Imperial College London

BIOE40002 – Computer Fundamentals and Programming 1

Part II – Programming 1, Lab 2

***Binghuan Webster Li** | Department of Bioengineering*

binghuan.li19@imperial.ac.uk

March 10, 2022

Today's Schedule

Imperial College London

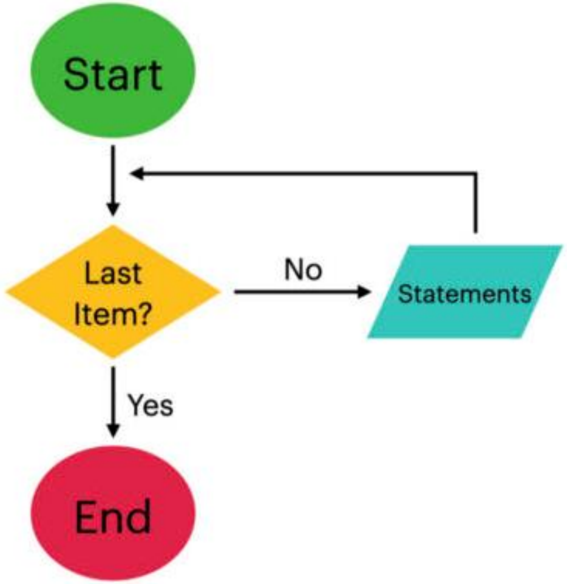
- **Recap (~ 10 mins)**
  - *Flow control:*
    - `for` loop
    - `if…else…` statements
    - `while` conditions
  - *Functions*

- **Lab work**

Flow control

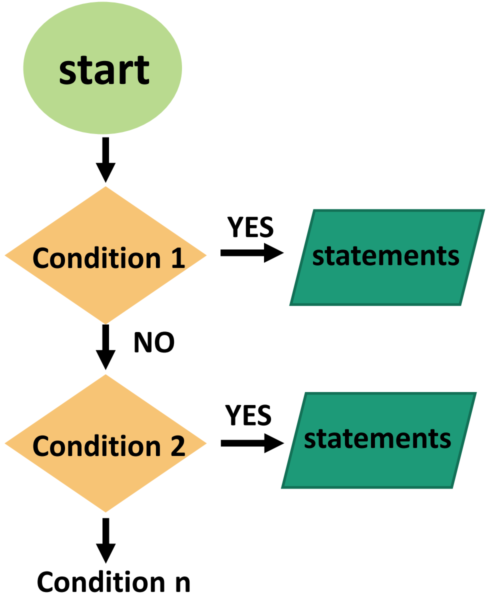
Imperial College London

For loop



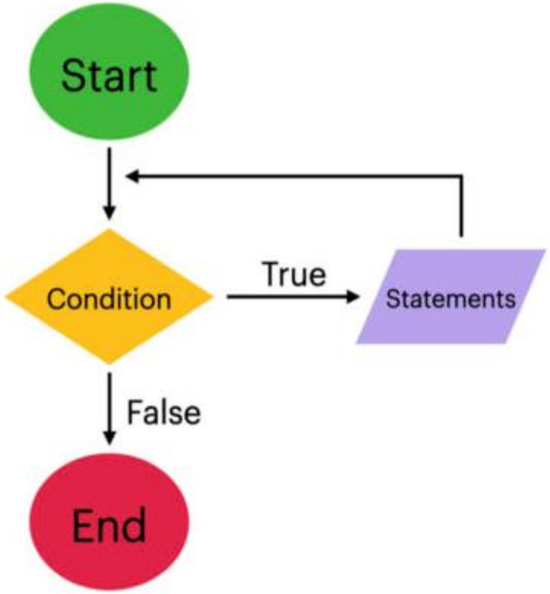
```
for condition:
     statements
```

If...else... statements



```
if condition1:
     statement21
else:
     statement2
```

while condition



```
while condition:
     statements
```

Functions

- Using functions to divide and conquer a large task

(A)

```
statement1
statement2
statement3
statement4
statement5
.
.
.
.
.
.
.
statementN
```

Long, complex sequences of statements

(B)

```
def function1(arg1):
    statement1

def function2(arg2):
    statement2

def function3(arg3):
    statement3

def main():
    function1(arg1)
    function2(arg2)
    function3(arg3)
```

Divide the task into smaller tasks, each of which is performed by a separate *function*

