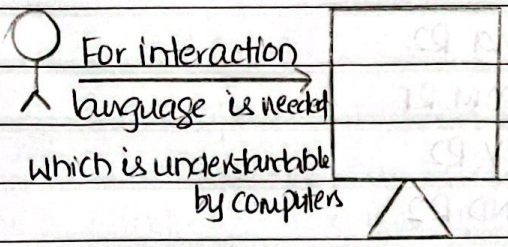
5/8/23

Programming Language

The Language used to communicate with the machines or computers by programmers is called a programming Language.



Programme → Set of instructions used to perform a specific task

Language → Mode of communication used to share ideas and opinions with eachother.

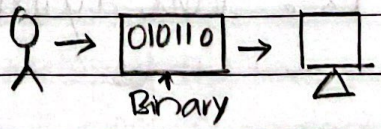
Types of Programming Languages :

1. Low Level Programming Language
2. Mid Level programming Language
3. High Level programming Language.

Low Level programming Language - They are machine understandable which is in the form of binary Language (0's and 1's)

Binary Code - Which is understandable by the machines which is in only 0's and 1's

* Translator not required
* Easy to Execute → Advantage.



Dis-advantage - programmer's can't understand easily and remember easily.

Mid Level programming Language - Assembly Level Language

There are some already pre-defined words and symbols which are called Mnemonics. This is partially understandable by humans and computers.

Ex : ADD R1 and R2
SUB R2 FROM R1
DIV R1 BY R2
MUL R1 AND R2

* Translators are needed called Assembler

High Level programming Language -

Any Language which is easy to read, write and understand by the programmers are called High Level Language. Here we will use general english components.

* Here Translator is required known as compiler.

Advantages → High Level Languages are easy to understand.

Disadvantages → Proccessor runs the code, it will consume time.

Ex : Java, Python, C#, C++, Pearl, C

7/8/23

Platform : The place where operating system runs.

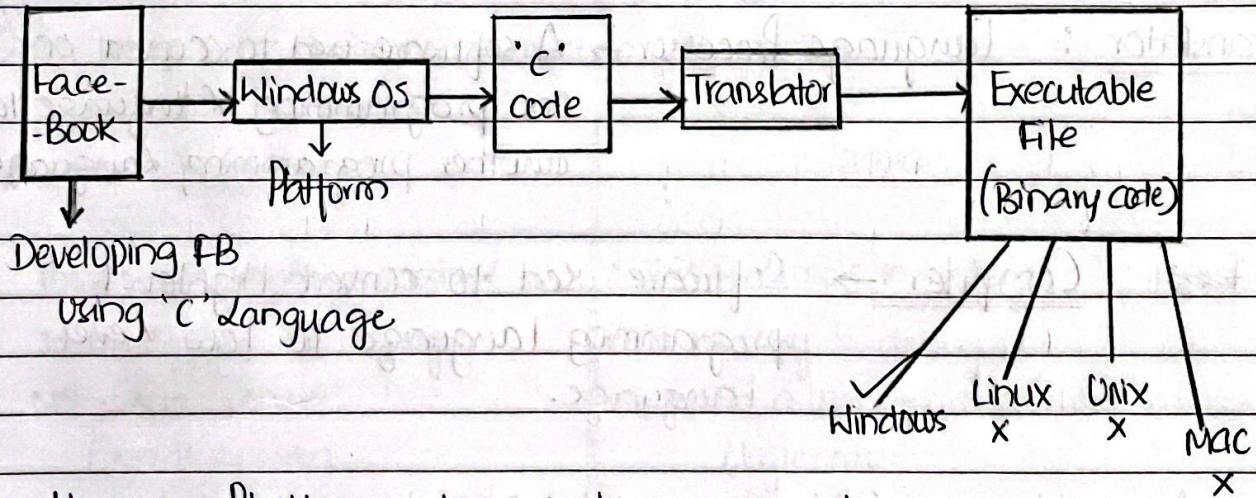
Platform Dependent platform Independent

Operating system : Software runs on a computer, it is going to manage our computer memory and processors

Ex : Windows, Linux, Unix, Mac

Software : Set of programmes used to do specific task.

Platform Dependent : An Application developed by using one platform is able to run on the same platform. Ex : If we develop on Windows, it will run on windows only.

