**CS506 Programming for Computing**

**HOP03 Functions, Debugging & Exceptions**

10/18/2020 Developed by Kim Nguyen

10/18/2020 Reviewed by Yu-Che Liu

Center for Information Assurance (CIAE) @City University of Seattle (CityU)

 

**Before You Start**

* The directory path shown in screenshots may be different from yours.
* Some steps are not explained in the tutorial**.** If you are not sure what to do:
  1. Consult the resources listed below.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

* Learn ways to pass information to functions
* How to write certain functions whose primary job is to display information and other functions designed to process data and return a value or set of values.
* Learn to store functions in separate files called modules to help organize your main program files.
* How to use debugger tools

**Resources**

Matthes, E. (2019). [Python Crash Course: A Hands-On, Project-Based Introduction to Programming, 2nd Edition](https://login.proxy.cityu.edu/sso/skillport?context=146803). No Starch Press. (ISBN 9781593279288)

**Preparation**

1. In Visual Studio Code, open the private repository generated when you accepted the HOP03 assignment (If you cannot find that repository in your machine, you might have not cloned the repo, if so, please do before proceeding).

A screenshot of a cell phone

Description automatically generated

Open the terminal from the VSCode by hitting the control + ~ key, navigate into Module 1 folder using the following command:

>>> cd Module 3

**Python Functions**

A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for your application and a high degree of code reusing.

A function can be written to as a set of statements that take inputs and perform specific computation and produces output.

**Syntax:**

def functionname( parameters ):

code

return [expression]

**Note:** The statement return [expression] exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as return None. It is not mandatory to use a return statement.

A function can process some data and then return a value or set of values. The value the function returns is called a return value. The return statement takes a value from inside a function and sends it back to the line that called the function. Return values allow you to move much of your program's grunt work into functions, which can simplify the body of your program.

1. Create a file **print\_numbers.py** and type the following code.

A screenshot of a cell phone

Description automatically generated

In the terminal type **python3 print\_numbers.py** to check the output

A picture containing object, clock

Description automatically generated

A function doesn’t always have to return something, sometime, a function just print something out, it is the same as return None. Example as below (You don’t have to write the below code):

A close up of a screen

Description automatically generated

Result:



**Passing Arguments**

Because a function definition can have multiple parameters, a function call may need multiple arguments. You can pass arguments to your functions in a number of ways.

Positional arguments: which need to be in the same order the parameters were written

Keyword arguments: where each argument consists of a variable name and a value

Arbitrary Number of Arguments: Sometimes you won't know ahead of time how many arguments a function needs to accept. In that case arbitrary can be used.

1. Create a file **positional.py** and type the following code.

A screenshot of a cell phone

Description automatically generated

In the terminal type **python3 positional.py** to check the output

A picture containing drawing

Description automatically generated

The definition shows that this function needs a type of animal and the animal's name. When we call describe\_pet(), we need to provide an animal type and a name, in that order. For example, in the function call, the argument ‘hamster’ is assigned to the parameter animal\_type and the argument ‘harry’ is assigned to the parameter pet\_name. In the function body, these two parameters are used to display information about the pet being described.

##### The order matters in positional arguments. The parameters specified in the function definition has to be in the same order as how the function is called. There won’t be any error showing when running the code, but the result might be not as you expected it to be.

**Challenge:**

**Add to the code above to receive the following result:**

A picture containing drawing

Description automatically generated

**Arbitrary number of arguments**

Sometimes you'll want to accept an arbitrary number of arguments, but you won't know ahead of time what kind of information will be passed to the function. In this case, you can write functions that accept as many key-value pairs as the calling statement provides.

1. Create a file **arbitrary\_dict.py** and type the following code

A picture containing drawing

Description automatically generated

In the terminal type **python3 arbitrary\_dict.py** to check the output



The “menu” is the definition where user can pass the item and quantity, then it allows the user to pass in as many name-value pairs as they want. The double asterisks before the parameter \*\*restaurant cause Python to create an empty dictionary called “restaurant” and pack whatever name-value pairs it receives into this dictionary.

**Python Debugging**

Create DuplicateZeros.py file, and type the following code:

Text

Description automatically generated

Watch the following video to see how we can debug this program using Debugger tool:

<https://youtu.be/wODNZqYU8fU>

Let’s change the while loop to for loop and see what happens:

Text

Description automatically generated

Clearly, the result is totally different.

**Challenge:**

**Explain why using for loop in the above program does not work.**

***Hint: Use Debugger Tool***

**Push your work to GitHub**

Open the terminal from the VSCode by hitting the “control” + “~” key and type the following command:

>>> git add .

>>> git commit -m “Submission for Module 3 – Your Name”

>>> git push origin master