**CS506 – Programming for Computing**

**HOS01 Introduction to Python Pt1**

01/03/2025 Reviewed by Jonathan Koerber

03/13/2025 Reviewed by Clark Ngo

04/03/2025 Reviewed by Naveena Moddu

School of Technology & Computing (STC)

City University of Seattle (CityU)

**Before You Start**

* **Screenshots may be different from your environment.**
* The directory path shown in screenshots may be different from yours.
* There might be subtle discrepancies along with the steps. Please use your best judgment while going through this cookbook-style tutorial to complete each step.
* Some steps may not be explained in detail. If you are not sure what to do:

1. Consult the resources from the course.
2. If you cannot solve the problem after a few tries (usually 15 -30 minutes), ask a TA for help.

**Resources**

* Python crash course: a hands-on, project-based introduction to programming: Matthes, E. (2019): [Available online link](https://cityu.alma.exlibrisgroup.com/discovery/openurl?institution=01CITYUNIV_INST&rfr_id=info:sid%2Fsummon&rft_dat=ie%3D5152833400004251,language%3DEN&svc_dat=CTO&u.ignore_date_coverage=true&vid=01CITYUNIV_INST:Services)

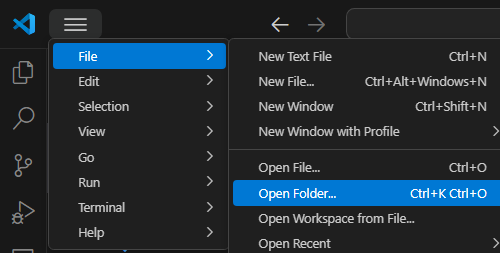
**Learning Outcomes**

* **Section 1: Preparing your environment**
* **Section 2: Variables, Data Types and User Input**
* **Section 3: Multiple assignments**
* **Section 4: String functions and Concatenation**
* **Section 5: Numbers and operators**
* **Section 6: Using Python for Decision Making**
* **Section 7: Using Loops in lists**
* **Section 8: Dictionary Data Type**
* **Section 9: Looping in Dictionary**

**Section 1: Preparing your environment**

Get started with your virtual environment here: <https://cityuseattle.github.io/docs/git/github_codepsace/#codespaces>

* 1. In Codespaces, open the private repository generated when you accepted the HOS01 assignment (If you cannot find that repository in your machine, you might have not cloned the repo, if so, please do before proceeding).



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

**Section 2: Variables, Data Types and User Input**

**Follow the video instructions below to code along:** <https://youtu.be/5Y9AENc6o2Y>

* Variable: Variables are containers for storing data values. Unlike other programming languages, Python has no command for declaring a variable. A variable is created the moment you first assign a value to it.
* More about variables: <https://www.w3schools.com/python/python_variables.asp>
* Data Types: Data types are the classification or categorization of data items. Data types represent a kind of value which determines what operations can be performed on that data. Numeric, non-numeric and Boolean (true/false) data are the most used data types.
* More about data types: <https://www.w3schools.com/python/python_datatypes.asp>
* User Input: To receive information through the keyboard, Python uses the input() function
* When the input() function is called, the program flow stops until the user enters the input via the command line. To enter the data, the user needs to press the ENTER key after inputting the string. The input() function, by default, will convert all the information it receives into a string.
* More about user input: <https://www.w3schools.com/python/python_user_input.asp>

**Section 3: Multiple assignments**

Python allows you to assign values to multiple variables in one line andyou can assign the same value to multiple variables in one line. For example:

A screenshot of a cell phone

Description automatically generated

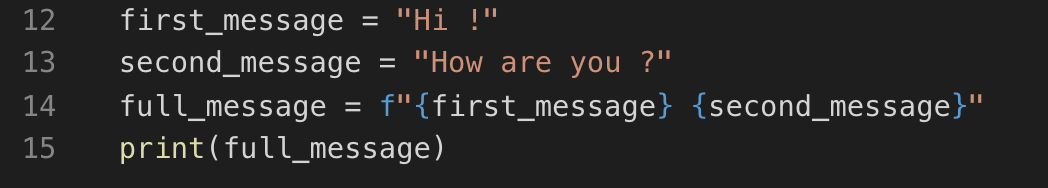
*(Note: this is optional, you do not have to write the code in the above screenshot)*

**Section 4: String functions and Concatenation**

**Follow the video instructions below to code along:** <https://youtu.be/9zW8GEogRKc>

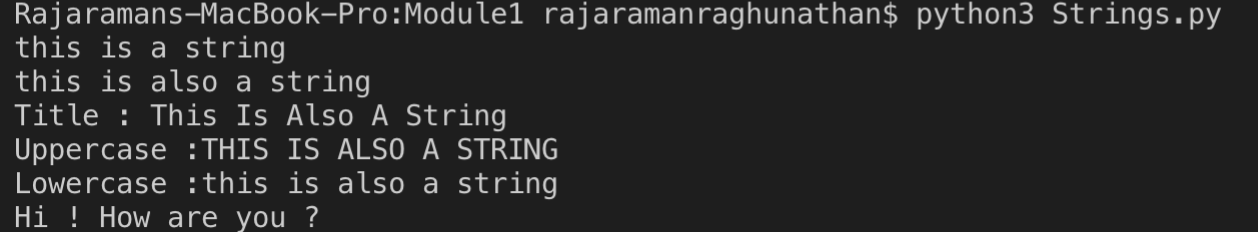
In Python, there are a few ways to concatenate – or combine - strings. The new string that is created is referred to as a string object. Obviously, this is because everything in Python is an object – which is why

Python is an object–oriented language. Another way to concatenate strings is as below example:



*(Note: this is optional, you do not have to write the code in the above screenshot)*

Output:



Explanation:

* Also called “formatted string literals,” f-strings are string literals that have an f at the beginning and curly braces containing expressions that will be replaced with their values. The expressions are evaluated at runtime and then formatted using the \_\_format\_\_ protocol.

**Section 5: Numbers and operators**

Operators are the constructs which can manipulate the value of operands. Consider the expression 4 + 5 = 9. Here, 4 and 5 are called operands and + is called operator.

Python language supports the following types of operators.

* + Arithmetic Operators
  + Comparison (Relational) Operators
  + Assignment Operators
  + Logical Operators
  + Bitwise Operators
  + Membership Operators
  + Identity Operators
  + The commonly used operators are underlined above.
  + More about numbers and operators: <https://www.tutorialspoint.com/python/python_basic_operators.htm>

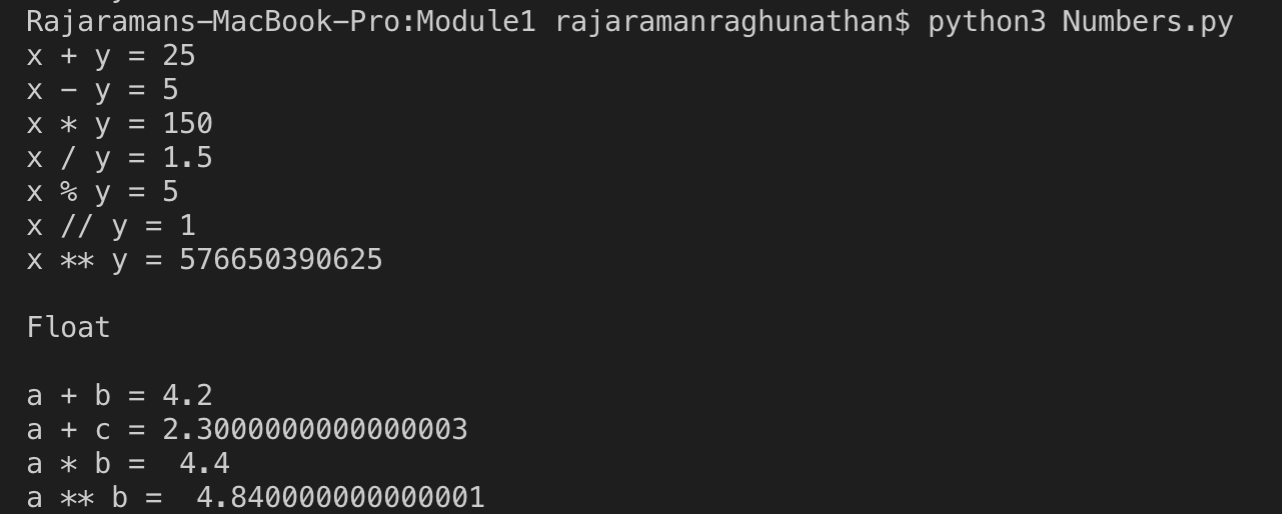
1. Under module1, create a file **Numbers.py** and type the following code.

