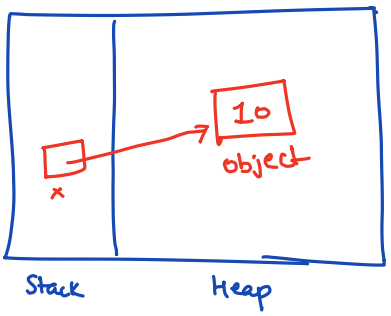
→ All objects are created on Heap memory.

→ In Python everything is an object.

RAM



X = 10

Reference to the object

Heap as it a object

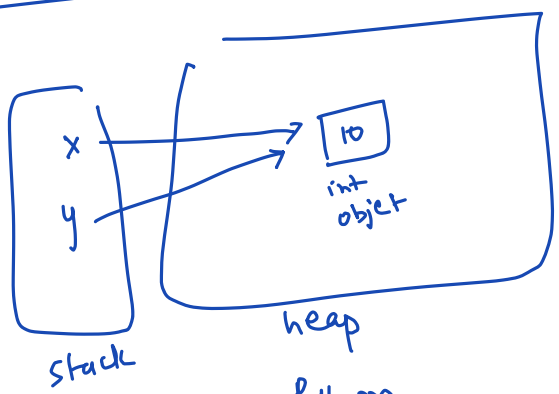
C++ | Java

int x = 10; // Primitive variable

↓

Stack

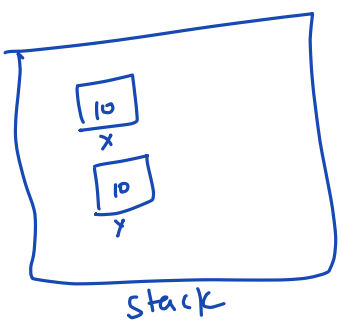
Python vs other Lang



x = 10
y = 10

C++ | Java

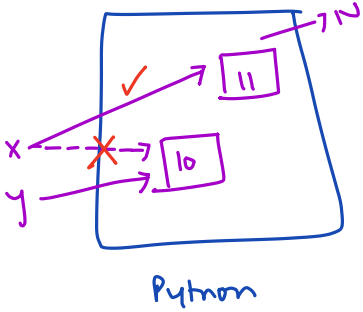
int x = 10;
int y = 10;

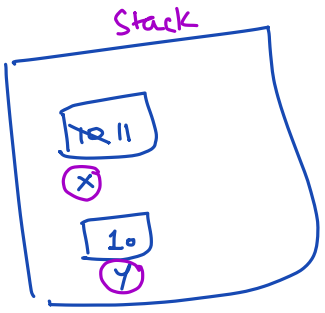


Update the value of x

x = 10
y = 10

x = x+1 => 11





optimisation [Python → object of same value exists, new object "may not" get created.]

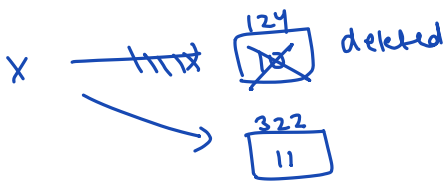
Garbage collection ↳ freeing up the objects which don't have reference automatically.

↳ Java & Python (Automatic)

↳ in C++, mannually delete unused objects.