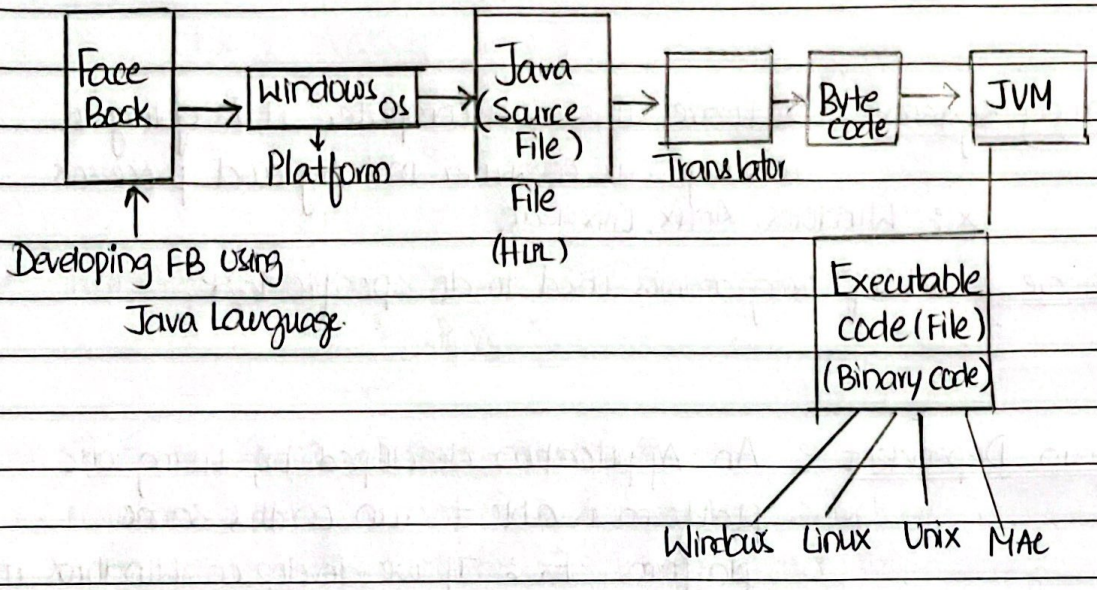


Here in Platform dependent we can not run on linux, Unix and Mac as we developed the application in windows it will work on windows only.

Ex : C, C++, C Embeded.

Platform Independent : An Application built in one platform can be run in multiple platforms.

Ex : If we develop on windows, it should run on windows, Unix, Linux Mac etc



Here Java is a platform Independent.

Ex : Java, Python, C#.

Translator : language Processor → A software used to convert one one programming Language to another programming Language.

~~Ex:~~ **Compiler** → Software used to convert Highlevel programming language to low Level language

Assembler → Software used to convert Midlevel language to low level programming Language.

Memory : A space where we will store the data.

1) Permanant Memory → Where we can store data Permantaly

Ex → ROM (Read only Memory).

2) Temporary Memory → Where we will store data temporarily

Ex : RAM (Random Access Memory)

File : A container where we will store written code or data.

Folder : Where we will save a set of files.

1) Source File → Where the instructions are written in high level languages by Programmers which generates a executable File.
(.Java)

2) Executable File → A file which is created for executing for the application, this is a file which is not created by a programmer. It will be in either high level language or low level language
(.Exe file)

8/8/23.

Why Only Java over python, C, C++?

C, C++	Java
* These are complex in nature	* This is simple in nature
* It is difficult to debug	* It is easy to debug.
* Not more secure (No Byte code)	* Java is a very much secured language. (Byte code)
* It is platform dependent	* Java is a platform independent language.
* It doesn't support multi thread operations	* Java support multi threading.

Python	Java
* NO compiler and Interpreter.	* Java is having compiler and Interpreter
* Not efficient and safer compared to Java.	* Efficiency and safer.

* Java *

- It is a high-level programming Language.
- It is developed by James Gosling and marketed by Sun Micro Systems. in the year 1995. (Now owned by Oracle)

1991 ⟶ Java only for Electronics
like TV, Tostares

They had a goal to develop Java ⟶ Simple, portable, highly reliable to the users.

