(5) PORTABLE

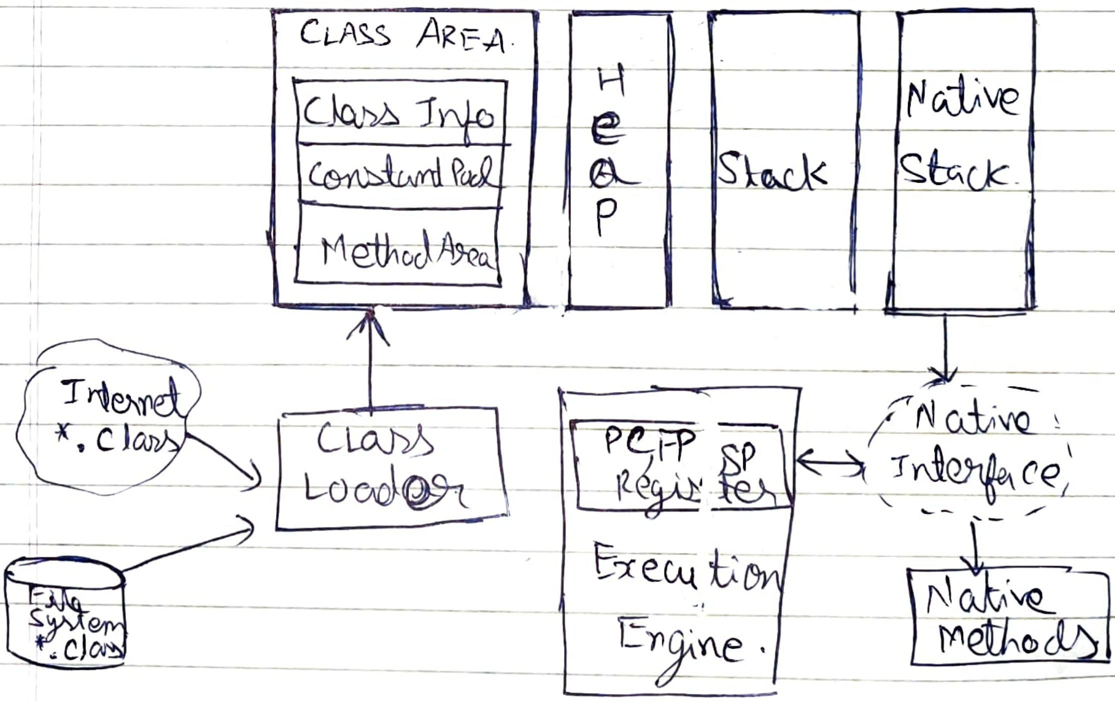
(7) HIGH PERFORMANCE

(8) MULTI-THREADED

(9) DYNAMIC

(10) INTERPRETED
Means go line by line.

★ ORGNISN OF JVM



INSTALL JAVA

(1) Install → $\boxed{\text{JDK} \quad + \quad \text{JRE}}$
Combo Setup.

Java has compressed its ~~[illegible]~~ ~~in~~ b.c. in a some files.

Compressed using 'jar'

CMD

```
D:\rt> jar xf rt.jar
```

xf → extract
rt.jar → file name
rt.jar → rt → run time ✓
→ contains all class files ✓

- Java is open-source prog. lang.

Today 5 types of Apps are made

Java { S → social, M → Mobile, A → Analy-sed, C → cloud, I → Internet

Finding source code

jdk → src.zip → copy it to other drive → Java → lang → String.class java

src.zip ↳ contains source code

✓ javadoc → creates web based doc. of class files (creates file named 'indx.html')

```
D/src/src/java/lang > javadoc String.java
```

↳ For String Only.

```
D:\src\src\java\lang >javadoc *.java
```

→ For all files with '.java' extn.

PATH SETTING

(1) Temporary

```
D:\ - - -        >path
```

→ checking

shows all path of windows

just ~~Show~~ write name of tool → it will show all switches of tool.

```
D:\ - - -        >set path=e:\f2;
```

(2) PERMANENT

D:\ - - - : Copy location of bin : Computer prop → Adv. comp ⚙ → Env. Variables → Change path in variable

12/06/17

☆ Java (or JDK) don't have any inbuilt editors.

~~IDE~~

For Java Program Writing
we can use any 'Text' Editor
↳ tool that doesn't encrypt into any perticular file format like MS Word saves file in ".docx"

PROGRAMS

~~PR~~

→ Mandetory Rule

1 MR: In Java every statement is to be written within a 'class' block'
↳ explained on next page

~~Notepad~~

~~Class~~

→ Conventional Rule

CR:
2. Always keep the 1st letter of your class name as a capital.
Why: It increases the readiblity of code.
(i)

Notepad.

```
class Demo
{
public static void main(String s[])
{
System.out.println
    ("hello java");
}
}
```

block

variable
↳ String name

New way

(String... s)

Variable length argument

Note→ Java has inherited a complete syntax from the 'C-language'.
→ A good example of INHERITENCE.

