DATE PAGE 2

Introduction to Java

1] What is Programming :-

- Programming is set of instructions that tell computer how to perform task.
- Programing can be done using variety of computer programming languages like, C, C++, Java, JS, Py. etc.

2] Category of Programming languages :-

- Programming languages can be categorised into
  1] High-level language → Human understandable langage
  2] Assembly language. → consist of binary characters, expressed in more human readable form
  3] machine level or low level language → It is made up of binary numbers or bits that m/c can understand.

3] Compilation VS Interpretation :-

- Compilation :- It is a process to convert high level (C, C++, Java) prog. lang. into m/c language (all at once) that computer can understand. The software which performs this called compiler
- Interpretation :- It is performed by an interpreter which (JS, Python, Ruby) directly executes instructions written in programming language w/out previously converting them to an object code or machine code. This conversion of code happens line by line

4] Types of Programming Paradigms :-

- Procedural Programing Paradigm
- Functional Programing Paradigm
- Object Oriented Programming Paradigm

- Procedural :-

i] The entire program code is organized as a set of procedures that are executed in order.
ii] These procedures are also knows as functions & contain a series of steps that will be carried out when procedure is called
iii] e.g. C, FORTRAN, BASIL, COBOL, ALGOL, & PASCAL.

- Functional :-

i] This program code is organized in pure functions (which always yields the same value for same set of input w/out any deviation) with each function set upto perform clearly defined task.
ii] The data/variables are passed in the function as parameters (for the function to interact with other functions or programs.
iii] Language that supports this approach are : JavaScript, Python, etc

- Object Oriented :-
  i] The program data is organized in classes and objects rather than functions and logic
  ii] A class is blueprint for creating objects and an object is referred to as an instance of a class that has unique attribute & behaviour

5] About Java :-

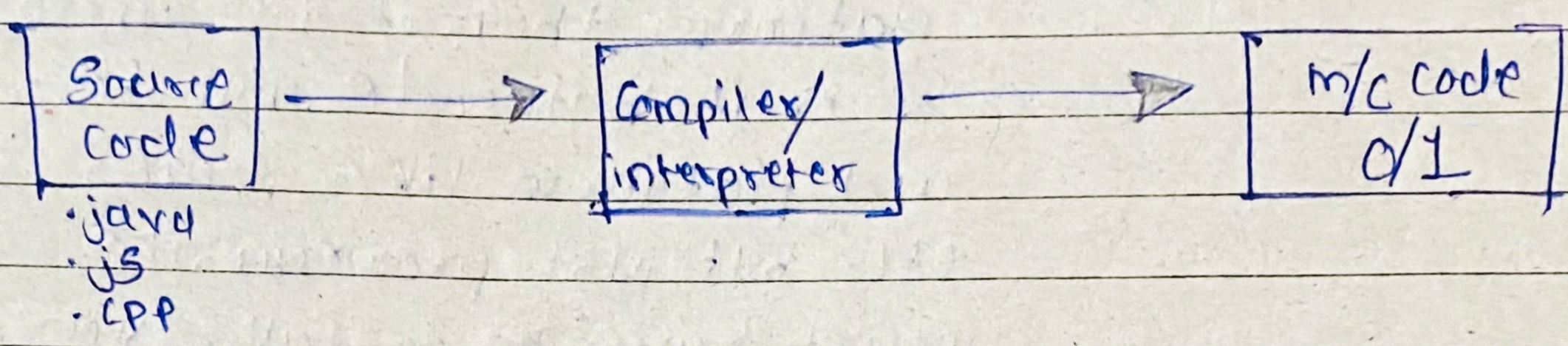
i] Java is class based object oriented programming language, which make it easy to write, compile & debug code.
ii] Java developed by James Gosling in 1995 & it is later aquired by oracle corporation.
iii] Java used to develop web application, mobile application, big data processing and embedded system.

6] Features of Java :-

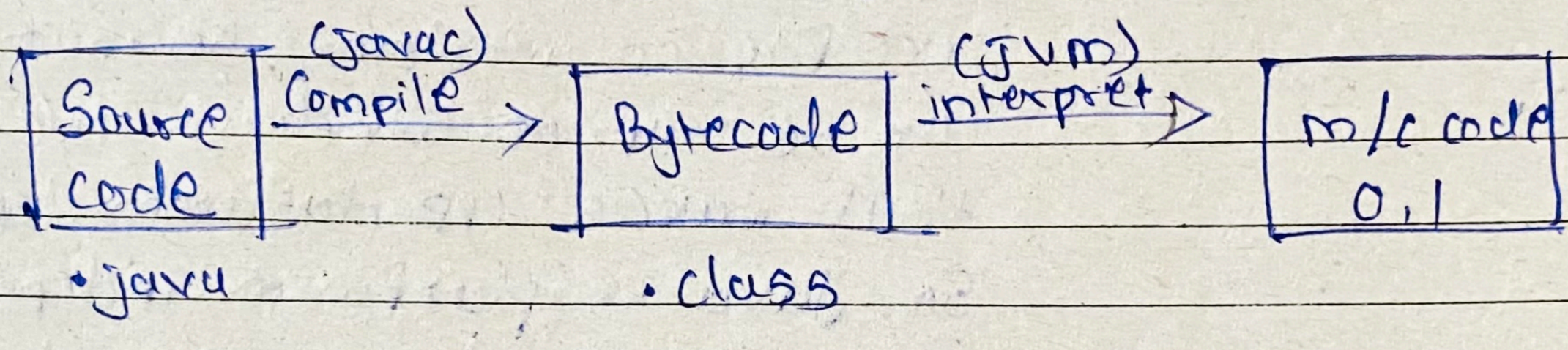
i] It is object oriented programming language.
ii] Java is platform independent programming language. (means it doesn't compile on dependn m/c code but it compile on platform independent byte code)
iii] Java is fairly simple & secure (like Java has automated garbage collection so you don't have to deallocate memory & there are no pointers & multiple inheritence.