As C, C++ had limitations, they want to overcome these limitations and came across Java which is Simple, portable and highly reliable.

The first name of Java is "OAK", as OAK is already popular using by other companies he changed the name to Green Talk still he is not satisfied and finally he finalized and name while at the coffee shop as 'JAVA' which is meant a coffee bean. JAVA is a not an acronym (which doesn't have any abbrevation) In 2010 it is owned by Oracle Company.

Features of Java

- Java is Very Simple language, we can study it from the scratch. Syntax in Java is easy.

- Java is a platform independent, which means Java programmes we can run in any other Operating systems.
  Ex: Windows, Linux, Unix etc.

- Java is an Object Oriented Language.
- Java is extensible (we can easily integrate with any other programming Languages to develop an application) Ex: C, C++, Python.

- Java is Robust → It can handle any huge applications and no virus will harm it. (Error, Exceptions → Handled by 1) Garbage collectors
  2) Multithreading)
  * Garbage Collector (Automatic) → Going to collect de-refference (unwanted) objects automatically where we can save the memory. (Memory managment is good)

- Java supports all OOP's Concepts ① Abstraction
  ② Encapsulation
  ③ Inheritance
  ④ Polymorphysim.

- Java is more secure because of the Byte code.

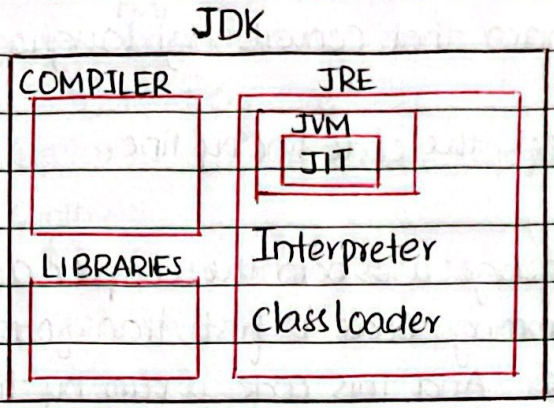
WORA - Write Once Run Anywhere.

* Java is a platform Independent
* JVM is platform Dependent.

9/8/23

JDK (Java Development Kit)

It is a software used to run Java programmes.



Compiler - It is a translator which is used to convert HLL to LLL and also it will check for the error and if it finds any error (bugs) (debug) it will through a error message, otherwise it will provide output.

Libraries - It's a collection of pre-defined interfaces, API's and classes.

API - Application Programming Interface.
Collection of essential resources, softwares which is used to develop a Java application.

JRE - Java Runtime Environment - It is a software which will provide an environment to execute Java program

JVM - Java Virtual Machine - Its going to convert Byte code to LLL (MLL)
- Its going to allocate memory for the file which is generated (Bytecode)
- JVM is platform dependent.

JIT - Just In Time - It is a tool which helps JVM to optimize the Bytecode (JIT will be present inside JVM)

Classloader - Loading class from a harddisk to execution area.

Interpreter - Its a program that converts high level language to low level language.

- It will check the code line by line.

Java programming language uses both the Compiler and an Interpreter because the source programming code is first transformed into binary programming code. And this code is run by the JVM, which is having a interpreter which converts high level language into a machine understandable language. (low level language)

GUI (Graphical User Interface) - It is user-friendly graphical or visual experience build for Java applications.

Ex: Cursor, scrollbars, icons

Command line Interface (Command prompt) - Communicating with the system using commands, which are already defined. (Pre-defined Commands)

To open → * Search as CMD in search bar and you can see Command prompt.

* Click Windows+R and type cmd in the search bar, Command prompt page will be opened.

Current/present -

*- Working Directory : A folder in which controller is present currently.

10/8/23 Commands Syntax

1) mkdir/md → It is used to create a new folder inside a current working directory.

Syntax → mkdir/mk ↓ space qspiders

c:/users/santh>

↓ ↳ Current working Directory.

Root Folder → It is the source from which the path of other folders starts.

Ex: c: (c drive),

2) cd → It is used to change the working directory. (or) It is used to move controller from one folder to another folder.

Syntax → cd ↓ space Qspiders.

3) mkdir ↓ space Folder1 ↓ space Folder2 .....

