**Outline**

Access the Python Development environment and follow the tutorial to gain an initial exposure to a programming language. Begin to develop an familiarity with basic programming concepts.

**Objectives**

* Use correct terminology to describe programming concepts;
* Describe the types of data that computers can process and store (e.g., numbers, text);
* Explain the difference between constants and variables used in programming;
* Use variables, expressions, and assignment statements to store and manipulate numbers and text in a program

**Materials**

* Python3 Development Environment at: //repl.it/
* Python Tutorial at: <http://www.letslearnpython.com/learn/>

**Accessing the Python3 Web IDE Environment**

Accessing the IDE

* Go to: <https://repl.it/>
* Select Python3
* Sign-up / Create an account
* Make sure you can remember your account information for the rest of the course.

Using the IDE

* Use the black area like a calculator to try simple statements or commands
* Use the white area to create programs with multiple statements

**Accessing the Tutorial**

Accessing the Tutorial

* Go to: <http://www.letslearnpython.com/learn/>
* Read up to “Lesson 3: Math”

**Level 1: Basic Math & Strings**

Access the Tutorial and start at “Lesson 3: Math”.

Questions

1. Complete “Lesson 3: Math – Math Basics” by typing the sample commands in the black area of the IDE.
   1. Create your own expression using 5 “+” and “-“ operators.
   2. List your expression and the result below.

*9 + 5 + 7 – 12 – 1*

*= 8*

1. Complete “Lesson 3: Math – More Operators” by typing the sample commands in the black area of the IDE.
   1. Create your own expression using 5 “\*” and “/” operators.
   2. List your expression and the result below.

*10 / 5 \* 3 \* 3 \* 7*

*= 126.0*

1. Complete “Lesson 3: Math – More Division” by typing the sample commands in the black area of the IDE.
   1. Create one division expression that gives a whole number answer
   2. And one division expression that gives a decimal number answer.
   3. List your expressions and the results below.

*15 / 3*

*= 5.0*

*25 / 4*

*= 6.25*

1. Complete “Lesson 3: Math – Floats” by typing the sample commands in the black area of the IDE.
   1. Use the “round()” function for the expressions you created in question #3 above.
   2. List your “round()” expressions and the results they return below.

*Round (15 / 3)*

*= 5*

*Round (25 / 4)*

*= 6*

1. Read through “Lesson 3: Math – Comparison Operators”.
   1. Why do you think Equals is “==” instead of “=”?

I think that Equals is “==” instead of “=” because it means equal to, not equals. For example, 50 out of 100 is equal to ½. So, there are two equals because both of them equal the same thing.

* 1. What does “=” mean?

“=” means equals, the solution to the problem.

1. Complete “Lesson 3: Math – Practice” and “Lesson 3: Math – Practice Answers” by typing the sample commands in the black area of the IDE.
   1. Create an expression using 5 different operators that returns a “True” result
   2. And an expression using 5 different operators that returns a “False” result.
   3. List your expressions and the results returned below.

*7 < 6 + 5*

*True*

*9 < 8 + 6*

*True*

*5 \* 5 == 25*

*True*

*6 \* 6 == 36*

*True*

*6 / 3 == 2*

*True*

*9 < 7 + 2*

*False*

*7 \* 7 == 2*

*False*

*9 + 9 == 75*

*False*

*23 ! = 23*

*False*

*7908 \* 2 == 55*

*False*

1. Complete “Lesson 4: Strings – Strings” and “Lesson 4: Strings – Examples” by typing the sample commands in the black area of the IDE.
   1. Explain why typing “apple” works and why typing apple without quotes gives an error.

Typing apple in quotes works because if you want Python to read a string, it has to be inside quotes. It doesn’t understand what you’re trying to do if the command isn’t in quotes.

* 1. Also explain why “2 + 5” does not equal 7.

“2 + 5” doesn’t equal 7 because Python is reading the string back to you, not answering the question.

1. Complete “Lesson 4: Strings – Operators” by typing the sample commands in the black area of the IDE.
   1. Explain why typing “appl” + “e” works and why typing “apple” - “e” gives an error.

Typing “appl” + “e” works because Python can concatenate (add and multiply), but not take away.

* 1. Also explain why “Hello” \* 10 works but why “Hello” / 10 does work.

“Hello” \* 10 works because Python knows how to concatenate, but not subtract.

1. Complete “Lesson 4: Strings – Indexes” by typing the sample commands in the black area of the IDE.
   1. List the letters in your first name and the index for each letter in your first name.

H a s r a t

0 1 2 3 4 5

1. Complete “Lesson 4: Strings – Indexes Examples” by typing the sample commands in the black area of the IDE.
   1. Explain why print(“Hello!”[4]) does not print “l”.

Print(“Hello!”[4]) doesn’t return “I” because there is two of them. They share the same index number so it goes to the next letter, “o”, which is what it returned.

