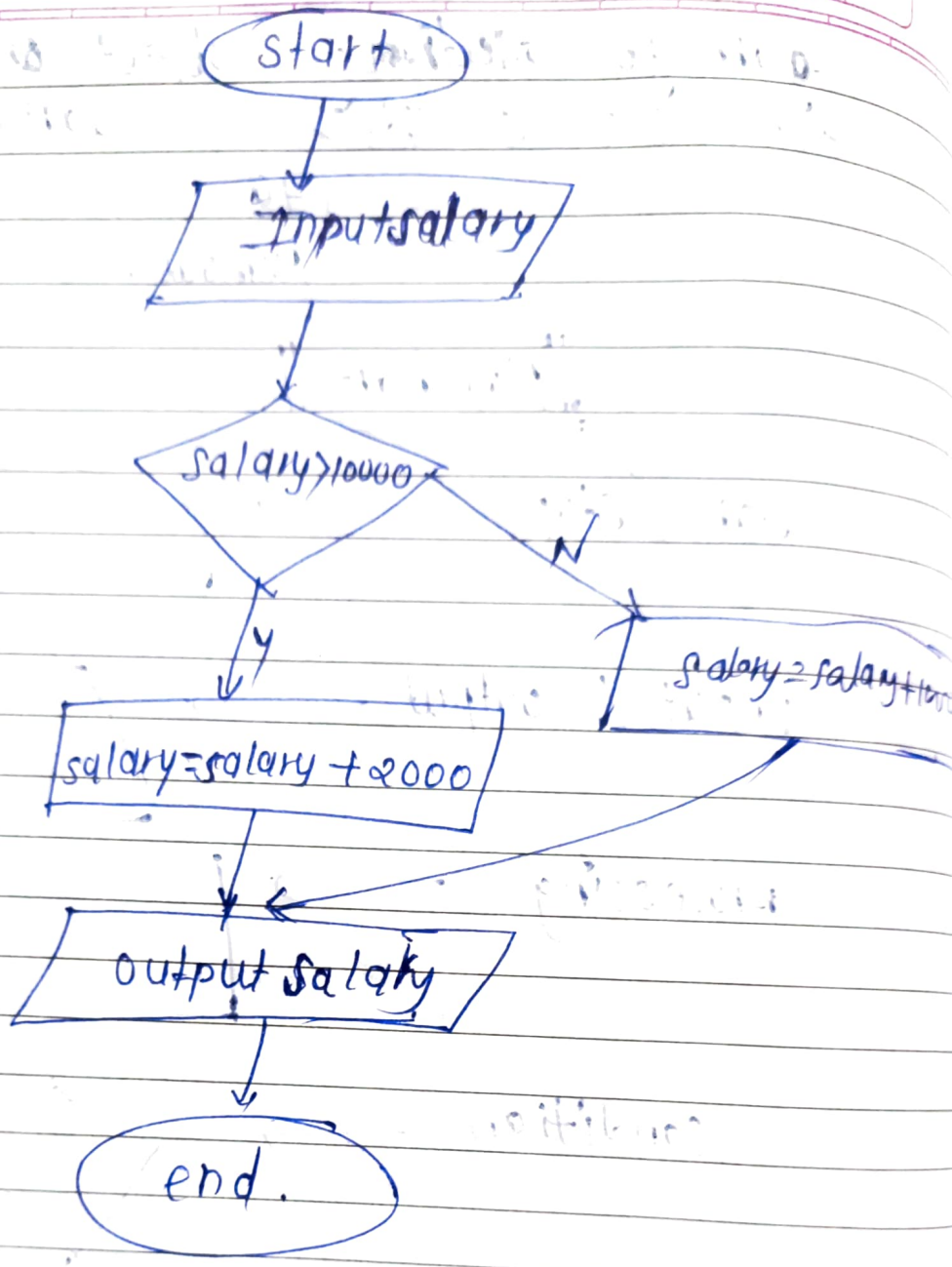


condition →



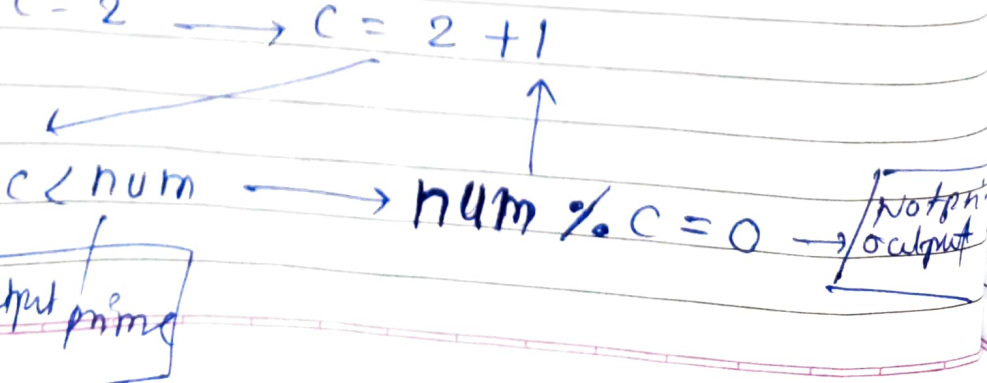
Flow direction of program. →

Q. take input of a salary. If the salary