4) To create a file inside a folder

Syntax is echo ↓ space content > filename.exe (.exe-extension)

5) dir → directory → It is used to list of files and directory present inside present/current working directory.

Syntax is dir

→ cls → clear screen

→ rmdir → Remove the particular folder

1. MKdir/md – create new folder (make directory)
   Syntax - mkdir foldername

2. cd – change directory/Folder

   Syntax – cd Foldername

3 cd.. - Goto previous Folder/ Directory

   Syntax - cd..

4. echo – create a File inside the directory/ Folder

   Syntax – echo content > filename.exe

5. dir - To see the files in the present working directory.

   Syntax - dir

6. cls - clear the screen

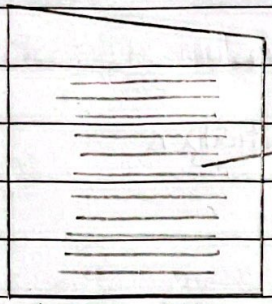
7. rmdir – remove the directory/Folder

8. del – To remove the file inside the folder

   Syntax - del Filename.

11/8/23

Structure of Java.



Source.file
(xyz.java)

- Class
- Interface.

(We cannot write anything in source file, we can define using class and Interface)

Class - It is a keyword (predefined word). It is always have lower case letters and this class is a compiler known words. class is used to create class block.

Syntax of the class is :

Class Block {

```
class    classname   <- We need to provide the name for a class and first letter should be in uppercase
{        <- opening of class block

}        <- Ending of classblock
```

Inside classblock we have to define declaration statement and Initializers.

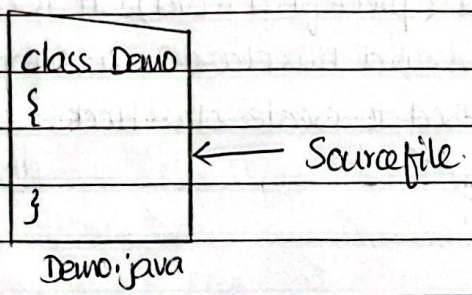
```
class classname
{
Declaration statement  ---> Information
Initializers  ---> Things happening in the beginning.

}
```

Declaration statement — Variables, Methods, Class

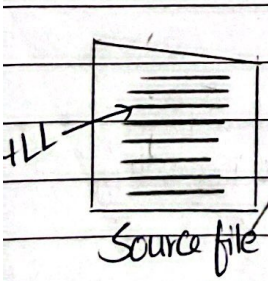
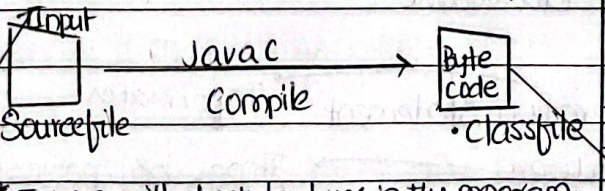
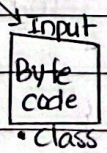
Initializers — Static Initializers, Non-Static Initializers, Constructor.

* According to Industrial standards file name should be same as a classname but not mandatory. (Easy to remember)



* Creating a program in Java -

1. Creating Java source file
2. Compile
3. Execute

Creating Java sourcefile	Compile	Execute
Editor - Eclipse IDE, Notepad	Java provides an software called Javac (Java Compiler)	For execution Java provide one software JRE
	 * Javac will check for bugs in the program. If it find any bugs it will a debug message to the programmer. It will create a .class file	Java Runtime Environment → It provides an environment to execute after compilation 
Sourcefilename.java	Javac sourcefilename.java	Java ClassName

Bytecode : It is an intermediate code which is neither understandable by human nor by computers.

* It will be inside .class file
* .class file will generate once after compilation successful.