* 1. What does print(“Hay, Bob!”[4]) print? For a hint try print(“Hay, Bob!”[3]) and print(“Hay, Bob!”[5])

Print(“Hay, Bob!”[4]) prints a space, or a blank spot.

1. Complete “Lesson 4: Strings – Rules” by typing the sample commands in the black area of the IDE.
   1. Explain why print(“Hello!”[7]) gives an error.

Print(“Hello!”[7]) gives an error because the index it’s looking for is out of range (not there).

**Level 2: Booleans & Variables**

Access the Tutorial and start at “Lesson 5: Variables”

Questions

1. Complete “Lesson 5: Variables – Save a Value” by typing the sample commands in the black area of the IDE.
   1. What do you get if you type puppies / 3?

If you type puppies /3, it returns 12.0. This is because Python is doing 36 / 3, which is 12.

* 1. Why doesn’t typing kittens / 3 work?  
     Typing kittens / 3 doesn’t work because there is nothing assigned to that variable.

1. Complete “Lesson 5: Variables – Assign a New Value” by typing the sample commands in the black area of the IDE.
   1. Explain how the following sequence of commands works:
      * puppies = 36
      * puppies = puppies / 6
      * puppies

Puppies is assigned as being 36. Puppies is then assigned as being puppies divided by 6, so 6. Typing puppies then yields 6, because puppies (36) divided by 6 is 36.

1. Read through “Lesson 5: Variables – Rules”.
2. Complete “Lesson 5: Variables – Math Operators” by typing the sample commands in the black area of the IDE.
   1. Explain what happens for following sequence of commands:
      * colour = “red”
      * puppies = 36
      * colour + puppies

Colour is assigned as being red, and puppies is assigned as being 36. Colour + puppies is a number plus a word, so it yields an error message.

1. Complete “Lesson 5: Variables – String Operators” by typing the sample commands in the black area of the IDE.
   1. Explain why the following commands give different results:
      * Color + day \* fishes
      * ( Color + day ) \* fishes  
        Those commands gave different results due to the order of operations Python was doing. It did the multiplication first; to change what it does first, we put what we want it to do in brackets.
2. Complete “Lesson 5: Variables – Indexes” by typing the sample commands in the black area of the IDE.
   1. What is the index of ‘r’ in “watermelon”?

The index of “r” is 5.

* 1. Write an expression using mynumber to return ‘r’.

Fruit[mynumber – 9]

= “r”

1. Complete “Lesson 5: Variables – Assignments or Comparisons” by typing the sample commands in the black area of the IDE.
   1. What is the difference between “=” and “==”?

The difference between “=” and “==” is that “=” means “this equals that”, but “==” means “is this thing equal to this thing?”

* 1. Create your own mnemonic to remember this difference.

“=” means “equals”, “=” means “this equals to this equals”

1. Complete “Lesson 6: Errors – Examples” by typing the sample commands in the black area of the IDE.
   1. What doesn’t “friend” + 5 work?

“Friend” + 5 doesn’t work because Python is trying to add two different pieces of data, but it doesn’t know what to do because one is a string and one is an integer.

* 1. What is the difference between int and str?

Int = integer

Str = string

1. Read through “Lesson 6: Errors – Parts of an Error Message”.
   1. Is “friend” + 5 an example of:
      1. A Syntax Error?
      2. A Runtime Error?
      3. A Logic Error?

It’s a Type Error.

1. Read through “Lesson 6: Errors – Fixing Errors”.
   1. Use the ‘print’ command to print your first name and last name.

Print(“Hasrat”)

= Hasrat

Print(“Sidhu”)

= Sidhu

Print(“Hasrat”, “Sidhu”)

= Hasrat Sidhu

1. Complete “Lesson 7: Booleans – Types of Data” by typing the sample commands in the black area of the IDE.
   1. What is the value of: type(“True”)

String

* 1. What is the value of: type( True )

Boolean

* 1. Why is the result different?

The result is different due to the lack of or addition of quotations. The first one is a string because it has quotations around it, but the second doesn’t.

1. Complete “Lesson 7: Booleans – What Is A Boolean” by typing the sample commands in the black area of the IDE.
   1. Why do you think that having a Boolean data type is important in computer programming?

A Boolean data type is important in computer programming because it helps to make decisions and comparisons.

1. Complete “Lesson 7: Booleans – Trying Out Booleans” by typing the sample commands in the black area of the IDE.
   1. Why do you think that there is no “Maybe” Boolean data value in computer programming?

There is no “maybe” Boolean data value because there’s only a right or wrong in the programming. It can’t detect if something can be a “maybe”, only a true or false.

**Level 3: Lists & Logic**

Access the Tutorial and start at “Lesson 7: Booleans”

Questions

1. Complete “Lesson 7: Booleans – AND Comparisons” by typing the sample commands in the black area of the IDE.
   1. Try the following Python statements and record the results.
      1. True and True
      2. True and False
      3. False and True
      4. False and False

*True and True*

*=> True*

*True and False*

*=> False*

*False and True*

*=> False*

*False and False*

*=> False*

* 1. Explain if there are any other combinations of True / False.