X = 10

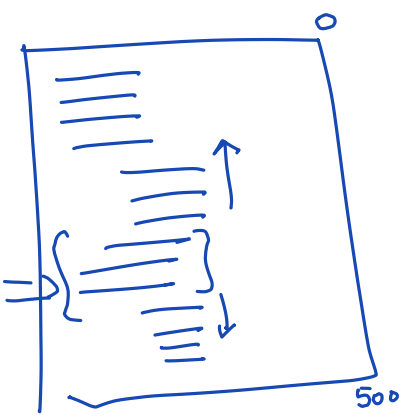
X = X + 1



Why Functions? → C/C++/Java/JS/---

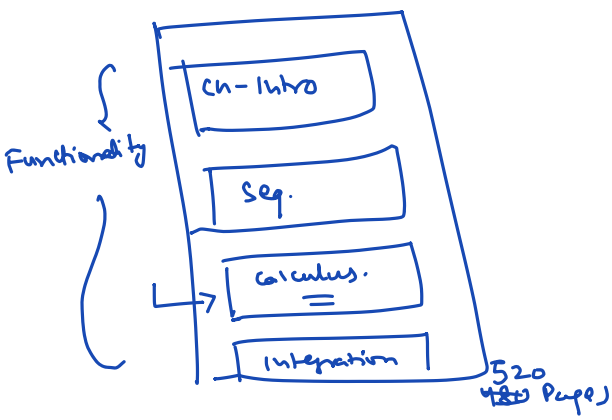
Code bases

Millions of lines of code



Book B1

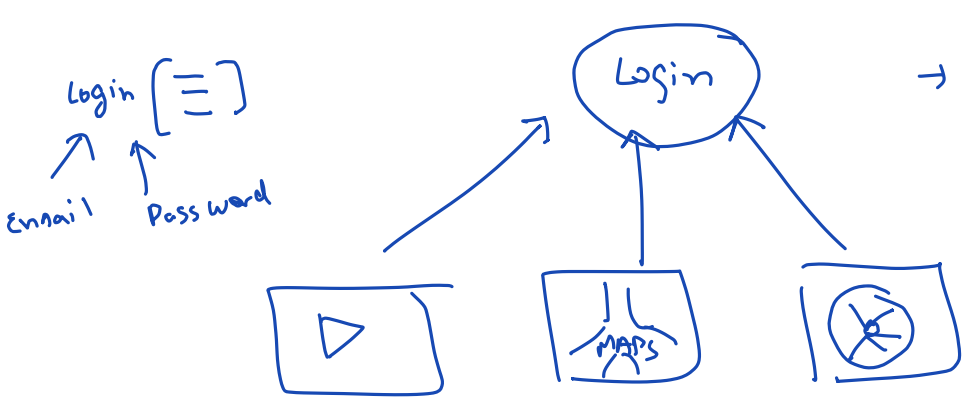
1 chapter



Book B2

→ organised, Modular

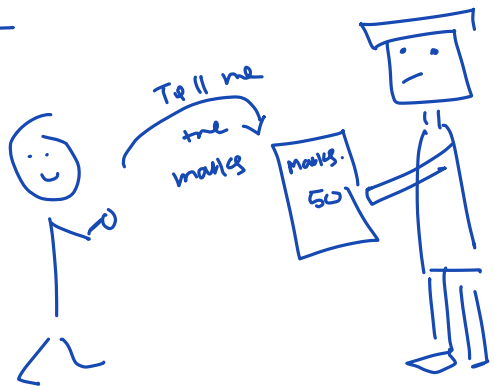
→ easy to read & maintain.



→ Reusability.

debug.

School



① you score 50 marks.

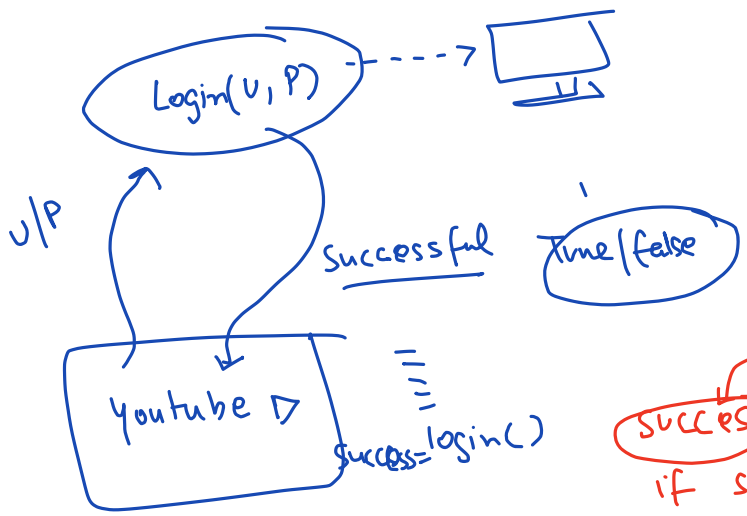
Print(....)

② Marksheet "return"

③ you got 50 marks

+ Take your marksheet

Why Return is useful?



True

success = login()

if success:

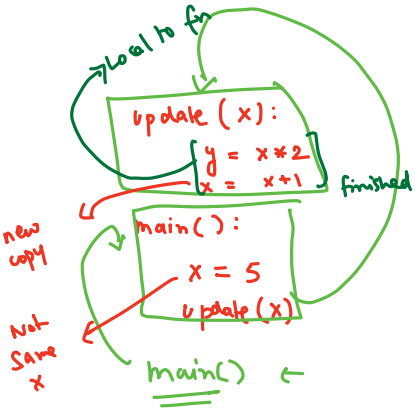
upload video

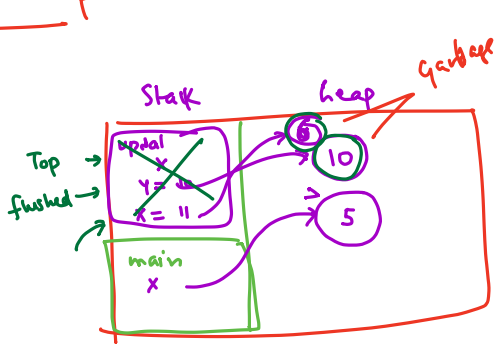
edit

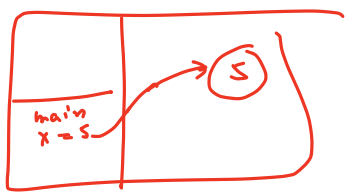
else:

Try login Again

Functions → Stack (LIFO) & Heap

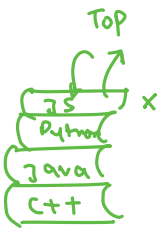


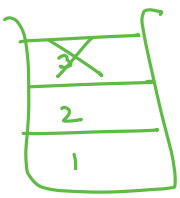




Update
6 → 1
10 → 1
5 → ~~9~~ 1

Update over
6 → 0
10 → 0
5 → 1





Last In
First Out