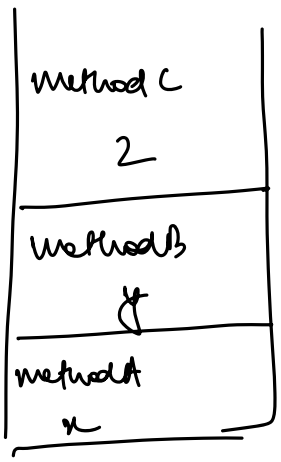
Java Interview Questions

1) What is the difference b/w stack and heap memory in Java?

Ans:
- Heap is the memory which is used to store objects ⇒ actual objects reside in the heap.
- Stack stores primitive & reference variables inside method calls.



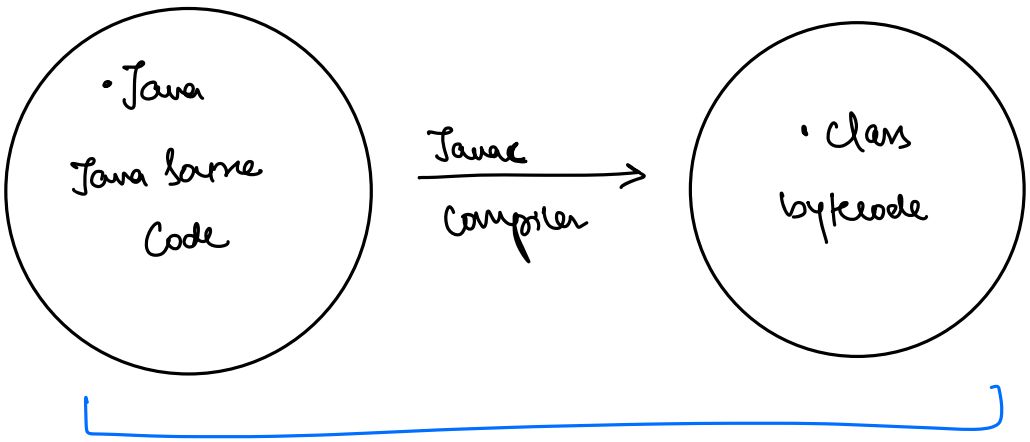

```
methodA() {
    x
    methodB() {
        y
        methodC() {
            z
        }
```