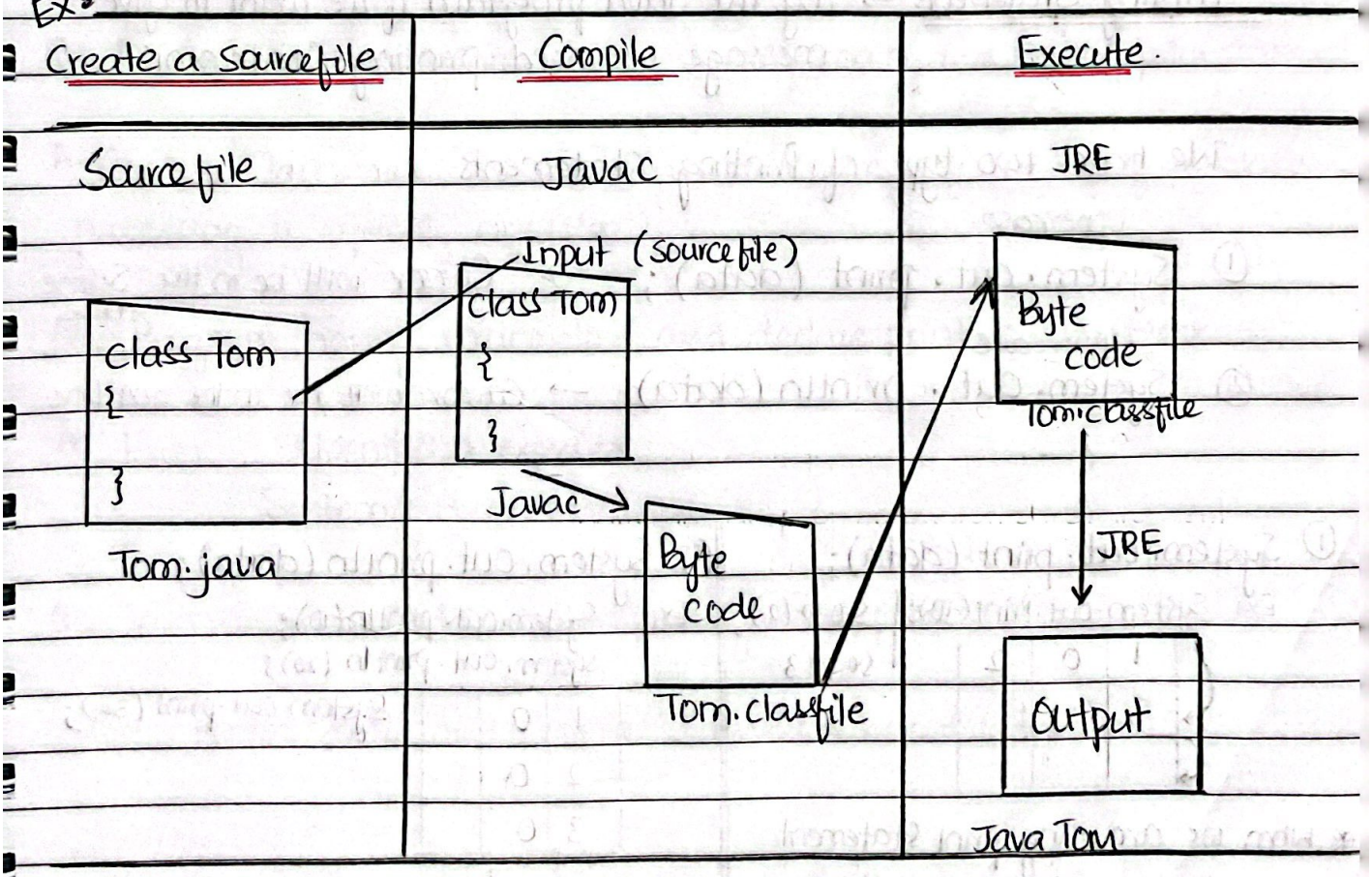
To Compile

Syntax — javac Sourcefile.Java
(name)

To Execute

Syntax — Java classname.

Ex:



12/8/23

* Execution of Java Program always start with main method.

* As soon as we execute a program JVM will go on search with the main method.

Syntax of Mainmethod -

lowercase → public static void main (String[] args) ← uppercase
{

}

Printing Statements → By the Java program if we want to give message we need printing statements.

We have two type of Printing statements.

① System.out.print(data); → Cursor will be in the same line

② System.out.println(data); → Cursor will be in the next line

① System.out.print(data);

Ex: System.out.Print(10); | S.O.p(2), S.O.p(3)

	1	0	2
	3		

* When we are using print statement cursor will move to the right side of same line until space covers. Once it covers the space it will go to new line.

