Introduction to Computer Programming
Lecture 11.3:

Review :
Functions,
Reading and Writing Files

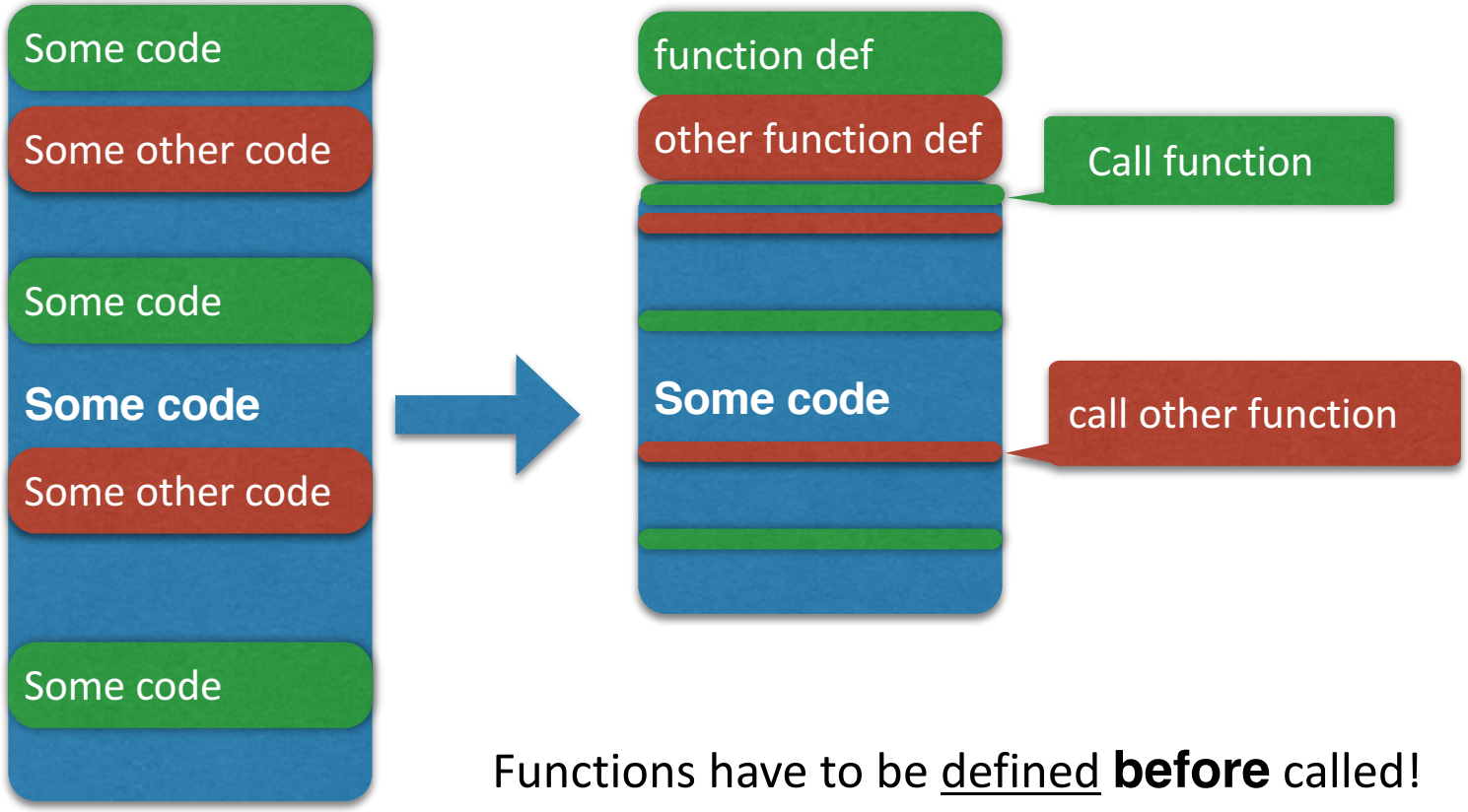
Hemma Philamore
Department of Engineering Mathematics

Functions

Cleaner code : easier to understand and maintain.

