Isepp Rebane
5/20/2023
Foundations of Programming: Python (IT FDN 110 A Sp 23)
Assignment 06

Functions & Classes

Introduction

The script that I created this week, consists of 180 lines of code. If not for functions, and classes, it could be possibly double! This write up is about classes and functions, both are great separation of concern topics.

Functions

A function is a way to group one or more statements together into a single block of code, that can be called multiple times.

```
Def print_some_strings():
     print (“A”
     print (“Some”)
     print (“Strings”)
```

Parameters

Functions can be created with, or without parameters. A parameter is a value that can be passed into a function to be processed.

```
Def print_name(username):
     print (name)

print_name(“Isepp”)
```

Variables → Argument

Using variables as an argument is common in programming. Especially if you want to access certain data multiple times in a script. Basic example bellow.

```
firstname = None #Argument1
lastname = None #Argument2

def new_employee(firstname, lastname):
     username = firstname + lastname
     print (username)

firstname = input(lower(“What is your first name?”))
lastname = input(lower(“What is your last name?”))
print (“You’re new username is:”, username)
```

```
$You're new username is: isepprebane
```

Return vales

With functions, you can have one or more values returned after it has been called. Return values make a function act like an expression.

```
def new_employee():
     return username

employee = username
```

Returning referenced data

With Python, you can specify data that is referenced elsewhere. Sound confusing?

```
# Simple data
x = 1
y = x
x = 4
print(y) #Prints 1 if y is called.

# Complex data
x = [1, 2]
y = x
x[0] = 3
print(y) # Prints [3,2], expected of a reference.

# Simple data
x = "Isepp"
y = x
x = "Rebane"
print(y) # Print "Isepp", as if only the value was passed.

# Complex data
x =["John", "Appleseed"]
y = x
x[0] = "Robert"
print(y) # Prints ['John', Appleseed], expected reference.


main_function(p1:float by val, p2:float by ref):
     return 'data'
```

Differences between global and local variables

Local: Variables declared in a function are 'local' and cannot be accessed outside of said function.

Global: Variables declared in the main script body are considered global, and can be used anywhere.

Shadowing: If you use a global variable inside of a function, watch out. If you want to use a global variable, use the "global" keyword, or your local variable will be "shadow" the local variable. Remember, a variable can only have one value assigned at a time.

Doc + String = DocString

When creating functions, you want to know what they do. Similar to a comment, when you call a function, a quick description pops up to show you how to use it.

```
def date_month_year (val1, va2, val3):
     """ This function combines 3 values to make a DOB """
     Return val1 + val2 + val3)
```

Classes

Classes are an organizational method of grouping functions, variables, and constants.

```
class price_process():
     """ Functions to process prices """

     def addprices(val1 = 0.0, val2 = 0.0):
          """ Function adds two values """
     return float(val1 + val2)

     def subtractprices():
          """Function to subtract prices """
     return float(val1 – val2)

#Process/IO
print(price_process.addprices(7,8))
print(price_process.subtractprices(10,5))
```

Assignment and code description

This week, we made the “To Do” list once again, but this time, we separated the code into classes and functions. Even though we used a template to simulate working with another individual’s code base, which is an important part of programming in a professional setting, this assignment was the most difficult yet. Computers are not open to interpretation; coding is pure logic. Hence, you must be explicit with your desires.

Line 1-10: File header that contains program information.

Line 12-18: Variables and constants being used.

Line 20-80: Processing section. Functions separated for reading and writing data to file, formatting then adding data to lists, and removing data from lists.

Line 81-146: The longest section in the script. Section for the user to input commands into the menu, ask for tasks, and subsequent priority values.

Line 146-180: Main script body. Beginning function takes place to load data from file upon first run. Second part runs in a while loop that shows the current data, prints and the user menu, adds tasks, removes existing tasks, saves the data to a file, or exits the program.