if greater than 10,000 add bonus as 2000, otherwise add bonus as 100

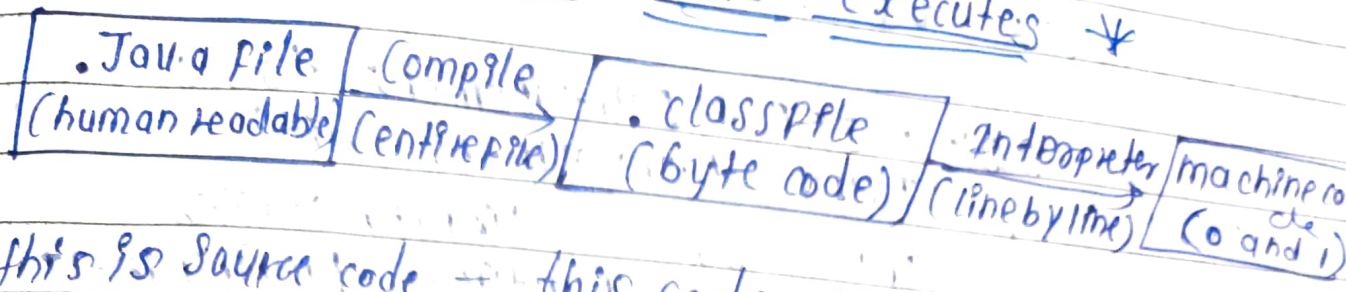


Input a number and print whether it is prime or not.

