Lecture 5

Functions

Announcements

1. There will be a practice midterm next **Tuesday** where you will be given 1 hour to complete it
2. The last 30 mins we will go over the answers
3. Midterm will contain everything that we have covered so far including this week's material
    a. Basic data types and syntax
    b. Conditionals and loops
    c. Data structures
    d. Functions

What is a Function?

Functions help reduce redundancy in our code.

Functions can take in arguments called **parameters**, and can **return** a desired object although you can **still have a function that doesn’t take any parameters nor does it need to return any value.**

Note that functions can take in **multiple** arguments but can only return **a single** object/value.

What is the difference between a Function and a Method?

From what we have worked with so far, we have seen built in functions such as len(), print(), input(), etc. and methods such as .append(), .remove(), etc.

To explain it in simple terms, methods have a “.” in front of it and they are a function but they **belong** to an object type.

We won’t be making methods yet but it’s good to understand the differences and limitations of what we can and can’t do yet.

For example I have a list called myList and a dictionary called myDict. .append() only works on myList since .append() is a method that belongs to List objects and not dictionaries.

You may be wondering what about .clear()? .clear() can be used on basically all built in data structures! That is because .clear() was built into every built in data structure, but .append() isn’t.

What is the difference between a Function and a Method?

Functions

- **Generally** defined on the global scope
- Can't directly modify the parameters that we pass into it as arguments. (in 99% of cases)
    - Example: we can't make a clear function and say clear(myDict) to make myDict empty
- There is no passing by reference or const functions in python
- Can be made to support any type of objects that you want to pass into it and aren't made for one type of object and that type only
- Unlike statically typed languages we can pass in any type for the argument as we don't have to specify the parameter types as a result we have to write good comments and annotations so we know what can and can't be passed in

Methods

- **Note: don't worry about this now it's just for your general information**
- Defined within the scope of a certain object therefore it can only be accessed by and used on a specific object type
- Can directly modify the object that it is being called on
- **Can't** write a single method to have it support multiple object types, has to be defined multiple times with every object
- Unlike statically typed languages we can pass in any type for the argument as we don't have to specify the parameter types as a result we have to write good comments and annotations so we know what can and can't be passed in

How do you write a function?

General Format

```
def myFunction( parameter(s) ):
    Task
```

Passing in parameters

C++ Example:

```
//doubles are like floats in python
double add(double x, double y)
  {
    return x + y;
  }
```

Python:

```
1  def add(x, y):
2      return x + y
```

As you can see, since python is dynamically typed we don't have to state what type the parameters and returned value have to be.

Therefore, we can use this function with any combination of object types and this is a good and a bad thing.

In this case, it's good that we don't have to write multiple add functions for strings, ints, floats, etc. but we sure don't want someone to pass in dictionaries or named tuples into our add function

Writing Useful Comments for Our functions

You don’t have to understand what this is doing its just an example of my code from something i'm working on. Since functions can take in any type in python it’s important to make useful comments about what your functions do so that other people who are using/reading your code know how to use it properly

```
"""Takes in a phone number as a string and returns a bool
   depending on if the phone number is valid or not"""
def checkNumber(number):
    phone = re.compile("^\s*(?:\+?(\d{1,3}))?[-. (]*(\d{3})[-. )]*(\d{3})[-. ]*(\d{4})(?: *x(\d+))?\s*$")
    if phone.match(number):
        return True
    return False

"""Takes in a phone number as a string and returns the phone
   number as a string formatted with only digits"""
def formatNumber(number):
    return "".join([i for i in number if i in [str(i) for i in range(10)]])
```

Using Type Annotations

Although you can have type annotations just like statically typed languages in python they are nothing more than a **suggestion**. Even if the type is annotated to be a string I can still pass in a list, integer, etc.

I personally don't use this as comments serve the same purpose and are easier to type than these special characters

```
"""Checks to make sure the number is valid"""
def checkNumber(number: str) -> bool:
    phone = re.compile("^\s*(?:\+?(\d{1,3}))?[-. (]*(\d{3})[-. )]*(\d{3})[-. ]*(\d{4})(?: *x(\d+))?\s*$")
    if phone.match(number):
        return True
    return False


"""Makes all the inputted numbers uniform"""
def formatNumber(number: str) -> str:
    return "".join([i for i in number if i in [str(i) for i in range(10)]])
```

Scope Revisited

Just like how we discussed in lecture 2, scope is how code is organized in what each statement can and can't access