Shorter

More reusable

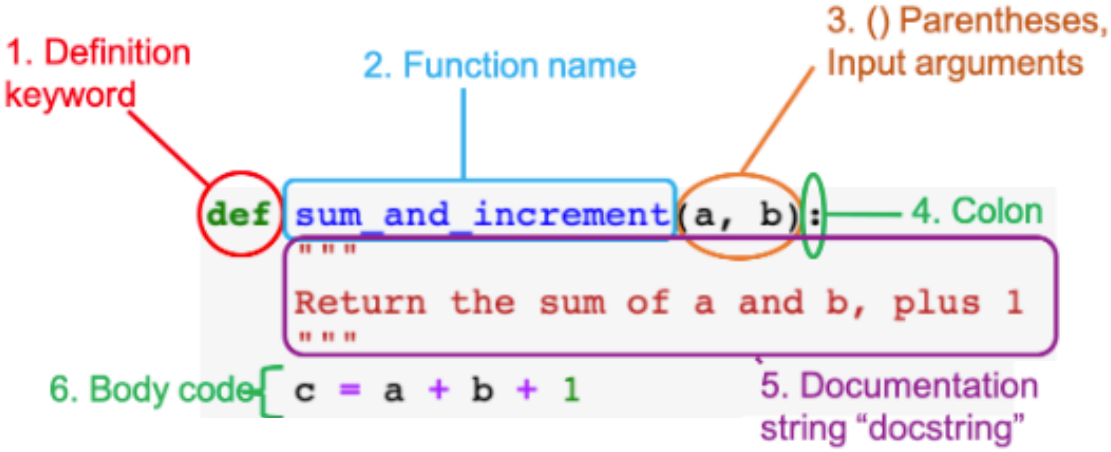


Functions have to be <u>defined</u> **before** called!

Function definition checklist

A function is **declared** using:

1. The definition keyword, `def`.
2. A **function name** of your choice.
3. **() parentheses** which optionally contain **arguments** (the *inputs* to the function)
4. **: a colon** character
5. A **documentation string** that says what the function does.
6. The **body code** to be executed when the function is *called*.



Q.11.3.A
Write a function, **is_even**.

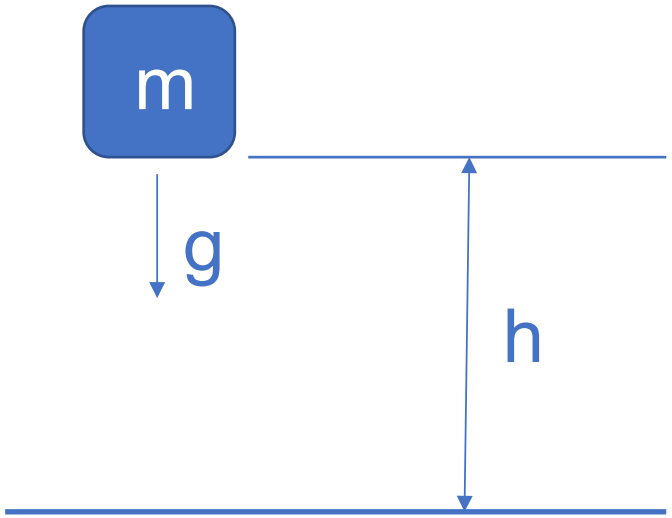
Input:
- n (integer)

Output:
- True if n is even
- False if n is not even

Q.11.3.B
Write a function to calculate gravitational potential energy of an object:

Input:
- m (mass)
- g (acceleration due to gravity)
- h (height)

Output:
- gravitational potential energy, E = mgh



Multiple arguments

```
# sums all items passed in as an arg
def SumItems(*Args):
    Sum = 0
    for Item in Args:
        Sum += Item
    print(Sum)
```

Python puts all arguments in a tuple

```
SumItems2(1,2,3)
>> 6

SumItems2(1,2,3,4)
>> 10
```

Default arguments

```
# prints the product of the 3 arguments
def product(x, y=2, z=10):
    print(x * y * z)
```

y and z have default values

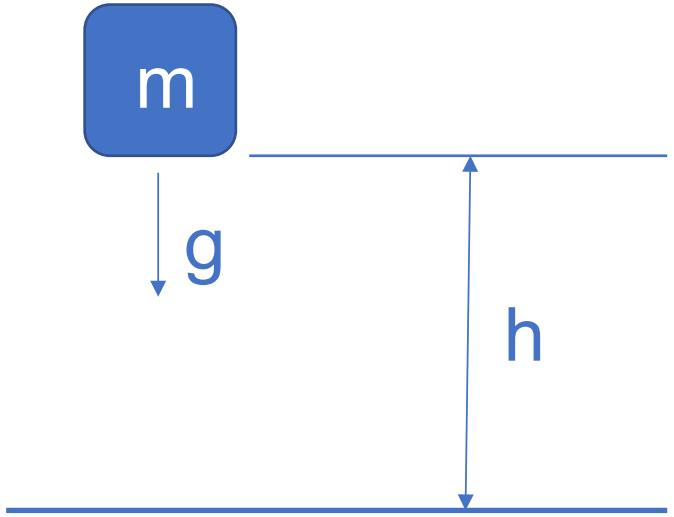
```
product(1)
>> 20

product(3,z=4)
>> 24
```