logic



* How Java code executes *



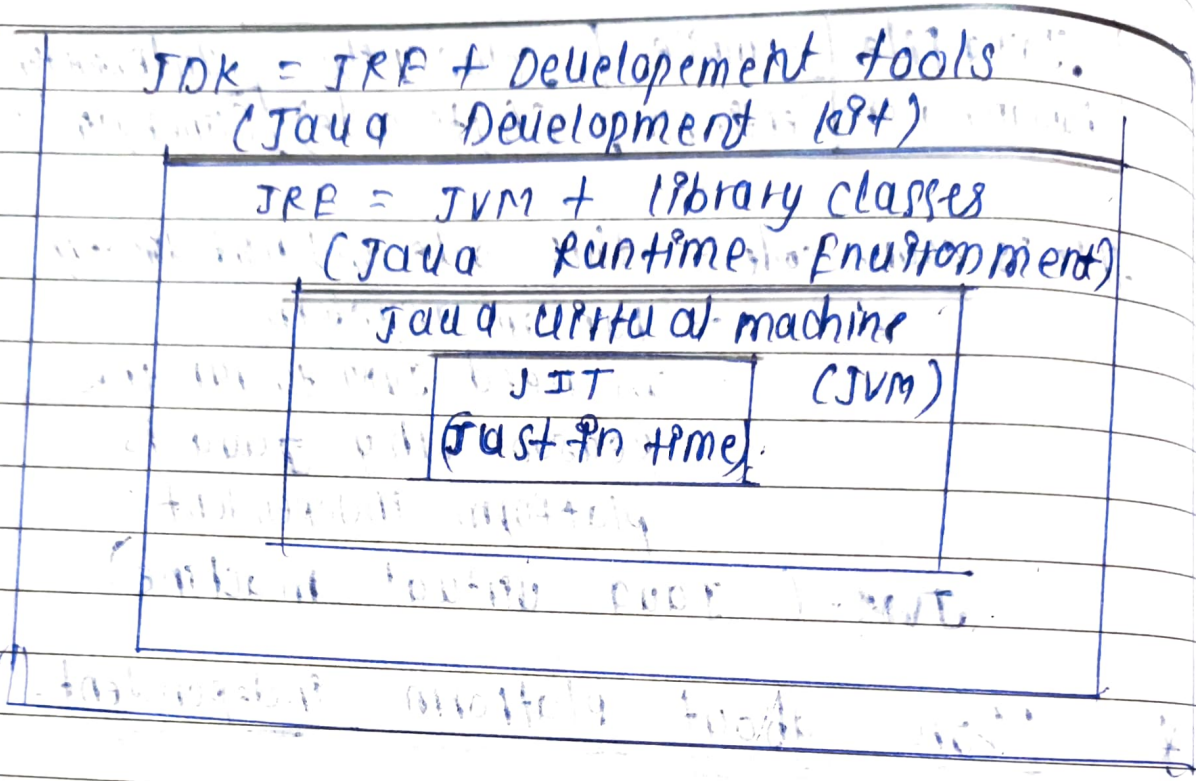
this is Source code → this code will not directly run on a system
- we need JVM to run this
- Reason why Java is platform independent.

JVM - (Java virtual machine)

* More about platform independent.

- It mean that byte code can run on all operating systems.
- we need to convert source code to machine code so computer can understood.
- Compiler helps in doing this by turning it into executable code
- this executable code is a set of instructions for the computer.
- After compiling c/c++ code we get .exe file which is plateform dependent
- In Java we get bytecode. JVM converts this to machine code
- Java is plateform - independent but JVM is platform dependent.

JDK Vs JRE Vs JVM Vs JIT



JDK :- → • Java → • class file in byte

- prouides enuironment to develop and run the java program
- It is package that includes
  - development tools - to prouides an enuironm to develop your programe
  - JRE - To execute your program.
  - a compiler - javac
  - archiver - Jar
  - docs generator - Javadoc
  - interpreter / loader.

JVM/JRE = Runtime

JRE.

- It is an installation package that provide environment to only run the programe
- It consists of
  1. developement technologies
  2. user interface toolkits
  3. integration libraries
  4. Base libraries
  5. JVM.
- After we get the class file, the next things happen at runtime
  1. class loader loads all classes needed to execute the program
  2. JVM sends code to byte code verifier to check the format of code.

(How JVM works) class loader

- loading
  - reads class file and generate binary data
  - an object of this class is created in heap memory.
- linking
  - JVM verifies the class file
  - allocates memory for class variables & default values
  - replace symbolic references from the referenced.

- Initialization

all static variables are assigned with their values defined in the code and static block

