equals()

LC → contests
(1) ↳ Bi-weekly → 8:00 pm
(2) Personal weak topics → focus on them

DP / graphs → 1 week → 10~12
→ 1 week

Agenda:

1. ~~Strings~~
2. ~~Interning in Strings~~
3. ~~== v/s equals()~~
4. ~~Immutability in strings~~
5. String Builder v/s String Buffer.

Interning C

Primitive data types:

int, float, double, boolean, long, char, short

int a = 5;

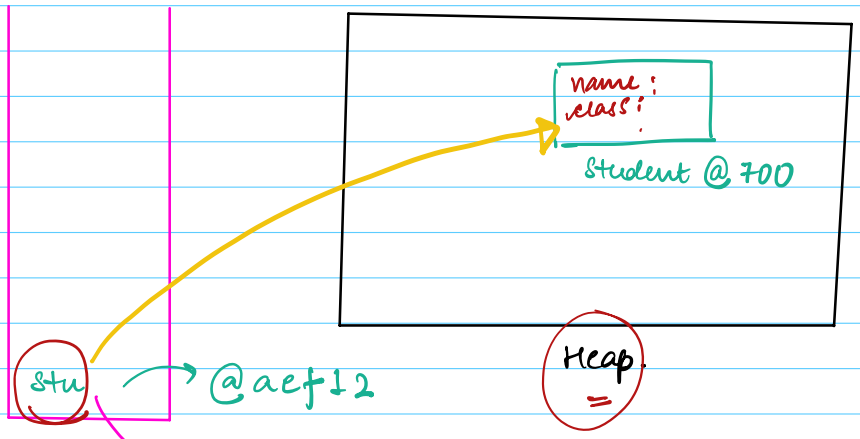
→ Variable stores the value itself in the stack.

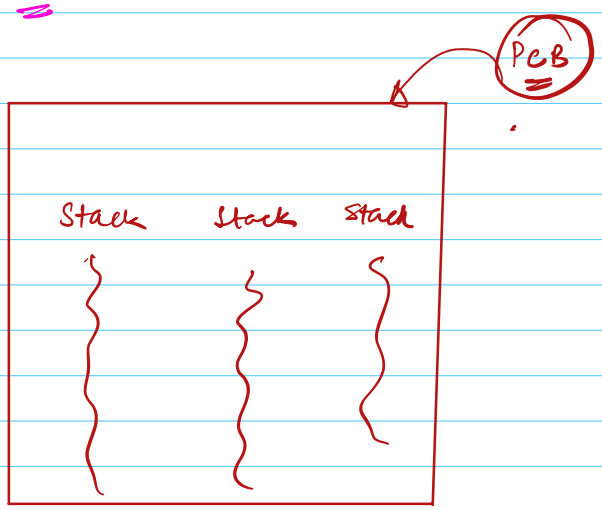
Non primitive types

Student stu = new Student();