1] Java Architechure :-

- General Compilation Process



- Compilation Process in Java



Here,

Source code = humane readable code with .java extextension.

javac = It Compile source code into bytecode. javac is name of compiler.

bytecode = after compiling source code into bytecode it generate .class file. ByteCode can be run on any platform which has jvm install in it. As it interprete bytecode in m/c code

this feature of bytecode makes java platform independent. If you are compiling java program in windows platform then it will generate bytecode & that bytecode will be run on any platform like macOS, linux, etc. but just prerequisite is we must have to install JVM on that platform.

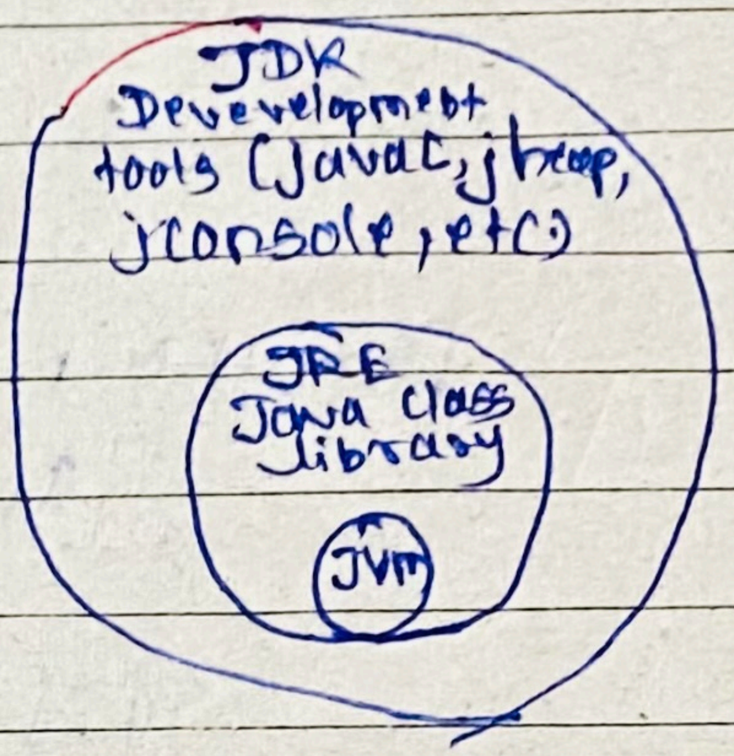
In CPP, we know .cpp file generate .exe file on windows. So that it can be run on windows m/c only. that makes cpp platform dependent. So, Java is platform independent & portable language. It is compile once & run everywhere.

JVM :- Java virtual machine. JVM is platform dependent. cause distribution of JVM on windows, macOS, linux etc is different.

JVM is interpreter that converts byte code (.class) into binary language.