Anything defined in the global scope (no indentation) can be accessed from anywhere in the program after its defined

Anything defined within a function will remain in that function's scope and cannot be accessed anywhere else

```
1  def myFunction1():
2      function_1Variable = 1
3
4  def myFunction2():
5      myFunction3() #This doesnt work, 3 isnt defined yet
6
7  def myFunction3():
8      print(function_1Variable) #This wont work
9      myFunction1() #This works
10     myFunction2() #This works
```

What is “pass”

Pass simply means *skip for now* or *don’t do anything*

We can use pass as either a temporary placeholder or if we want a function or task to do nothing

Without pass the program won’t run so we use it while we are mapping out what functions and features we need in a project

This program won’t run and will throw an exception

```
1  def myFunction1():
2      if True:
3          print("hello")
4
5  def myFunction2():
6
7
8  def myFunction3():
```

This one will

```
1  def myFunction1():
2      if True:
3          print("hello")
4
5  def myFunction2():
6      pass
7
8  def myFunction3():
9      pass
```

```
1  def myFunction1():
2      if True:
3          pass
4
```

Can be used on conditionals too

Returned vs Print

A print statement is like saying a message to the console or monitor

It does not end the function/program

```
1  #My function
2  def myFunction():
3      print("I like the Honda s2000")
4
5  #My function call
6  myFunction()
```

A return statement will return an object that you can use to assign to a variable, be passed into another argument, save for later use, etc.

When a function returns a value it, **the function will stop running**

```
1  #My function
2  def myFunction():
3      return "I like the Honda s2000"
4
5  #My function call
6  #Doesn't print anything cause the function returns a string
7  #and were not doing anything with the string that's being returned
8  myFunction()
9
10 #Prints the returned value from the function call
11 print(myFunction())
```

Returned vs Print (built in functions example)

We have seen that the built in function len() **returns** the length of a container.

This returned value can be assigned to variables, passed into conditionals and functions as arguments, etc.

We can’t see what the value of len() is unless we print it to display it to the console

The only example of a built in function that does not return a value is the print() function itself.

This function takes in arguments and will output them to the console but that’s it. We can’t use the outputs of the print statement to do anything more in our program

You may be thinking about the input() function but the input function returns the string that the user typed

What if I don’t have a return statement in my function

Functions will terminate / stop running when it reaches a return statement.

If we have a function with no return statement it will run until it reaches the bottom and returns None.

Similar to null or void in other languages

```
1  def myFunction():
2      x = 4
3      y = 3
4      z = x
5
6  print(myFunction())
```

```
None
```

Passing in arguments / parameters

The parameters that we specify in the function declaration is similar to the variable that we state when we make a for loop. (for variable in object)

Parameters serve as temporary place holders for theoretical values and objects you plan to pass into the function and these parameters only exist within the scope of the function call itself

The variables x, y and z that are defined globally don't affect the parameters of myFunction even though the parameters are also named x, y and z

```
1  x = 1
2  y = 2
3  z = 3
4
5  def myFunction(x, y, z):
6      return [x, y, z]
7
8  print(x, y, z)
9  print(myFunction(0, 9, 8))
```

```
1 2 3
[0, 9, 8]
```

Functions Practice #1

```
1  def myFunction(a):
2      a = 8
3      return a
4
5  a = 20
6  first(a)
7  print(a)
```

What does this program output?

A. 8
B. 20
C. Error, a can’t be assigned twice
D. None
E. 0

Functions Practice #1 - Solution

```
1  def myFunction(a):
2      a = 8
3      return a
4
5  a = 20
6  myFunction(a)
7  print(a)
```

A. 8
B. 20
C. Error, a can't be assigned twice
D. None
E. 0

The program prints 20 since myFunction returns a value but it isn't printed

Functions Practice #2

```
1  def func1():
2      return 5
3
4  def func2():
5      print(5)
6
7  def func3():
8      return print(5)
```

Which assigns x to 5?

A. x = func1()
B. x = func2()
C. x = func3()
D. All of the above

Functions Practice #2 - Solution

```
1  def func1():
2      return 5
3
4  def func2():
5      print(5)
6
7  def func3():
8      return print(5)
```

Which assigns x to 5?

A. x = func1()
B. x = func2()
C. x = func3()
D. All of the above

Only the first function returns a value that we can use to assign to a variable

Functions Practice #3

```
1  def calculation(x, y, z):
2      a = y
3      b = x + 1
4      return a + b + 3
5
6  print(calculation(3,2,5))
```

What does this program output?