There aren’t any more combinations. The combinations already used show that one false is all that’s needed for it to be false.

* 1. Explain how the AND operator is similar to a math operator and how it is different.

The AND operator is similar to a math operator and also very different. It’s similar because it’s comparing the two, but it’s different because it’s comparing words/statements and not numbers.

1. Complete “Lesson 7: Booleans – OR Comparisons” by typing the sample commands in the black area of the IDE.
   1. Try the following Python statements and record the results.
      1. True or True
      2. True or False
      3. False or True
      4. False or False

*True or True*

*=> True*

*True or False*

*=> True*

*False or True*

*=> True*

*False or False*

=> False

* 1. Explain how the OR operator is similar to the AND operator and how it is different.

The OR operator is similar to the AND operator and different in a way. They’re similar because they’re comparing two things to make it true or false; however, the AND operator only needs one false for it to be false, while the OR operator only needs one true for it to be true.

1. Complete “Lesson 7: Booleans – NOT Comparisons” by typing the sample commands in the black area of the IDE.
   1. Try the following Python statements and record the results.
      1. not (True or True)
      2. not (True or False)
      3. not (False or True)
      4. not (False or False)

*not(True or True)*

*=> False*

*not(True or False)*

*=> False*

*not(False or True)*

*=> False*

*not(False or False)*

*=> True*

* 1. Explain how the combination of the NOT & OR operators is similar to the AND operator by itself and how it is different.

The NOT & OR operators are similar to the AND operator because they’re both comparing two things between true or false; however, the NOT operator reverses the answer entirely, while the OR operator makes it so one true makes the whole thing true.

1. Complete “Lesson 7: Booleans – Expressions” by typing the sample commands in the black area of the IDE.
   1. Explain why the following two Python statements give different results.
      1. not (True or True)
      2. not True or True

The two Python statements give different results because one has brackets (which makes whatever’s in the brackets be calculated first), while the other doesn’t.

* 1. Explain why the following two Python statements give the same results.
     1. not (True and True)
     2. not True and True

The two Python statements give the same results because

1. Complete “Lesson 7: Booleans – Practice” by typing the sample commands in the black area of the IDE.
   1. Create three more practice expressions similar to those in the tutorial.
   2. Provide the results for your practice expressions

*3 == 3 and True*

*=> True*

*"dog" == "more than one dog"*

*=> False*

*7 !=0 and True*

*=> True*

1. Complete “Lesson 8: Lists – A Collection of Objects” by typing the sample commands in the black area of the IDE.
   1. Create a list of your favorite sports teams.
   2. Assign your list to a variable.
   3. Confirm that your variable and your list are the same.

*["pizza", "burger", "watermelon", "pasta"]*

*=> ['pizza', 'burger', 'watermelon', 'pasta']*

*Yum = ["pizza", "burger", "watermelon", "pasta"]*

*Yum*

*=> ['pizza', 'burger', 'watermelon', 'pasta']*

1. Complete “Lesson 8: Lists – List Indexes” by typing the sample commands in the black area of the IDE.
   1. What is the list index of the last team in your list of favorite sports teams.
   2. In the tutorial, the error produced by typing “fruit[3]” is an example of:
      1. A Syntax Error?
      2. A Runtime Error?
      3. A Logic Error?

It’s an example of an Index Error.

1. Complete “Lesson 8: Lists – Practice” and “Lesson 8: Lists – Practice Answers” by typing the sample commands in the black area of the IDE.

Color = ["pink", "purple", "blue"]

Color[1]

=> 'purple'

NOTE: Starting with Lesson 9 you should use the WHITE area of the IDE for entering example code with multiple statements.

1. Complete “Lesson 9: Logic – Making Decisions” by typing the sample commands in the white area of the IDE.
   1. Modify the tutorial code to print “Hi Alfred!” based on a decision using numbers

*num = 5*

*if num > 2:*

*print('Hey, Alfred')*

*else:*

*print('You are not Alfred')*

*Hey, Alfred*

1. Complete “Lesson 9: Logic – Adding A Choice” by typing the sample commands in the white area of the IDE.
   1. Modify the tutorial code to print your first name or your last name based on a choice (using “else”).

*myname = "Hasrat"*

*if myname == "Hasrat":*

*print("Hi Hasrat!")*

*else:*

*print("Liar!")*

*Hi Hasrat!*

1. Complete “Lesson 9: Logic – Adding Many Choices” and “Lesson 9: Logic – Practice” by typing the sample commands in the white area of the IDE.
   1. Modify the tutorial code and “elif” statements to make a choice using at least 4 of your friends names.

*myname = "Hasrat"*

*if myname == "Hasrat":*

*print("Hi Hasrat!")*

*elif myname == "Gobina":*

*print("Hi Gobina!")*

*elif myname == "Sharlene":*

*print("Hi Sharlene!")*

*elif myname == "Muktika":*

*print("Hi Muktika!")*