stu → Reference

new → new object gets created

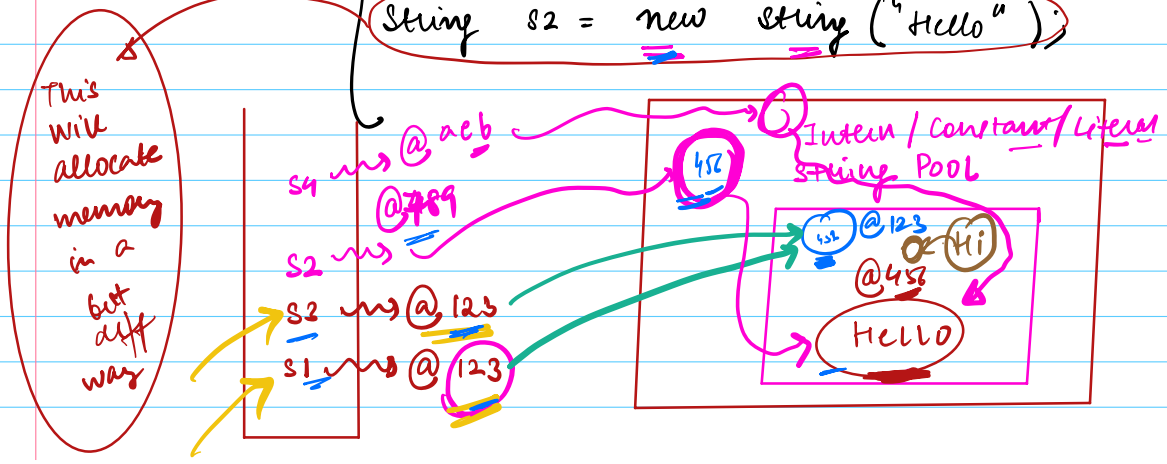




String s1 = "Hello";

String s2 = new String ("Hello");

This will allocate memory in a bit diff way

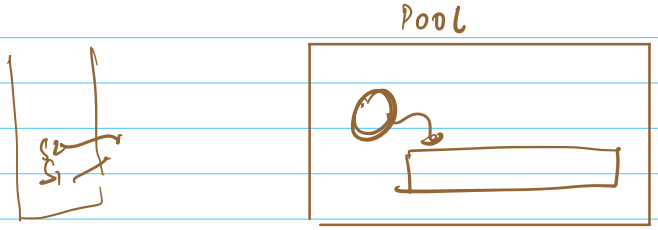


String s3 = "Hello"

String s4 = new String("Hello")

String s5 = "Hi"

What is Interning?



Why?

→ Optimization (saves memory)

Security

Immutability

== v/s equals()

== → compares reference

equals() → over

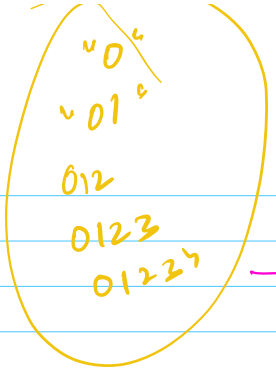
Overridden → how to

compares actual String values

s1 == s2

s1.equals(s2)

Mutating strings




```
s = " "
for ( int i=0 ; i<5 ; i++ )
    s += i
sout(s)
```

01234

is the string getting changed?

sc → ?

1 + 2 + 3 - - - n

⇒ $\frac{n(n+1)}{2}$

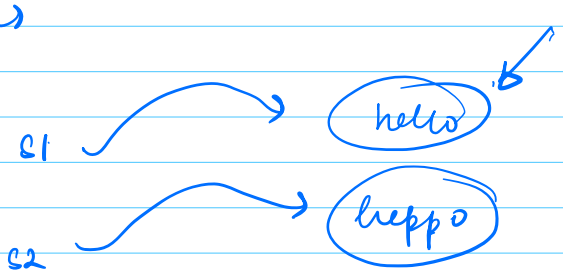
$O(N^2)$

(*) String inside the pool will never change once created. When we try to update string object, it instead creates a new string in the pool

s = "Hello"

s.replace('l', 'p')

"Heppo"