* Possible to run any Java program without using heap memory ⇒ Yes

* Possible to run any Java program without using stack memory ⇒ No

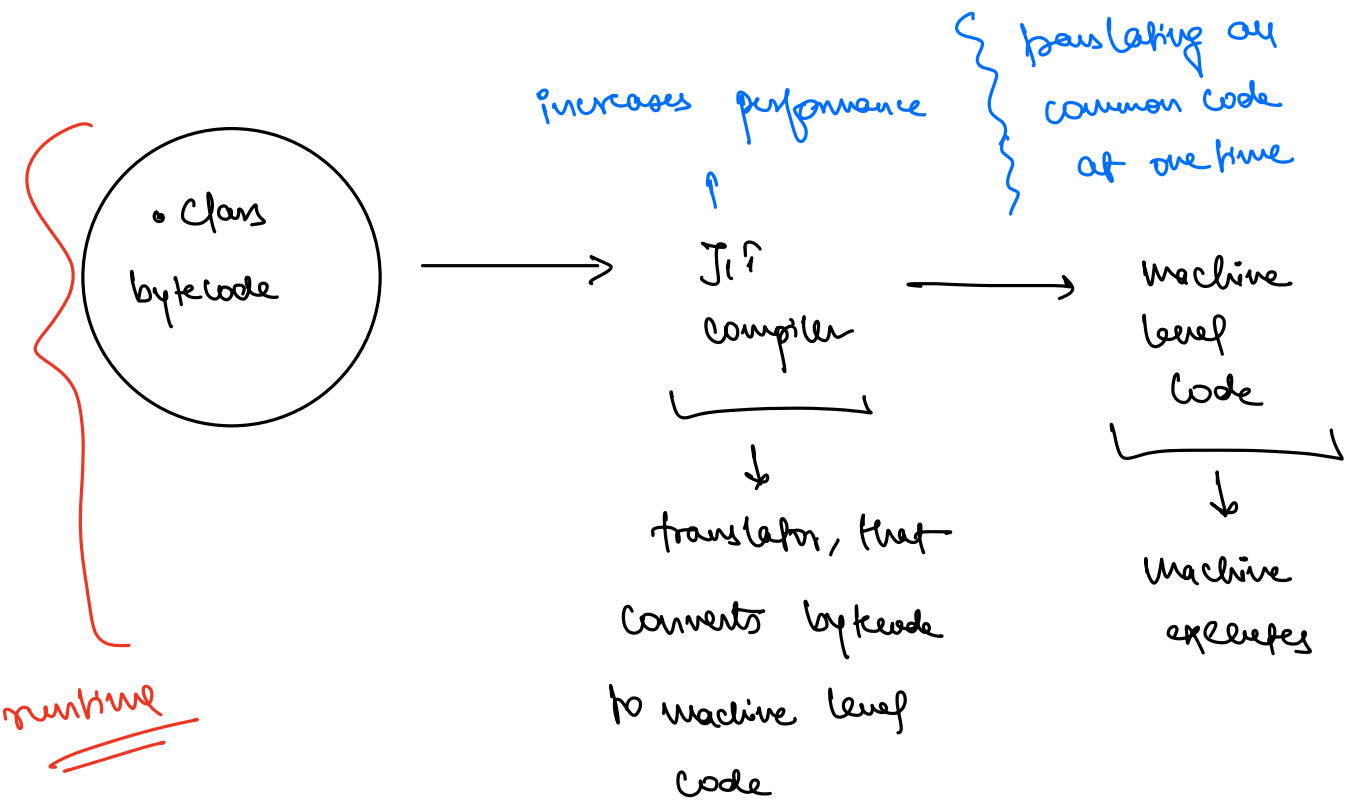
