Cmpe 150 Lab 3: For Loops

Last Week

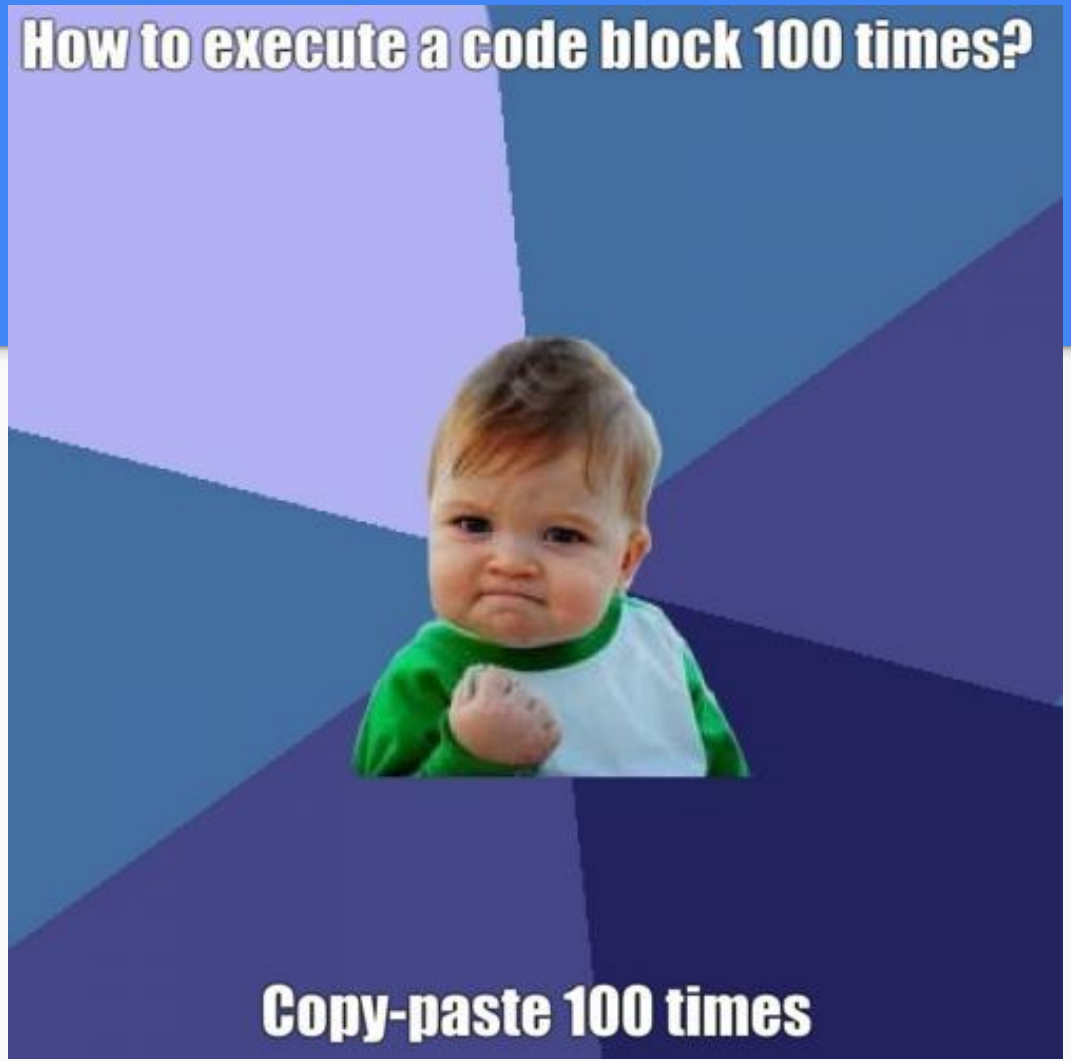
- We divided our code into smaller chunks called functions.

- Today we have a new goal: Being able to execute some portions of our code multiple times without explicitly writing them again and again.

Example Problem

- I want to print "Cmpe 150 is a very fun class" 1000 times.

- Should I write print("Cmpe 150 is a very fun class") 1000 times? Or call a function 1000 times? There should be a better way.


How to execute a code block 100 times?
Copy-paste 100 times

For Loops

- Here is the syntax of for loops
- for variable_name in range(100):

    code line 1

    code line 2

    code line 3

Example

- for i in range(100):

  print("This is want I want to say")

  print("Cmpe 150 is a very fun class")

  print()

Question

- Does it have to make the same thing in each iteration?