JVM contains the stack and Heap memory allocations

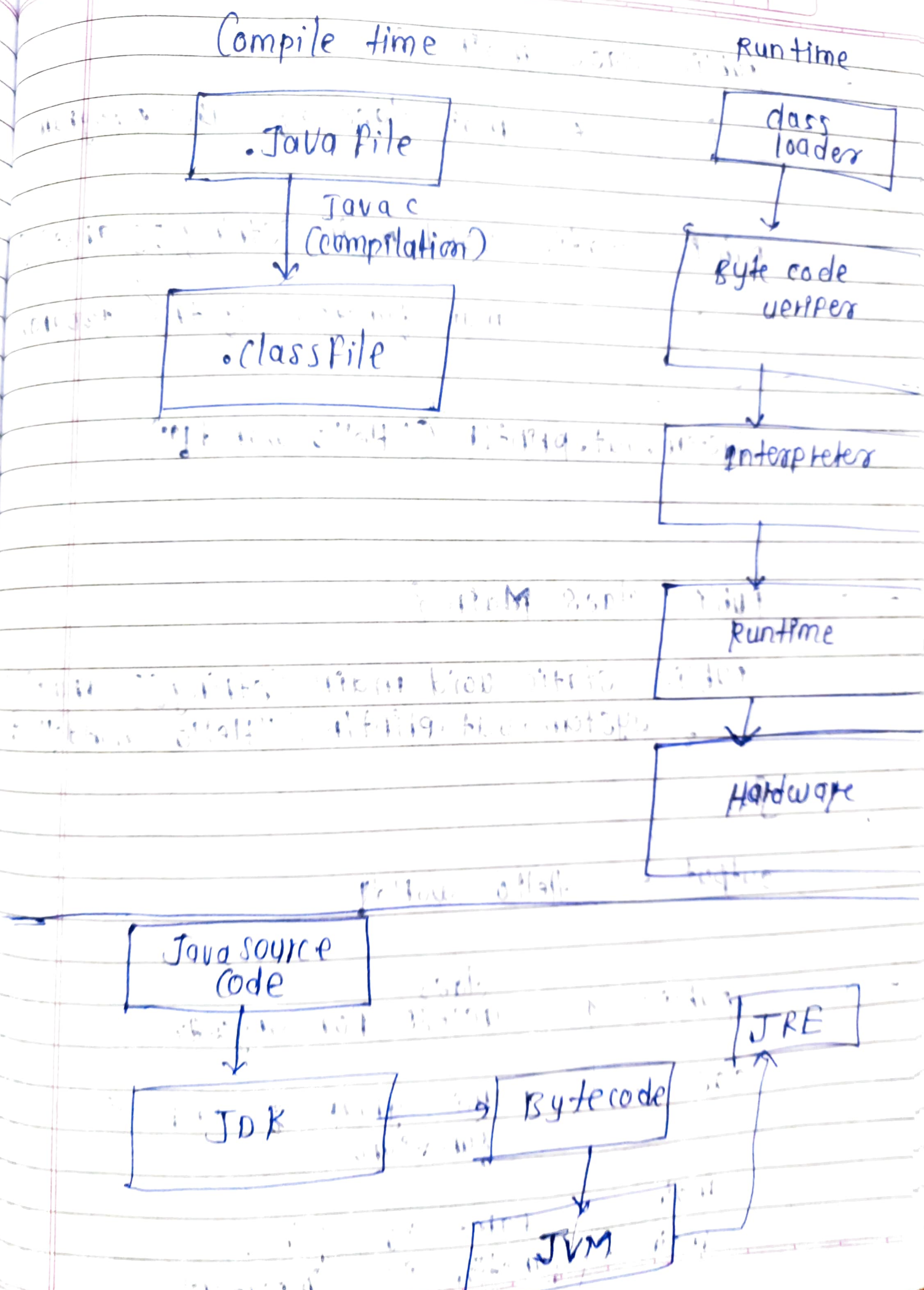
* JVM Execution

Interpreter

- line by line execution
- When one method is called many times it will interpret again and again

JIT

- those methods that are repeated
- JIT provides direct machine code so reinterpretation is not required
- makes execution faster Garbage Collector


Compile time
Runtime
.Java file
Java c
(compilation)
.class file
class loader
Byte code verifier
Interpreter
Runtime
Hardware
Java source code
JDK
Bytecode
JVM
JRE

- public class main {
  } ← main class of the program

- public static void main (String [] args)
  } ← main function of the program

System.out.println ("Hello world!")

→ public class Main {

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

output – Hello world!

