Introduction to Computing
Python Functions
Andrew Block

What is a Function?

› You should all be familiar with math functions
  \[ f(C) = \frac{9}{5}C + 32 \]
  \[ f^{-1}(F) = \frac{5}{9}(F - 32) \]

› Python Functions are very similar
  - Can be thought of as a subprogram within our
    Python Program
  - We write sequences of statements within our
    program and give it a name. We can call this
    sequence at any point of our program by referring
    to its name

How do we create a function?

› Start off with a Function Definition
  - The part of the program that creates the function

› Simple Function Example:

  \[ \text{def myFunction():} \]
  \[ \text{print("This is a simple example of a Function")] \]
How do we use our Functions?

- We can use the function we just created by Calling or Invoking it
- We call it by referencing the function
  - myFunction()

Putting the pieces together

def myFunction():
    print("This is a simple example of a function")

myFunction()

- We have our function on Lines 1 and 2 and we call our function on line 4. This simple program prints out “This is a simple example of a Function”
- Note: All statements within our function must be indented!

Passing in Parameters

- If we want to make our program more dynamic, we can use parameters
  - Allows us to pass in arguments to our function
  - Can have more than one parameter

    def myFunction(name):
        print("My Name is " + name)

    myFunction("Andy")

    Result prints out “My Name is Andy"
Function Syntax

- `def <name> (<parameters>):`
  - `<Body of the Function>`

- Multiple parameters separated by a comma
  - `def example(parameter1, parameter2):`
    - `print(parameter1 * parameter2)`
  
  `example(5, 4)` —— Results in 20 being printed out

- The name of the function must be an identifier

Function Details

- The calling point of the function suspends execution of the program at that point
- The parameters of the function get assigned by the values assigned in the call
- When the function completes, it returns to the point where it was called.

The Return Statement

- The return statement allows us to get results from our function and it gets sent back to the caller expression.
  - `def example(parameter1, parameter2):`
    - `return parameter1 * parameter2`
  
  `print(example(5, 4))` —— prints out 20

- When Python encounters a `Return`, it exits the function at that point
- Multiple return statements can exist in one function
Why do we use Functions?

- Functions reduce code duplication
- As our programs get more complex, functions make it easier to make sense of them

Writing your own Functions (Part 1)

- Without using the Computer, write a simple function on a sheet of paper that accomplishes the following:
  - Create a function definition and give your function a name
  - Create two strings (stringA, stringB) of random text
  - Print out the concatenation of the two strings
  - Make sure to call the function

- When you are completed, check your answer using the Python IDLE on the computer

Writing your own Functions (Part 2)

- Working with your Partner, you will create three functions that will print out information about your group:
  - Create a def main() function to call your teamInfo function
  - Create a function called teamInfo
    - The function will take in 3 parameters
    - 2 Strings. One for each member of the group
    - 1 Integer. Your group number
    - Print out the name of each group member and the Tablet PC Password
  - Create a second function called tabletPassword
    - The function will take in 1 parameter
    - 1 Integer. The Group Number
    - Return the Password used on the Tablet PC’s
Pseudocode

main()

	Call your teamInfo method and pass in the necessary parameters

teamInfo(parameters)

	Print out the team number and the names of the group members by using that parameters

	Print out the password by calling the tabletPassword function and passing in the group number into the tabletPassword function from the group number parameter in this function

	tabletPassword(parameters)

	Return the password by multiplying the parameter by 2 and appending it to the string "HP"

Your program should output something similar to this

Team <team number>: Member1, Member2
Tablet Password: <password>  Eg: HP14

One group member submit the assignment on Blackboard.
Make sure to include both Names as comments in the Python program