Q2 What is JIT ?

Ans JIT => Just in time compiler.

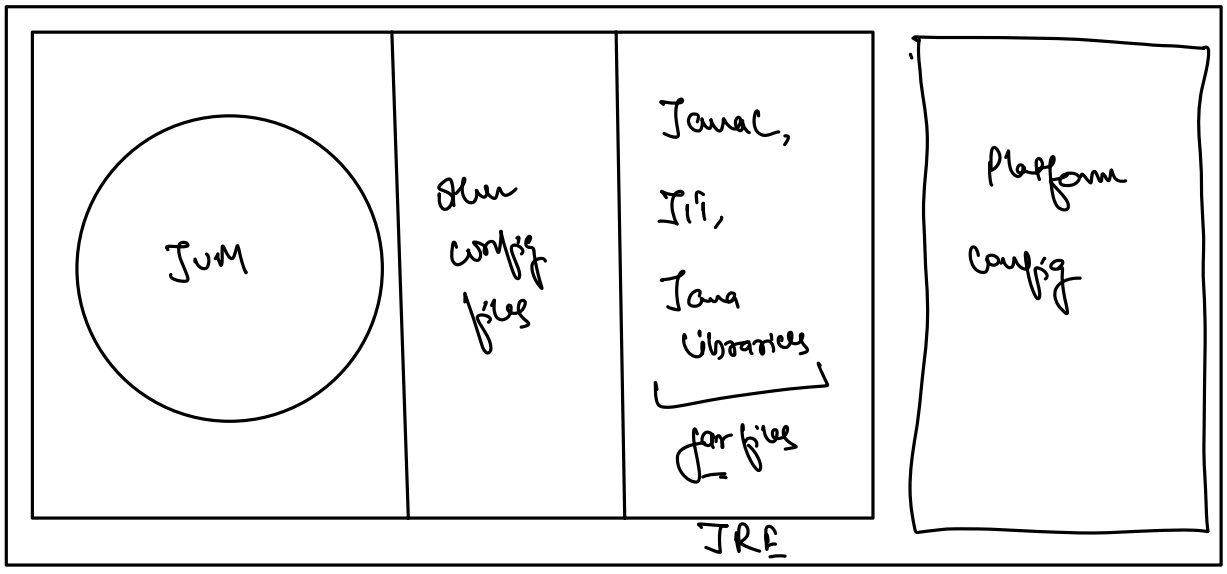


Compile time flow

TODO: Run a Java program from terminal / command prompt.