int a = 5

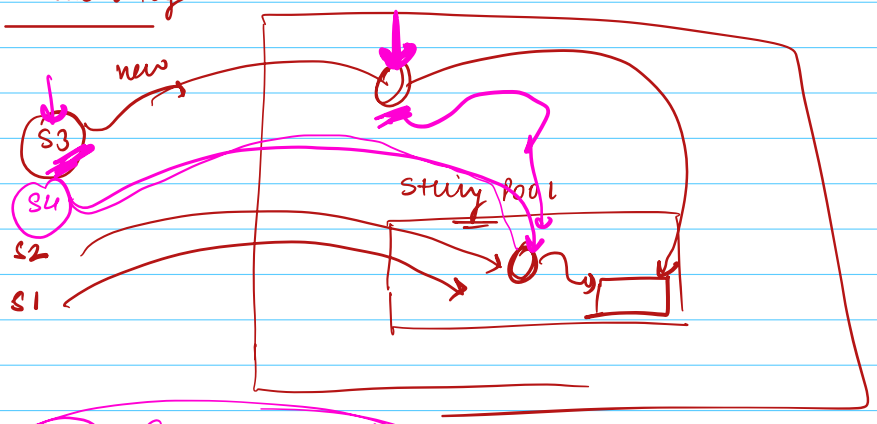
Integer a = 5

a = new Integer(5)

10:05

10:10

String Interning



S4 = S3.intern()

S4 = "hello"

S4 == S1 → True

StringBuilder and StringBuffer

Are Strings thread-safe?

↓

Yes → Immutable

Immutability

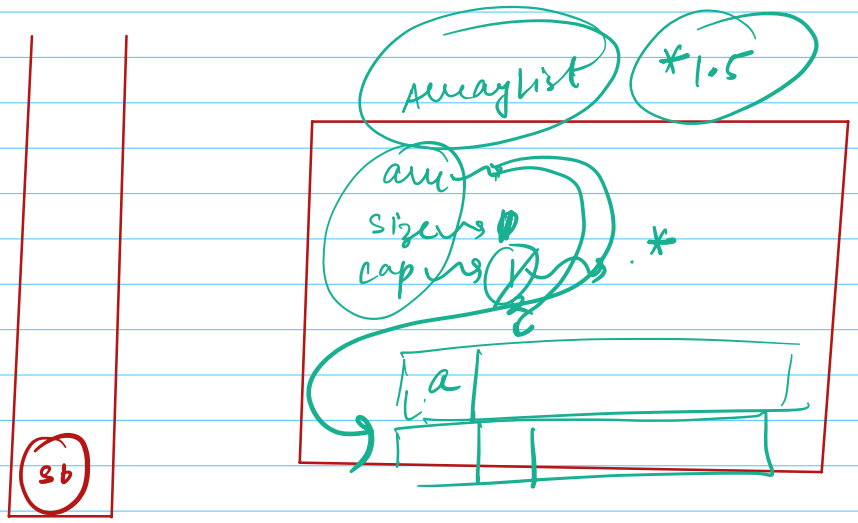
StringBuilder sb = new StringBuilder()

↓

similar to ~~ArrayList~~ vectors

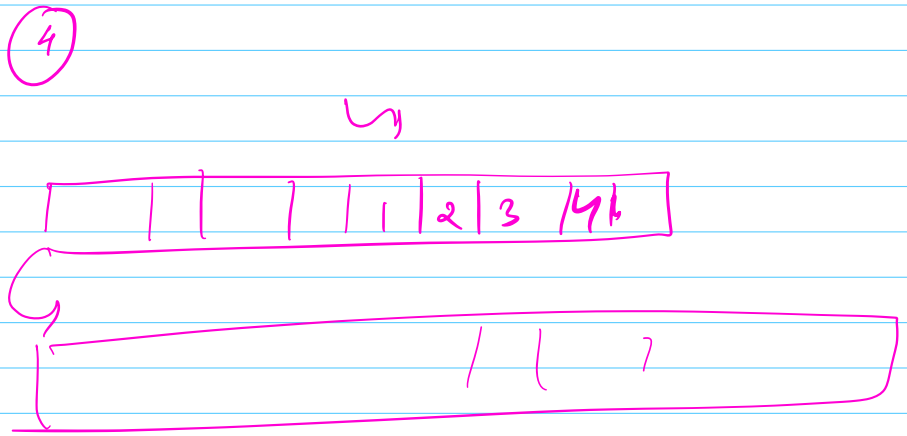
sb.append('1')

⇒ NOT THREAD-SAFE



insert operations TC ?

amortized O(1)


4
1
2
3
4