- No, we can use the variable (e.g. i) in our code so that what it does depends on i. We can directly use it or give it as an argument corresponding to a parameter in our function calls.

Example

- for i in range(100):

  print("This is the number I print", i)

  print("Also its square is:", i * i)

  print()

Example

- for i in range(100):

        print_something(i)

  def print_something(i):

        print("This is the number I print", i)

More About Range

- It is not limited to range(10). Here is the general idea
- range(start, end, step_size)
  - end is **EXCLUSIVE**

- By default, start is 0 and step_size is 1, so
  - range(10) -> 0, 1, 2, 3, … 9
  - range(5, 9) -> 5, 6, 7, 8
  - range(4, 10, 2) -> 4, 6, 8
  - range(97, -53, -30) -> 97, 67, 37, 7, -23

Stair steps starting at 0 ;) Builder must be a programmer.

Therefore

- We do not have to say `for i in range(10):`
- It can be things like `for iterator in range(1, 103, 10):`

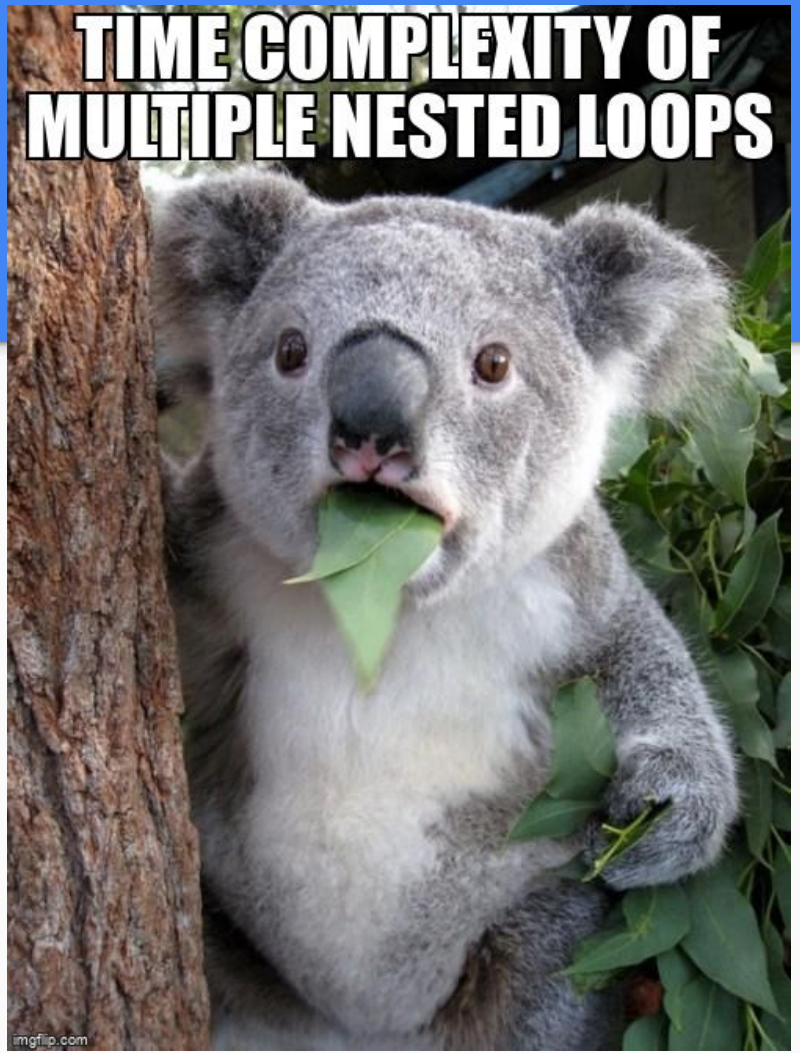
Nested Loops

- Can we use a loop inside another loop? Sure

```
for i in range(10):
    for j in range(10):
        print(i * j, end=' ')
    print()
```

Nested Loops (Cont.)

- Can we use a loop inside another loop which is already inside a loop? Sure



Context

- When the loop is over, it does not forget the variable we used in the loop. Not like forgetting the variables used in a function after its execution is completed.

```
for i in range(10):
        print(i)
print(i)
```

Thanks

Any questions?

References

1. https://www.iloveimg.com/meme-generator
2. https://twitter.com/overflow_meme/status/1358873684639969283