A close up of a black background

Description automatically generated

2. Type the following to get the output of the code in the terminal

python Numbers.py



If you look at the answer of a+c and a\*\*b, it is displayed as an arbitrary number of decimal places. This happens in all languages and is of little concern. Python tries to find a way to represent the result as precisely as possible, which is sometimes difficult given how computers must represent numbers internally.

1. **Python Decision making**

Decision making is anticipation of conditions occurring while execution of the program and specifying actions taken according to the conditions.

1. Create a **HOS01\_ELIF.py** file, follow the below video instruction to code along: <https://youtu.be/4rIRG1aMEvM>

**Challenge:** Fix the program so that the user input can be recognized as an integer. Expected result:

Text

Description automatically generated

**Elif:**

The elif keyword is pythons' way of saying "if the previous conditions were not true, then try this condition". Create IfControl.py file in Module 1 folder and type the following code:

A screen shot of a social media post

Description automatically generated

1. Use the following command and give different input to test the program:

python IfControl.py

A close up of a logo

Description automatically generated

A close up of a logo

Description automatically generated

A close up of a logo

Description automatically generated

More about If statements: <https://www.w3schools.com/python/python_conditions.asp>

**Loops**

In general, statements are executed sequentially: The first statement in a function is executed first, followed by the second, and so on. There may be a situation when you need to execute a block of code several times.

Programming languages provide various control structures that allow for more complicated execution paths. A loop statement allows us to execute a statement or group of statements multiple times.

**While loop**

**Follow the below video instruction to code along:** <https://youtu.be/_3665KzFlO8>

Syntax:

1. while expression:

2. statement(s)

statement(s) may be a single statement or a block of statements. The condition may be any expression, and true is any non-zero value.

In Python, all the statements indented by the same number of character spaces after a programming construct are part of a single block of code. Python uses indentation as its method of grouping statements.

**How does the while loop work?**

1. The while loop runs as long as, or while, a certain condition is true.

2. The loop iterates while the condition is true.

3. When the condition becomes false, program control passes to the line immediately following the loop.

More about while loops: <https://www.w3schools.com/python/python_while_loops.asp>

**Challenge:** Fix the program so that the attempt of guessing is 3 times instead of 4.

**For loop**

*“for”* loops are traditionally used when you have a block of code which you want to repeat a fixed number of times. The Python *for* statement iterates over the members of a sequence in order, executing the block each time.

Syntax:

1. for iterating\_var in sequence:

2. statements(s)

If a sequence contains an expression list, it is evaluated first. Then, the first item in the sequence is assigned to the iterating variable iterating\_var. Next, the statements block is executed. Each item in the list is assigned to iterating\_var, and the statement(s) block is executed until the entire sequence is exhausted.