⇒ JRE | JDK | JVM :



JDK => Java Development kit

JRE => Java Runtime Environment

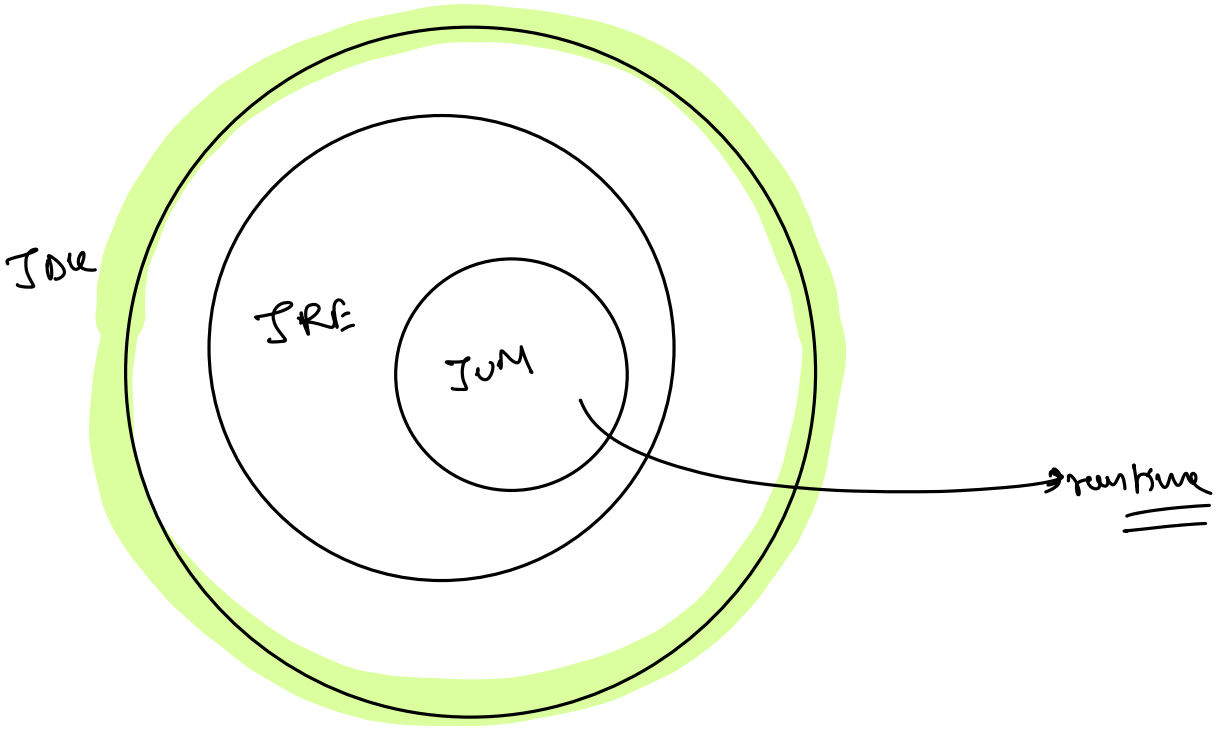
JVM => Java Virtual Machine

JIT => Just in time compiler

JDK → JRE → JVM

JRE: development + running

JVM: development



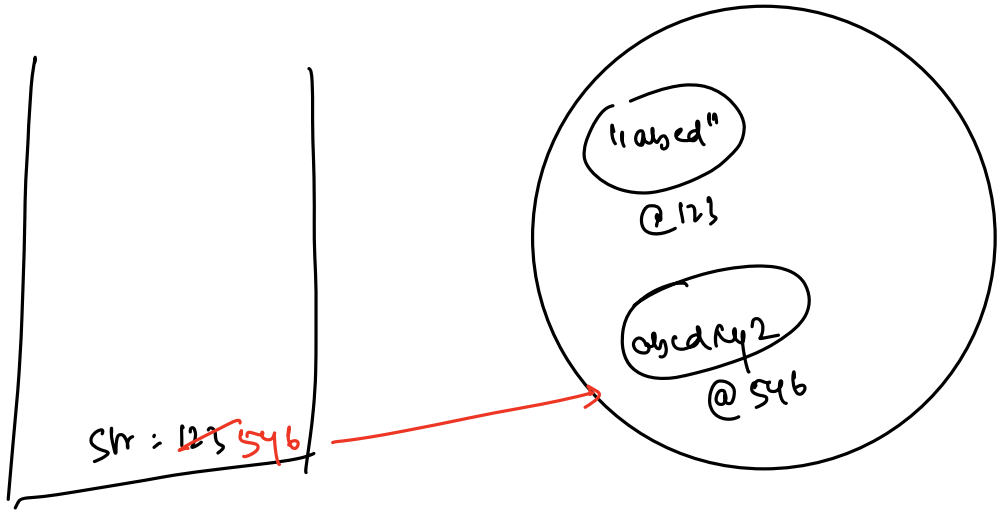
⇒ Strings are immutable in Java: (Why)

↓

Something that can't be changed

```
String str = "abcd";
str = "xyz";
str = str + "pqr";
```

String str = "abcd";



str = str + "xy2";

StringPool :- memory location inside heap. all string objects are stored here

String is used extensively in programs, are expensive in terms of space and might have multiple duplicates.

