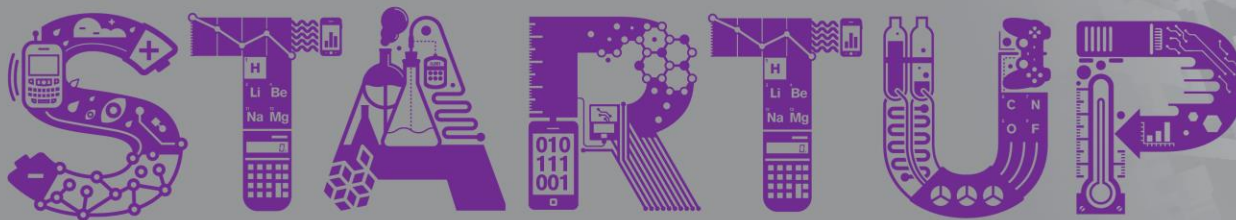


Bloomberg



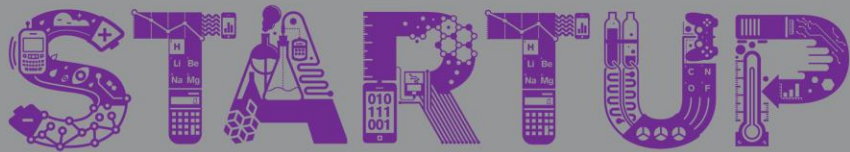
Copyright 2018 Bloomberg L.P.

Licensed under Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0)

Non-commercial use only. Modifications must not be distributed.

Please see <http://creativecommons.org/licenses/by-nc-nd/4.0/>

Bloomberg



Python Lesson 6

Functions



Review

Write code that generates a random number between 1 and 100 and prints the letter grade associated with it (so if the number is between 90 and 100, print A)

```
import random
score = random.randint(1,100)
if score >= 90:
    print('A')
elif score >= 80:
    print('B')
elif score >= 70:
    print('C')
elif score >= 60:
    print('D')
else:
    print('F')
```



Review: Anatomy Of A Function

Functions have names

We can input data, called **parameters**, into functions. Parameters are passed in between parentheses.

```
answer = len("How long is this question?")
```

Functions can **return** data, which we can store in variables to use later.



Functions: Introduction

- Suppose we want to convert 20° Celsius to Fahrenheit:

```
temp_in_f = (20*9/5) + 32
```

- If we want to convert more temperatures, we'll have to copy and paste this.
- It would be nice if there was a conversion function we could use instead.



Writing a Function: Step 1

- Start with code that does what you want your function to do once.

```
temp_in_f = (20*9/5) + 32
```

- Ask yourself: What part of this code should the user be able to **CHANGE** when they run it?
- When we convert from C to F, the temperature in Celsius should be able to change.
- The temperature in Celsius will be a **parameter**.



Writing a Function: Step 2

- Refactor your code to use variables for things that should change:

```
temp_in_c = 20
temp_in_f = (temp_in_c*9/5) + 32
```

- Ask yourself: What should the user GET BACK when they run this function?
- The user should get back the temperature in Fahrenheit.
- The temperature in Fahrenheit will be the **return value**.



Writing a Function: Putting it Together

```
def convert_C_to_F(temp_in_c):  
    temp_in_f = (temp_in_c*9/5) + 32  
    return temp_in_f
```




Functions have names

You define the parameters that can be passed in

The `:` tells the program you are starting the code for the function

def tells the program you are defining a function

Anatomy Of A Function

```
def convert_C_to_F(temp_in_c):  
    temp_in_f = (temp_in_c*9/5) + 32  
    return temp_in_f
```

Function code must all be indented the same amount

You can **return** one value at the end of your function.

Inside the function, the parameter works like a variable



Calling your New Function!

```
def convert_C_to_F(temp_in_c):  
    temp_in_f = (temp_in_c*9/5) + 32  
    return temp_in_f  
  
room_temp = convert_C_to_F(20)  
boiling = convert_C_to_F(100)  
very_cold = convert_C_to_F(-20)  
print(room_temp, boiling, very_cold)
```



Example 2

- Suppose we want to write code to draw an equilateral triangle:

```
turtle.forward(100)
turtle.left(120)
turtle.forward(100)
turtle.left(120)
turtle.forward(100)
turtle.left(120)
```

- We want to draw a lot of triangles. Let's turn this into a function.

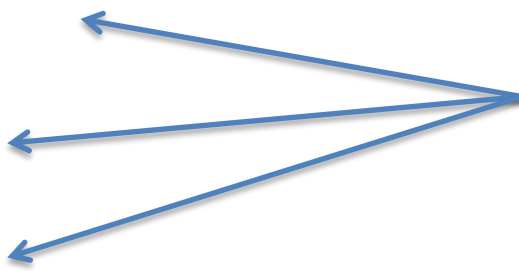


Example 2

What part of this code should be able to **CHANGE** every time we run it?

```
turtle.forward(100)
turtle.left(120)
turtle.forward(100)
turtle.left(120)
turtle.forward(100)
turtle.left(120)
```

Length can change every time.



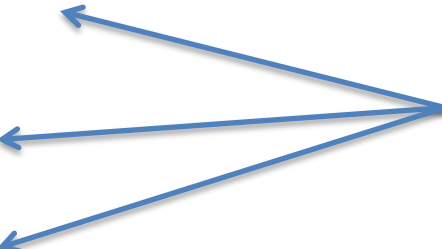


Example 2

Refactor the code to use variables for the things that can change:

```
length = 100  
turtle.forward(length)  
turtle.left(120)  
turtle.forward(length)  
turtle.left(120)  
turtle.forward(length)  
turtle.left(120)
```

Length can change
every time.



Does this function need to return anything?



Example 2

We're ready to convert to a function!

```
def draw_triangle(length):  
    turtle.forward(length)  
    turtle.left(120)  
    turtle.forward(length)  
    turtle.left(120)  
    turtle.forward(length)  
    turtle.left(120)
```

```
draw_triangle(100)
```



Recap

- A function is a reusable, named piece of code.
- To write a function, you need to identify the parameters and the return value.

- Functions start with

```
def your_function_name(parameter_1, parameter_2):
```

- Functions can return values:

```
    return thing_I_want_to_return
```

- Functions will STOP after you return.