→ public ← class access for anywhere

class ← name group of properties and functions

main – name of file.
main – function is important

public static void main

without programe are not run

static → we want to run this main function without class of demo (main) that's why static over here.

psvm ← shortcut

sout

println ← new line

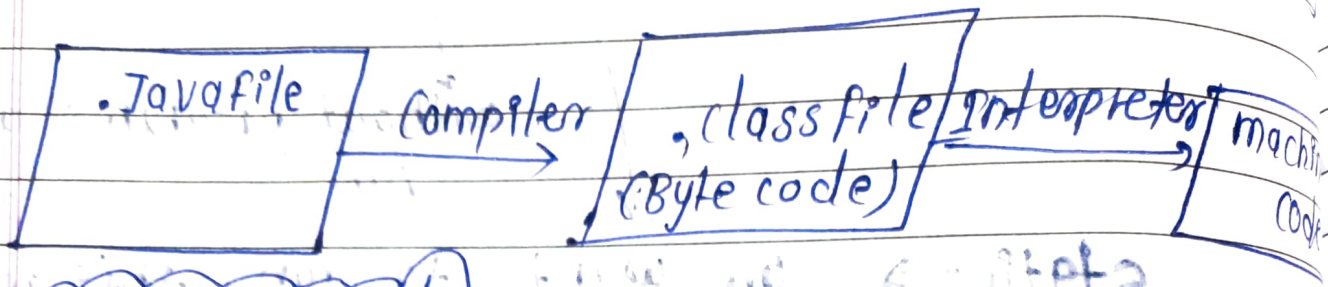
printstream ← class.

Introduction to Java

Q. Why do we use programming languages?

→ Machine only understood 0's and 1's for humans it is very difficult to instruct computer in 0's and 1's so to avoid this issue write out code in human readable language (programming language)

"Java is one of the programming langua



Source code

- the code written in Java in human readable and it is saved using extensi .Java
- This code is known as Source code

Java Compiler

- Java compiler converts the Source code into byte code which have the extension .class
- this byte code not directly run on system
- we need JVM to run this
- Reason Why Java is platform independent

Java Interpreter

- converter byte code to machine code i.e 0's and 1's
- It translate the byte code line by line to machine code.

class Main

↳ start with capital letter it shows it is class

public class Main

↳ class access for any where

First Java Program

Structure of Java file

"Source code that we write will be saved using extension .java"

- Every things written in .java file must be in classes or we can say that every file having .java extension in a class
- A class with same name as file name must be present in Java.file

first alphabet of class name can be in upper case. It is naming convention of class name however, It is not compulsory to do so.

- class which is having same name as file must be public class.
- A main function / method must be present in this public class, main is a function from where the program starts.

Converting .Java to .class

- using Javac compiler we can convert .Java file to .class command to convert .Java to .class

Javac and .Java file name

let the name of .Java file is main, do the command to convert .Java to .class is

```
Javac Main.Java
```

- Aboue command create a .class file (Main.class) which Contains Bytecode.

.Java file —— Javac Main.Java ——▶ .class file

Running the program

By using java and name of file we can run the program

- Command > Java Main

Hello World Program

```
>> public class Main {
    public static void main (string [] args)
    {
        System . out . println ("Helloworld");
    }
}
```

IMP - IMP - IMP - IMP - IMP - IMP - IMP

1. public (in first line) :- public is an access modifier which allows to access the class from anywhere.

2. class :- It is name of group of properties and functions.

3. Main :- It is just the name of class as same as name of file

4. public (In second line) :- It is allow to program to use main function from anywhere.

5. static :- It is a keyword which helps the main method to run without using objects.

6. void :- It is keyword used when we do not want to return anything from a method / function

7. main :- It is name of method.

8. String [] args :- It is command line arguments of string type array.

9. System : It is final class defined in Java.lang package.

10. out :- It is a variable of printstream type which is public and static member field of the system class.

11. println :- It is a method of printstream class, It prints the arguments passed to it and adds a new line. print can also be used here but It prints only arguments passed to it. It do not adds a new line.