→ Optimise memory ⇒ Java doesn't create multiple copies of the same String

Class1

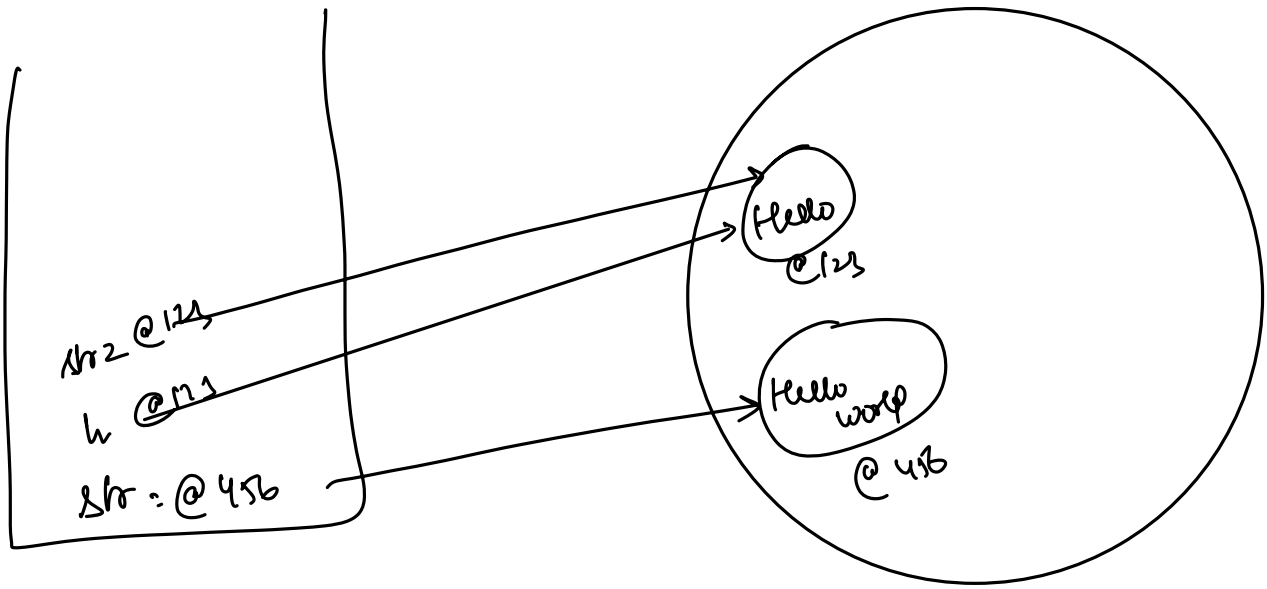
str : Hello

Class2

h : Hello

Class3

str2 : Hello



str = str + "world"

str : Hello

arr[] = [a, b, c, d, e, f] ——.

for(char c : arr) {

str = str + c

}

↓

StringBuilder → mutable String

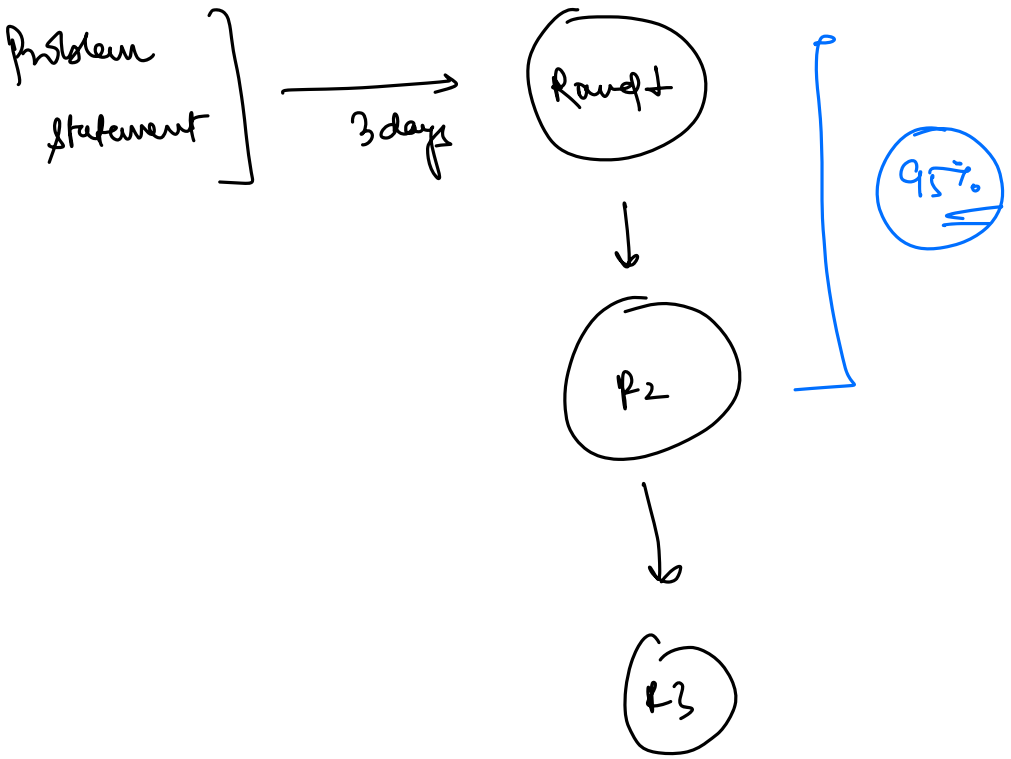
↓

String (creates)

StringBuffer | StringBuilder ⇒ Mutable Strings

↓

[thread safe]



String str = " " → Hello

String str = new String(" ") ← check

↓

Hello

⇒ final, finally & finalize()

final ⇒ keyword

finally ⇒ code block [try | catch]

finalize() ⇒ method

JVM before destroying an object calls the finalize() method.

finalize() ⇒ clean up resources, get a confirmation for task completion.

↓

pre destruction tasks

```
Object o = new Child()

o.finalize()
```