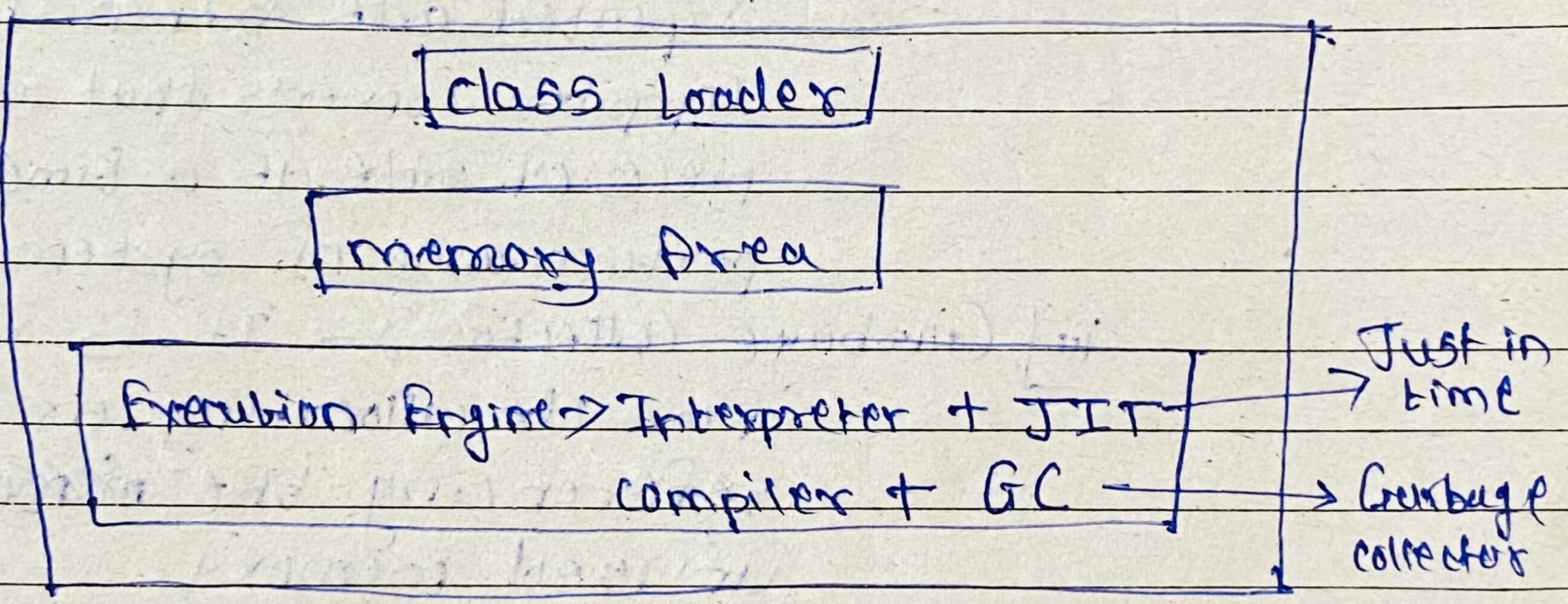
8] Java Architecture Components

JVM (Java Virtual Machine)
JRE (Java Runtime Envt)
JDK (Java Devrlopmet Kit)



- **JVM :-**

JVM provides runtime enviroment to execute the bytecode. It translates bytecode into m/c code



Here,

class loader → It is responsible for loading your byte code into the main memory.

memory Area → If any programme is executing by JVM so it will allocate that memory where op^n takes place.

Execution engine →

i] Interpreter → It translate code line by line.

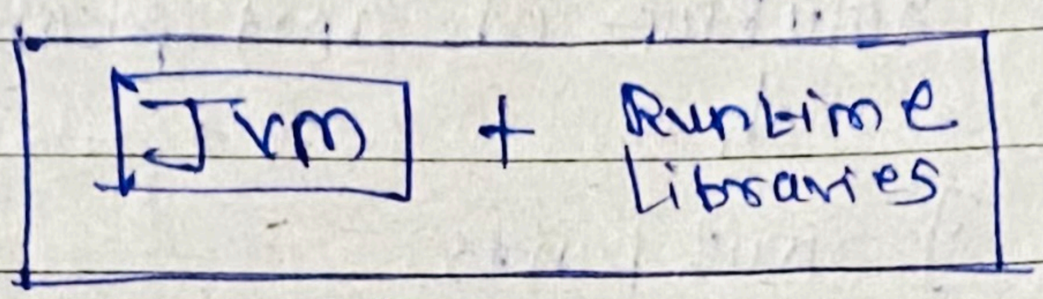
ii] JIT Compiler → It optimise the performance of JVM.

lets understand, when interpreter in execution engine checks code line by line & when it finds same repeated code & then that JIT compiler converts that repeated piece of code at a time so performance of system get optimised

iii] Garbage collector :- It is responsible for de-allocating or de-referencing the unused memory.

GC finds unused memory & delete it. So we don't have to do that manually like we do in CPP.

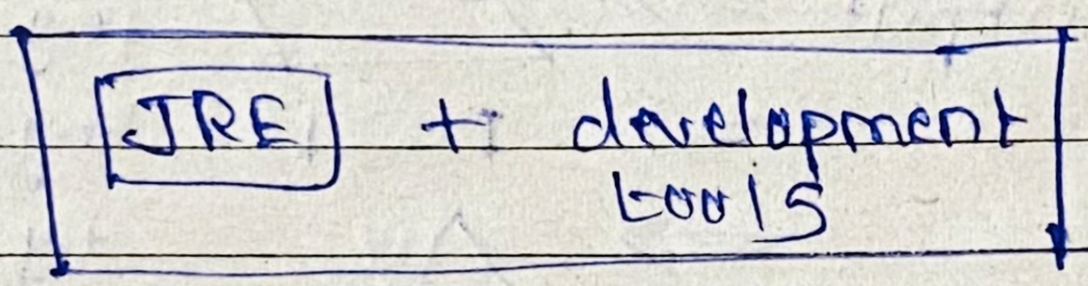
- JRE :→ Java Runtime Envt.
  JRE is nothing but Java libraries like collection, math. JRE provides that library



JRE

So, JRE = JVM + Runtime libraries.

- JDK → Java Development ~~Tools~~ kit.
  JDK provides ~~tools~~ tools like javac, debugging tools, monitoring tools, javadoc.



JDK

So, JDK = JRE + Development tools.

Summary.

JVM	JRE	JDK
↓	↓	↓
class loader + memory area + execution engine	JVM + runtime libraries	JRE + development tools