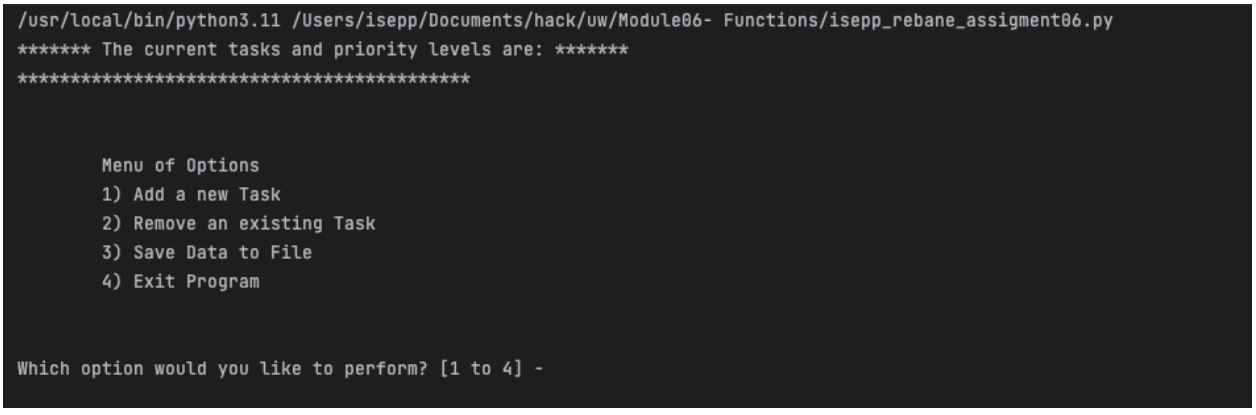

Figure 1. Script running from PyCharm

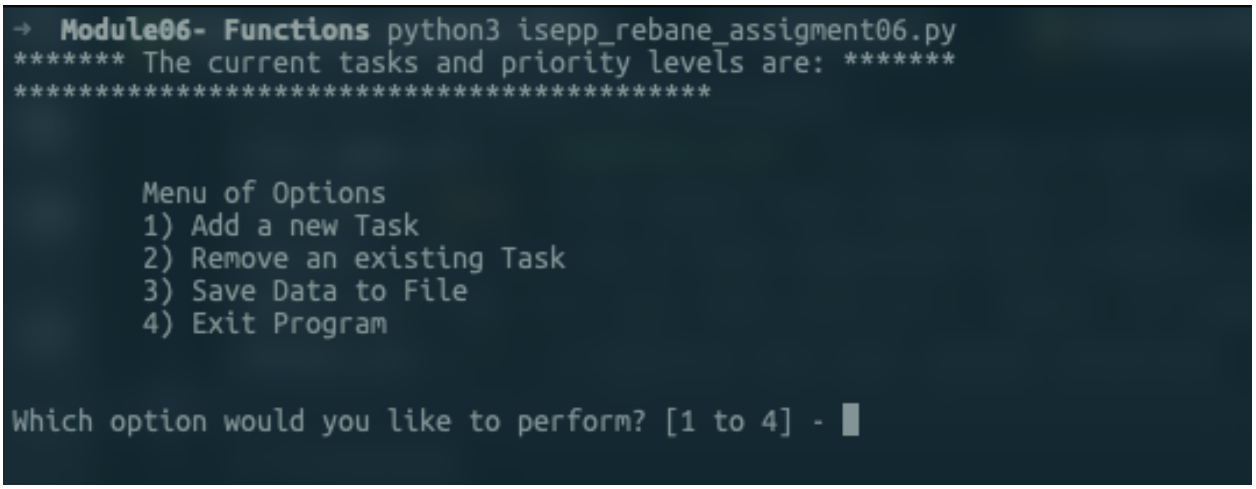

Figure 2. Script running from terminal window.

Summery

- Functions and how they can be grouped.
- Using parameters inside of a function.
- Using variables as an argument that can be passed to a function.
- Return values, referenced data, and how it can be utterly confusing.
- Global variables, local variables, and the pitfalls of shadowing.
- DocStrings, which provide details on functions information.
- Classes.
- This week's assignment, code overview, and screenshots.