Q.11.3.C

a) Write a function to calculate gravitational potential energy of an object, **assuming g=9.81ms^{-2}** .

(Rewrite your answer to Q11.3.B)

b) Write a function to calculate gravitational potential energy of an object of **mass 1kg**, **assuming g=9.81ms^{-2}**.



Q.11.3.D

Write a function to find the magnitude of an n-dimensional vector:

$$\|\boldsymbol{a}\| = \sqrt{a_1^2 + a_2^2 + \ \ldots + a_n^2}$$

Local and global variables

Global variables
accessible anywhere

```
global_var = "Global variable"

def my_func():
    print(global_var)

my_func()
```

```
Global variable
```

Local variables
accessible within function only

```
def my_func():

    local_var = "Local variable"
    print(local_var)

my_func()
```

```
Local variable
```

```
def my_func():
    global global_var
    global_var = "Global variable"

my_func()
print(global_var)
```

```
Global variable
```

```
print(local_var)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-3bfff76e6cd3> in <module>
----> 1 print(local_var)

NameError: name 'local_var' is not defined
```

Reading/Writing files

Functions for reading and writing to external files: **open, read, write, close**

open(file_path, mode_specifier)

Mode Specifiers

`r` : open an existing file to read

`w` : open an existing file to write to.
If no file exists: creates a new file.
If file exists : over-writes previous contents.

`a` : open an existing file to write to.
If no file exists: creates a new file.
If file exists : appends text to end of file.

`r+` : open a text file to read from **or** write to.
File must already exist.
If file exists : over-writes previous contents.

`w+` : open a text file to read from **or** write to.
If no file exists: creates a new file.
If file exists : over-writes previous contents.

`a+` : open a text file to read from **or** write to.
If no file exists: creates a new file.
If file exists : appends text to end of file.

Reading Files

scores

```
Hemma 33
Farhad 44
Manesha 80
Hannah 67
Sajid 50
```

Open returns an iterable object

```
file = open("scores.txt", "r")

for line in file:
    i = line.split()
    print(i)

file.close()
```

Remember to close the file!

```
['Hemma', '33']
['Farhad', '44']
['Manesha', '80']
['Hannah', '67']
['Sajid', '50']
```

Writing Files

Open returns an iterable object

```
scores = {"Keren " : 80,
          "Daniel "  : 67,
          "Maya "    : 50
          }

file = open("scores.txt", "a")

for k, v in scores.items():
    file.write( k + str(v) + "\n" )

file.close()
```

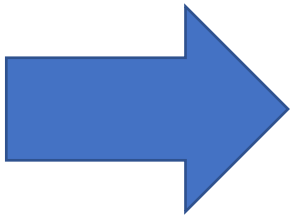
Remember to close the file!

scores

```
Hemma 33
Farhad 44
Manesha 80
Hannah 67
Sajid 50
Keren 80
Daniel 67
Maya 50
```

Closing Files Automatically

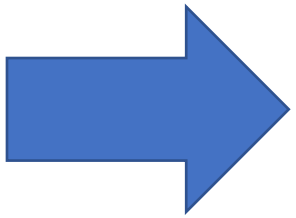
```
file = open("scores.txt", "r")

for line in file:
    i = line.split()
    print(i)

file.close()
```



```
with open("scores.txt", "r") as file:

    for line in file:
        i = line.split()
        print(i)
```

File closes automatically at end of indented code block

```
scores = {"Keren " : 80,
          "Daniel "  : 67,
          "Maya "    : 50
          }

file = open("scores.txt", "a")

for k, v in scores.items():
    file.write( k + str(v) + "\n" )

file.close()
```



```
with open("scores.txt", "a") as file:

    for k, v in scores.items():
        file.write( k + str(v) + "\n" )
```

File closes automatically at end of indented code block