A. 5
B. 9
C. 0
D. 3

Functions Practice #3 - Solution

```
1  def calculation(x, y, z):
2      a = y
3      b = x + 1
4      return a + b + 3
5
6  print(calculation(3,2,5))
```

What does this program output?

A. 5

B. 9

C. 0

D. 3

Functions Practice #4

```
1  def func1(x, y):
2      x = x + 1
3      y += 1
4      print(x * y)
5
6  def func2():
7      x = 2
8      y = 4
9      func1(x, y)
10     print(x * y)
11
12 func2()
```

What does this program output?

A. 8 8
B. 15 15
C. 8 15
D. 15 8
E. None of the above

Functions Practice #4 - Solution

```
1  def func1(x, y):
2      x = x + 1
3      y += 1
4      print(x * y)
5
6  def func2():
7      x = 2
8      y = 4
9      func1(x, y)
10     print(x * y)
11
12 func2()
```

What does this program output?

A. 8 8
B. 15 15
C. 8 15
D. 15 8
E. None of the above

Function Practice #5

```
1  def func1(x, y):
2      a = x
3      x = y
4      if a == x:
5          y = a
6      else:
7          y = x
8      return y
9
10 print(func1(2,3))
```

What does this program output?

A. 2
B. 3
C. Y
D. None of the above

Function Practice #5 - Solution

```
1  def func1(x, y):
2      a = x
3      x = y
4      if a == x:
5          y = a
6      else:
7          y = x
8      return y
9
10 print(func1(2,3))
```

What does this program output?

A. 2
B. 3
C. Y
D. None of the above

Do it yourself: Function Practice #1

The built in function sum() takes in a list as a parameter and returns the sum of all values in the list.

Write a function called mySum() which does the same thing as the built in function sum(). You can assume that the list that will be passed into mySum() will be all integers.

Solution:

Do it yourself: Function Practice #1 - Solution

The built in function sum() takes in a list as a parameter and returns the sum of all values in the list.

Write a function called mySum() which does the same thing as the built in function sum() without using the sum() function.

You can assume that the list that will be passed into mySum() will be all integers.

Solution:

```
def mySum(array):
    count = 0
    for i in array:
        count += i
    return count
```

Do it yourself: Function Practice #2

Write a function print_longest which takes in two parameters, both strings and returns the string that is the longest.

If both strings are the same length than return both strings.

Solution:

Do it yourself: Function Practice #2 - Solution

Write a function print_longest which takes in two parameters, both strings and returns the string that is the longest.

If both strings are the same length than return both strings.

Solution:

```
def print_longest(string1, string2):
    if len(string1) == len(string2):
        return (string1, string2)
    elif len(string1) > len(string2):
        return string1
    else:
        return string2
```

Do it yourself: Function Practice #3

Write a function checkDup() that takes in a list and returns True if the list has no duplicates and False if it does.

You can not implicit type convert the list to a set and back to a list (Never ever do this)

Solution:

Do it yourself: Function Practice #3 - Solution

Write a function checkDup() that takes in a list and returns True if the list has no duplicates and False if it does.

You can not implicit type convert the list to a set and back to a list (Never ever do this)

Solution:

```
def checkDup(array):
    new = []
    for i in array:
        if i not in new:
            new.append(i)
    if len(new) == len(array):
        return True
    else:
        return False
```

Do it yourself: Function Practice #4

Write a function distinctSum() which takes in a list of integers and returns a sum of all numbers that are unique.

Input: myList = [10,20,20,10,30,10]

Output: 60

Solution:

Do it yourself: Function Practice #4 - Solution

Write a function distinctSum() which takes in a list of integers and returns a sum of all numbers that are unique.

Input: myList = [10,20,20,10,30,10]

Output: 60

Solution:

```
def checkDup(array):
    new = []
    for i in array:
        if i not in new:
            new.append(i)
    return sum(new)
```

Do it yourself: Function Practice #5

Write a function listIntersection() which takes in two lists and returns a list of items that appears in both lists.

Input: [1, 3, 5, 2] , [3, 0, 9, 1]

Output: [1, 3]

Solution:

Do it yourself: Function Practice #5 - Solution

Write a function listIntersection() which takes in two lists and returns a list of items that appears in both lists.

Input: [1, 3, 5, 2] , [3, 0, 9, 1]

Output: [1, 3]

Solution:

```
def listIntersection(array1, array2):
    new = []
    for i in array1:
        if i in array2:
            new.append(i)
    return new
```