**Outline**

Access the Python Development environment and continue the tutorial to gain an additional exposure to the Python programming language. Begin to develop an familiarity with intermediate 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 Tutorial**

Accessing the Tutorial

* Go to: <http://www.letslearnpython.com/learn/>
* Read up to “Lesson 12: Input”

**Level 1: Input & Output**

1. Read through “Lesson 12: Input – What Is Input?” and “Lesson 12: Input – Example” and “Lesson 12: Input – Shortcut”.
2. Type the following code into the white area of the IDE and run the program. Explain what you see in the black area of the IDE.

print("Type your name:")

name = input()

print("Hi", name, "how are you?")

Black area asks for input, when input is entered, message is printed filling in part of it with the input

1. Create a short program that reads numerical input from the console and does the following:
   1. Uses the input() function to read a numerical value from the console.
   2. Calculates the square root of the number
   3. Prints the result to the console output
   4. Provides appropriate prompt and message strings to go with the input and output.
   5. Provide your complete program below.

import math

def squaring():

print("Type a # to find its square root:")

number = input()

number= int(number)

print(math.sqrt(number))

**Level 2: Tic-Tac-Toe Game**

<https://www.geeksforgeeks.org/python-implementation-automatic-tic-tac-toe-game-using-random-number/>

<https://github.com/emilyaviva/simple-python-tic-tac-toe/blob/master/tictactoe.py>

1. Write a Python program to play a game of Tic-Tac-Toe. (You may modify a program that you found on-line to meet the expectations of this module.)
   1. The program may be either player v. computer or player 1 v. player 2.
   2. The program does not need to determine a winner
   3. The program just needs to keep track of moves and spaces in the game board
2. Provide a complete listing of your program.
   1. Your listing **MUST** include line numbers .

import turtle

myPen= turtle.Turtle()

print("Lets play tic tac toe!")

print ("You win once a row, or column is filled with your sign!")

print ("In order to play, enter the row and the column you wish to put your sign on!")

game = [[0, 0, 0],

[0, 0, 0],

[0, 0, 0],]

def game\_board():

print(" 0 1 2")

count = 0

for row in game:

print(count, row)

count +=1

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

row = int(input("player 2 enter a row:"))

col= int(input("player 2 enter a column:"))

game[row][col]="O"

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

row = int(input("player 2 enter a row:"))

col= int(input("player 2 enter a column:"))

game[row][col]="O"

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

row = int(input("player 2 enter a row:"))

col= int(input("player 2 enter a column:"))

game[row][col]="O"

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

row = int(input("player 2 enter a row:"))

col= int(input("player 2 enter a column:"))

game[row][col]="O"

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

print ("If you got a column or row full with your sign, You win!")

1. Explain how your program keeps track of the game board.   
   (Provide specific code references by line number.)
   1. What python types and data structures are used?

Lists, functions, loops, input, print, int, strings

* 1. How are moves by player X and player O recorded?

The program asks each player (player 1 or player 2) to enter a move. Once a move is entered by input, the game board is reprinted with the new move added on to it:

game = [[0, 0, 0],

[0, 0, 0],

[0, 0, 0],]

def game\_board():

print(" 0 1 2")

count = 0

for row in game:

print(count, row)

count +=1

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

* 1. How are free spaces recorded?

Free spaces are recorded as the number zero, as the game board consists of three small lists, inside one large list, which all consists of zeros until a player inputs a “X” or an “O”.

1. Explain how moves and commands are input from the console.  
   (Provide specific code references by line number.)
   1. How does the player tell the program about the move location (row, column)?

Player inputs the number of the row and column in which the player wants to place their move:

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

* 1. How does the program verify that the move location is valid?

The program verifies the move by reprinting the game board with the move on it. If the move entered isn't valid, this will not happen

* 1. How does the program verify that the space is free?

It does not. But, after each move, the program re-prints the game board. Players can visually see which spots are available

* 1. What does the program do if there is something wrong with the move?

The program sends back an error, and needs to be run again (this needs to be altered).

1. Explain how the program keeps track of gameplay.  
   (Provide specific code references by line number.)
   1. How does the program switch between player X and player O moves?

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

row = int(input("player 2 enter a row:"))

col= int(input("player 2 enter a column:"))

game[row][col]="O"

game\_board()

As listed above, the code asks for different players (1 or 2) to enter their move. When player 1 enters input, an X is printed. When player 2 enters input, an O is printed.

* 1. How does the program keep asking for moves?

game\_board()

row = int(input("player 1 enter a row:"))

col= int(input("player 1 enter a column:"))

game[row][col]="X"

game\_board()

row = int(input("player 2 enter a row:"))

col= int(input("player 2 enter a column:"))

game[row][col]="O"

game\_board()

This code above is repeated enough for nine moves to be placed. After that, the program does not ask for moves

* 1. How does the program decide when to stop asking for moves?

Currently, the program does not stop asking for moves until 9 moves (the max amount of moves that can be made) are there. However, in level 3, when code for determining a winner is found, then the program will be able to stop asking for moves.