MR: (i) Source code of a Java program must be stored in a file having a extension '.java'.

CR: Always keep the name of '.java' file same as your 'class name'.

- Save as ~~Demo~~ "Demo.Java" OR use "All files option.
  use squates

to compile

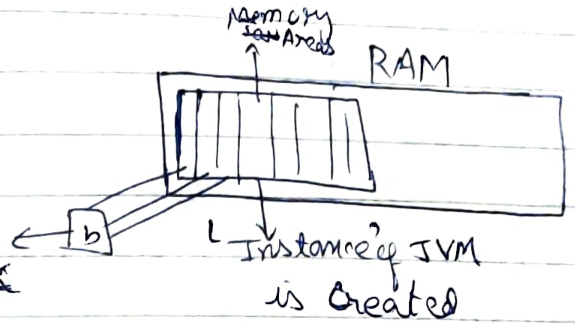
```
>javac Demo.java
```

Now, byte code has been generated.
Now, to run

D:\> ~~Demo~~ java Demo

class name

C:\> java



~~b on hardisk~~

- This Instance remains in RAM as long as Java code runs & gets removed when last line code is completed

13/06/17

IQ: Can you run more than 1 JVM simultaneously on ~~One System~~ (Same+single) ~~Single~~ OS?

Ans → Yes, As you can open one Or more java program using diffrent command Prompt. So All the instance are of Same JVM.

- Also we Can temporarily set path for JVM v7. And use a new JVM. And two JVM are running at same time

---

☆ VIMP
At a time only One class ~~can~~ can be executed into the 1 instance of JVM which is having the main method.

- Just as we open multiple 'Windows' of same program.
  Similarly are ~~In~~ multiple 'Instances' of same JVM.

Note:

(i) A runtime instance of the JVM has a clear mission in life: to run one Java application.

(ii) When a Java app. starts, a runtime instance is born. When the app. completes, the instance dies.
If you start 3 Java apps at the same time on same computer, you will get 3 JVM instances.

(iii) Each Java app runs ~~runs~~ inside its own JVM.

(iv) A JVM instance starts running its solitary app. by invoking the main() method.

Evergreen Page No. Date: / /201

V. Imp.

Golden Rule

✓ The byte Code of any class is always stored into a separate '.class' file & the name of that '.class' will be the same as that of your class name.

IQ: Can wee keep more than 1 class in a single java file?

Ans → Yes

✓ But their byte cod will be generated in different '.class' files

~~Development~~

CR: Always Keep a seperate Java file for each class.

IQ → If you are having more than 1 class in a single Java file, can we Keep a main f^n in each class.

Ans → Yes.

PRACTICAL

(i) Keep both class in same ~~Java~~ a '.java'

```
✓class Demo                       ✓class Demo1
 {                                  {
  - - - - - -                        - - - - - -
 }                                  }
  System.out.println("Hello Java    System.out.println
                  by Demo")           ("Hello Java
 }                                     by Demo1")
 }                                  }
```

* A prog ~~in~~ having 2 class in which 1 class don't have main.
* A prg. having 1 class that too without main

IQ Can we keep more than 1 'main' f^n in single class?

Ans → Yes,

- java follows 'F^n Overloading'.

↓

we can ~~to~~ have more than 1 f^n of same name in same class.

Practical

```
Class Demo
{
 public static main void (String... S)
 {
  System.Out.println("Hello java");
 }
}
```

* A prg. in which 1 class have 2 mains
  - → (i) Both main are same
  - → (ii) Both " " diff.

★ If you are to make any class of Java executable than that class must be having a f^n ~~having~~ named 'main()'.

(ii) Not every class is requiring a main f^n.

— x — x — x — x —

IQ → Can we call the main method explicitly? → f^n call by user

Ans → Yes

Eg →
```
class Demo
{
public static void main(String... S)
                        └ variable length Argument
{
System.out.println("Hello from Demo");
}
})
Class Demo1.
{
pubic static void main (String... S)
System.out.println("Hello ~~from demo1~~"); from Demo1
String Z[]={"Hello"}
Demo.main() → main called ~~from~~ by us / or USER
              this is known as explicit calling
```
we can have as many or zero arguments as we like since we used 'variable length argument'

IQ → What if "public static ~~main~~ void main" is written as "static public void main".

Ans → Both are same i.e. have no difference. You can do a practical in which in same class make two main on by