* Data is Mandatory

② System.out.println(data);

Ex: System.out.println(10);
System.out.println(20);
System.out.print(30);

1	0	
2	0	
3	0	

* When we are using println statement cursor will go to nextline after printing the data.

* Data not mandatory

* Directly we cannot declare the print statement without the main method inside the class block. because we knew that we have to declare declaration statement and initializers.

* Q1. Execute Java program without main method and write the error message.

A. Compilation sucessful and showing runtime error as -
A - Main method not found in class NOmainmethod, please define the main method as : public static void main (String[] args)
(or) a JavaFx application class must extend javafx.application.Application.

Q2. Execute Java program giving uppercase in class and write the error.

A. Error - class, interface, enum or record expected

Q3. Execute Java program without having main and having print statement

A. Error - Class Name, 'PrintWithNOMainMethod', are only accepted if annotation processing is explicitly requested

Q4. Program having mainmethod and declare print in class block

A. Error - <identifier> expected
System.out.print(HELLO);

Q5. Java program Print() without passing data. println()

A. Printed data and cursor came back to the terminal.

14/8/23.

Token – Smallest components or elements present in any programming language.

Ex: WELCOME → Here we can say the alphabets are small tokens to develop a word 'WELCOME'

Tokens – 1. Keywords
2. Identifires
3. Literals

} Types of Tokens in Java.

Keywords – Pre-defined words which means already existed and these are compiler known words and always be in lowercase.
- keywords also called as Reserved word. (We can't able to use that keywords for any other purpose.

Ex: class, if, else ....

Rule for the keyword → keywords are written in lower case
Ex: class – NO compile time error
CLASS – Compile time error

Identifires – Identifiers are the names giving to the components of a Java program.

Identifiers — Method
Variables
class

Ex:

```
class sauthi        ← Identifier
{
    public static void main(String[] args)    ← Identifier
    {                  ↑ Keywords
    }
}
```

Rules for the Identifiers :

1. Identifiers should not start with numericals.
2. Only $ and _ (underscore) are the special characters that are allowed

   Ex : classname → $Demo → Compile Successful.
   → @Demo → Error

Numbers can be used in between and at the end of the classname Special characters except $ and _, we can used in between or at the end of the classname.

3. Keywords cannot be used as Identifiers.

Conventions – These are Industrial standards programming culture a program should follow.

Note : Compiler won't check for the conventions.

Ex: class ClassName → Always the first letter of the classname should be in Uppercase. If you are using two names they should be in upper camel case Ex : 'SantoshRamesh.'

Conventions	Rules
* Conventions are not mandatory. Its just an Industrial standards that we have to follow.	* Rules are mandatory.
* Compiler won't checks for the conventions	* Compiler will check for the Rules
* Compile time successful	* Compile time error

Literals – Literals are the data written by the programmer in the program.

Ex:
```
class Demo ——— Identifier
{
public static void main (String[] args)
  {
   System.out.println ("HELLO"); ——— Literal
  }
}
```
Keyword

output — HELLO → Information or data in the program.

Types of Literals :

1) Number Literal
2) Characterstics Literal
3) Boolean Literal
4) String Literal.

15/8/23 1) Number Literal – The program in which programmer is passing the data in the form of Integers (numbers and decimals) is.

numbers – 0, 1, 2, 3, -4, -5 etc

decimals – 10.2, 12.5 etc.

Ex:
```
class Santhi
{
 Public static void main (String[] args)
{
 System.out.print ("1"); → Number literal.
}
}
```

2. Character Literal - These are enclosed in Single quote and length of the character is only 1 - either number, alphabet or character.

Ex: 'A' '2' '#' 'AB', 'JA'

Programmer passing character data in the program.

3. String Literal - Programmer passing the data/characters in double quotes. Set of characters in double quotes is String

For

Ex: "SANTHI", "1AB", "$#_", "1"

```
class Isha
{
  Public Static Void main(String[] args)
  {
    System.out.Print("Welcome");
  }
}
```

String.

4. Boolean Literal — Programmer pass the data as TRUE or FALSE. Used to represent logical data.

There are two logical data's

① Logical High - TRUE
② Logical Low - FALSE

} Keywords

TRUE → It is a keyword represents logical High

FALSE → It is a keyword represents logical low.