What is Package ?

→ It is just a folder in which Java files lies

→ It is used to provide some rules and stuff to our program.

Permitive data types

- permitive data type are those data type which is not breakable.

EX :-

String is not permitive data type so we can break this data type into char.

i.e. String "kunal" can be divide into

'k' 'u' 'n' 'a' 'l'

But permitives data type are not breakable.

we cannot break char, int etc.

list of primitive data type in java are :-

Data type	Description	Example
int	int is used to store numeric digits	int i = 26;
char	character is used to stored characters	char c = 'A';
float	float is used to store floating point number	float f = 98.67f;
double	double is used to store large decimal numbers	double d = 45676.58975
long	store numeric digits which is not able to stored in int	long l = 15878256 7789
boolean	it is only stores two values true or false	boolean b = false;

In float and long we have used f and l denotes that the number in the variable is float or long type. If we do not use this Java consider float value as double and long value as int

- Literals :- It is synthetic representation of boolean character, string and numeric data
  EX - int a = 10;
  Here 10 is called literal

- Identifiers :- name of variable, methods, class, packages, etc are known as identifiers.
  EX int a = 10;
  Here a is identifier

* Comments in Java

Comments are something which is written in source code but ignored by compiler

- Two types of comment

1. Single line comment :- used to comment down a single line (// is used for it)
2. multiple line comment /* ..... */

Input in Java

We have scanner class available in Java.util package to take input.

To use this class we have to

1. Import Java.util package in our life
2. Create object of the Scanner class
3. Use that object to take input from the keyboard

Syntax

```
import Java.util.scanner;
public class main{
    public static void main (string[] args){
     scanner input = new scanner (system.in);
    }
}
```

→ 1. scanner = It is a class required to take input; it is present in Java.util package.

2. input :- It is an object that we are creating to take input

3. new :- It is keyword used to create an object in Java.

4. System.in :- system is a class and in is variable that denotes we are taking input from standard input stream (i.e keyword)

* Int Input :- nextInt() is a function used take input of Int

⟹ Syntax :-

```
Scanner input = new Scanner(system.in);
Int rollno = input.nextInt();
```

* float input :- nextFnt() is a Function used to take input of float

⟹ Syntax :

```
Scanner input = new Scanner(system.in);
float marks = input.nextFloat();
```

* String Input :- Two ways to take string input

1. using next() method :- It will take one word input till a space occurs

⟹ Syntax :-

```
Scanner input = new Scanner(system.in);
Sting s1 = input.next
```

2. using nextline() method :- It will take all string input including space.

Syntax :-

```
Scanner input = new Scanner(System.in);
String s2 = input.nextLine();
```

* Sum of two numbers.

```
import java.util.Scanner;

public class sum {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter first number");
        int num1 = input.nextInt();
        System.out.print("Enter second number");
        int num2 = input.nextInt();
        int sum = input.nextInt();
        int sum = num1+num2;
        System.out.println("sum = "+sum);
    }
}
```

output :-

enter first number 70
enter second number 80
sum = 150.

Date

* Type Conversion

When one type of data is assigned to another type of variable an automatic type conversion will take place under some condition.

Conditions :

1. Two type should be compatible.
2. Destination type should be greater then the source type.

* Type casting

When we convert one type of data to another type is known as type casting

Ex. int num = (int) (67.564f)

* Automatic type Promotion in Expression

while evaluating expression the intermediate value may exceed the range of operands and hence the expression value will be promoted.

There are some Condition For type Promotio

1. Java automatically promotes each type, shor or char operand to int when evaluating an expression.
2. If one operand is a long, float or double the value the whole expressio is promoted to long, float or double respectfuely.

ex

```
byte a = 40;
byte b = 50;
byte int c = 100;
int d = (a * b)/c;
System.out.println(d);
```

~~While loop~~

While Loop

```
public class Basic{
    p s v m {
    int a=10;
    if (a==10) {
        S.out.(  );
    }
}
```