6. Java Variables | Reference / Non-Primitive Data

Constant In Java → (static final) Keyword to make constant.

Each Primitive datatype has its subsequent Wrapper Class.

int → Integer

Short → Short

long → Long

byte → Byte

double → Double

boolean → Boolean

String → String

char → Character.

Now the question is.

Q) Why do we need these Wrapper Classes?

A) Lets see with an example,

```
psvm(){
    int a = 10;
    modify(a);
    sout(a);
}

void modify(int a){
    a++;
}
```

What do you think will be the value of a?

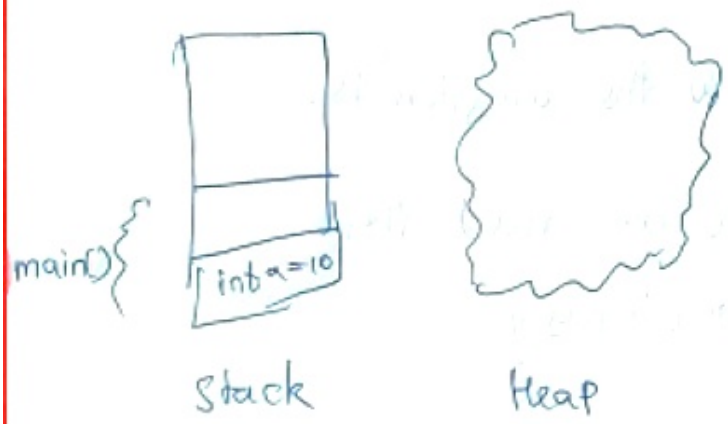
Value of a will be 10 only. Value of a will Not change.

Why?

Because Java is a pass by value language.

P.T.O.

We know that primitive datatypes are not allocated space is the Heap. Primitive datatypes are allocated space on the stack itself. Within the scope of the method.



Java does not have pointers concept.

Solution?

Along came the wrapper Classes-

Integer inta = new Integer(10); → memory Allocated in Heap

```
psvm(){
    Integer a = new Integer(10);
    modify(a);
    sout(a); → Now value will be 11.
}

void modify(Integer a){
    a++;
}
```

Auto Boxing / Wrapping

```
int a = 10;
Integer wrapperA = a;
```
[Automatic Boxing Happened]

UnBoxing / UnWrapping

```
Integer a=10;
int b = a;
```
[b does not have reference to anything]

String Constant Pool

Inside heap, there is a specific memory allocated just to store String Literals.

If any Literal is already present and a reference is created using same Literal, both reference point to same Literal