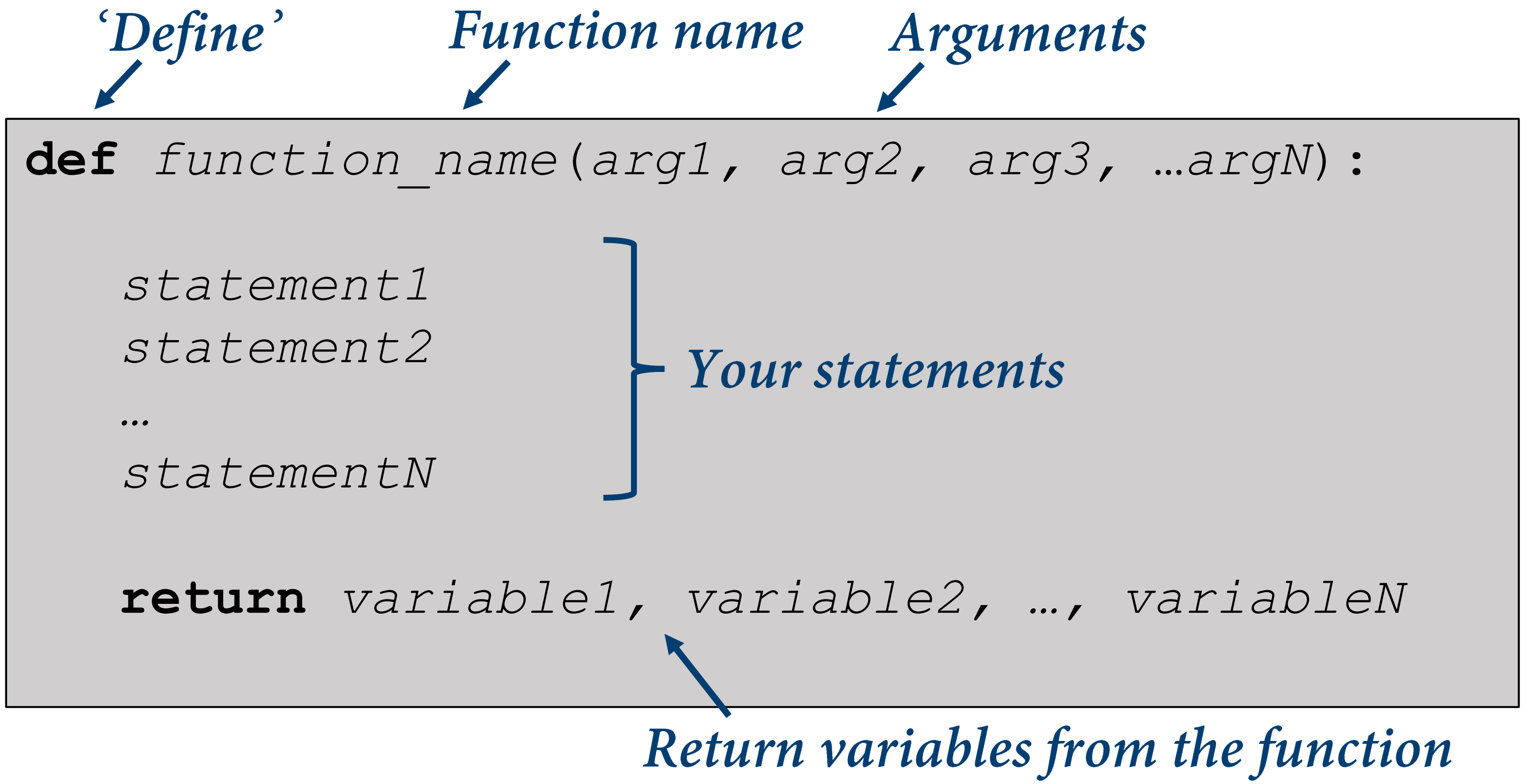
✓Simpler code

✓Code re-use

✓Better testing

✓Teamwork

Functions



```
def function_name(arg1, arg2, arg3, …argN):

    statement1
    statement2
    …
    statementN

    return variable1, variable2, …, variableN
```

- ➢ ***Argument:*** data send into the function. It is optional.

- ➢ ***Return:*** data send out the from function. It is optional.

Q: Why do we need arguments to pass data into the function, and use ‘return’ to get the data from the function?

Scope

Q: Why do we need arguments to pass data into the function, and use ‘return’ to get the data from the function ?

```
def addOne(x):
    result = x+1

def subOne(x):
    result = x-1

def square(x):
    result = x**2

addOne(1)
print(result) # ???
```

Which ‘result’ is the one we want??

- Variables inside the functions and variables outside the functions are ‘isolated’ by ***scope.***
  - The functions do NOT know the variables outside the function unless you *pass* them into the it.
  - Your program does NOT know the variable inside the functions unless you *return* them from the function.

After a slight modification…

```
def addOne(x):
    result = x+1
    return result

def sub1(x):
    result = x-1
    return result

def square(x):
    result = x**2
    return result

inc1 = addOne(1)
print(inc1) # 2
```

The variable ‘inc1’ now holds the same value as the variable ‘result’ in the ‘addOne’ function.

Questions?

That’s it for now.

You can now proceed to the Lab 2 exercises.