**1. Create a ForControl.py file in Module 1 and type the code as below:**

A picture containing drawing

Description automatically generated

Note: we import the random function to generate a random number from 5 to 15 (exclusive) for the end range number.

* 1. In the terminal type the following command:
  2. python ForControl.py

A close up of text on a black background

Description automatically generated

**Section 7: Using Loops in lists**

1. Let’s use a for loop to print out all the elements of a list. Create a file **replaceNegative.py** under Module 1 and enter the following code:

Text

Description automatically generated

The range() function causes Python to start counting at 0 and it stops when it reaches the length of the list. The len() function calculates the length of the list. The i consists of the range index and it is used for navigating throughout the list. If the value in the list is less than 0, that is a negative value then that value is converted to absolute value and returned to list.

* 1. In the terminal type the following command:
  2. python replaceNegative.py

A picture containing meter

Description automatically generated

**Challenge: Edit the program to replace any negative numbers in the list with positive ones. Expected result:**



*Hint: you can use abs() function to turn negative numbers to positive*

**Section 8: Dictionary Data Type**

A dictionary is a collection of many values. Unlike List, indexes for dictionaries can use many different data types, it’s called keys, and a key with its associated value is called a key-value pair. Each key is separated from its value by a colon (:), the items are separated by commas and the whole thing is enclosed in curly braces. Keys are unique within a dictionary while values may not be. The values of a dictionary can be of any type, but the keys must be of an immutable data type such as strings, numbers, or tuples.

In the Dictionary functions like accessing values, updating the dictionary, adding new pairs and deleting the elements can be performed.

1. Under Module1 create a file **dictionary.py** and type the code below. This program describes how to access, update, add and delete items in a dictionary.

A screen shot of a smart phone

Description automatically generated

1. In the terminal type the following to check the output for the above code

python dictionary.py

A picture containing drawing

Description automatically generated

*Note: If the key you ask for doesn't exist, you'll get an error. More than one entry per key not allowed. Which means no duplicate key is allowed. When duplicate keys encountered during assignment, the last assignment wins.*

Dictionary is unordered. Thus, if you compare two dictionaries with the same content but not the same order, it will return true.

1. Let’s find out what will happen when we compare two lists and dictionary with similar values but different orders. Under Module 1 create a file **compare.py** and type the code below.

A screenshot of a cell phone

Description automatically generated

1. Type the following in the terminal to check the output of the above code

python compare.py

A picture containing drawing

Description automatically generated

The output on the terminal should be **False** and **True** respectively, since order matters in List but not in Dictionary as long as it contains the same values.

**Section 9. Looping in Dictionary**

A single Python dictionary can contain just a few key-value pairs or millions of pairs. Python lets you loop through a dictionary. You can loop through all a dictionary's key-value pairs, through its keys, or through its values.

1. Under Module 1 create a file **dict\_for.py** and type the code below. This program explains how multiple dictionaries are combined and used inside loop for accessing the key and value.

A screenshot of a cell phone

Description automatically generated

Type **python dict\_for.py** in the terminal for checking the output.

A close up of a logo

Description automatically generated

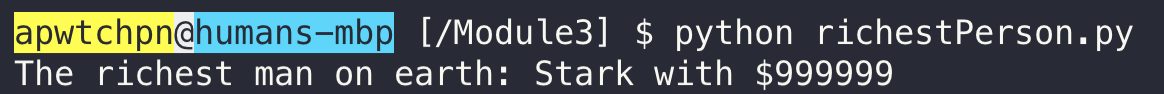
Note: For accessing only keys, use.keys() and for values use .values() in the loop.

2) Let’s create a program to find the richest man from the dictionary and save it as **richest.py**

Text

Description automatically generated

Type “python richest.py” in the terminal. The result would show Stark is the richest man on earth.



**Challenge: Add to the code above to show the person with the lowest income.**

**Push your work to GitHub**

Follow instructions here: <https://cityuseattle.github.io/docs/git/codespaces_submission/>