**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?")

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 Root():

print(What number do you want to Square Root;)

num = int(input())

print(;The Sqaure Root of quot;,num,&quot;is;,math.sqrt(num))

Root()

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

1. Write a Python program to play a game of Toc-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 random

def drawBoard(board):

print(' | |')

print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3])

print(' | |')

print('-----------')

print(' | |')

print(' ' + board[4] + ' | ' + board[5] + ' | ' + board[6])

print(' | |')

print('-----------')

print(' | |')

print(' ' + board[7] + ' | ' + board[8] + ' | ' + board[9])

print(' | |')

def inputPlayerLetter():

letter = ''

while not (letter == 'X' or letter == 'O'):

print('Do you want to be X or O?')

letter = input().upper()

if letter == 'X':

return ['X', 'O']

else:

return ['O', 'X']

def whoGoesFirst():

if random.randint(0, 1) == 0:

return 'computer'

else:

return 'player'

def playAgain():

print('Do you want to play again? (yes or no)')

return input().lower().startswith('y')

def makeMove(board, letter, move):

board[move] = letter

def isWinner(bo, le):

return ((bo[7] == le and bo[8] == le and bo[9] == le) or # across the top

(bo[4] == le and bo[5] == le and bo[6] == le) or # across the middle

(bo[7] == le and bo[8] == le and bo[9] == le) or # across the bottom

(bo[1] == le and bo[4] == le and bo[7] == le) or # down the left side

(bo[2] == le and bo[5] == le and bo[8] == le) or # down the middle

(bo[3] == le and bo[6] == le and bo[9] == le) or # down the right side

(bo[1] == le and bo[5] == le and bo[9] == le) or # diagonal

(bo[7] == le and bo[5] == le and bo[3] == le)) # diagonal

def getBoardCopy(board):

dupeBoard = []

for i in board:

dupeBoard.append(i)

return dupeBoard

def isSpaceFree(board, move):

return board[move] == ' '

def getPlayerMove(board):

move = ' '

while move not in '1 2 3 4 5 6 7 8 9'.split() or not isSpaceFree(board, int(move)):

print('What is your next move? (1-9)')

move = input()

return int(move)

def chooseRandomMoveFromList(board, movesList):

possibleMoves = []

for i in movesList:

if isSpaceFree(board, i):

possibleMoves.append(i)

if len(possibleMoves) != 0:

return random.choice(possibleMoves)

else:

return None

def getComputerMove(board, computerLetter):

if computerLetter == 'X':

playerLetter = 'O'

else:

playerLetter = 'X'

for i in range(1, 10):

copy = getBoardCopy(board)

if isSpaceFree(copy, i):

makeMove(copy, computerLetter, i)

if isWinner(copy, computerLetter):

return i

for i in range(1, 10):

copy = getBoardCopy(board)

if isSpaceFree(copy, i):

makeMove(copy, playerLetter, i)

if isWinner(copy, playerLetter):

return i

move = chooseRandomMoveFromList(board, [1, 3, 7, 9])

if move != None:

return move

if isSpaceFree(board, 5):

return 5

return chooseRandomMoveFromList(board, [2, 4, 6, 8])

def isBoardFull(board):

for i in range(1, 10):

if isSpaceFree(board, i):

return False

return True

print('Welcome to Tic Tac Toe Mr. Nestor!')

while True:

theBoard = [' '] \* 10

playerLetter, computerLetter = inputPlayerLetter()

turn = whoGoesFirst()

print('The ' + turn + ' will go first.')

gameIsPlaying = True

while gameIsPlaying:

if turn == 'player':

drawBoard(theBoard)

move = getPlayerMove(theBoard)

makeMove(theBoard, playerLetter, move)

if isWinner(theBoard, playerLetter):

drawBoard(theBoard)

print('Hooray! You have won the game!')

gameIsPlaying = False

else:

if isBoardFull(theBoard):

drawBoard(theBoard)

print('The game is a tie!')

break

else:

turn = 'computer'

else:

move = getComputerMove(theBoard, computerLetter)

makeMove(theBoard, computerLetter, move)

if isWinner(theBoard, computerLetter):

drawBoard(theBoard)

print('The computer has beaten you! Nice Try.')

gameIsPlaying = False

else:

if isBoardFull(theBoard):

drawBoard(theBoard)

print('The game is a tie!')

break

else:

turn = 'player'

if not playAgain():

break

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?
   2. How are moves by player X and player O recorded?
   3. How are free spaces recorded?

To keep track of my game board, I made a list titled Board, the list had indexes starting

from 1-9. I used the indexes 1-9 for my game board. To do this, I made a function called drawBoard

in this function, I placed the indexes 1-3 in the top row, 4-6 in the middle row and 7-8

9 in the bottom row. To add the grid shaped lines, I included vertical lines between the indexes

contained the rows and I included horizontal lines between the rows to complete the grid. The lines

are shown as strings. To do all this I used the print() command in the showBoard function. The free spaces remain in 1-9 order with the index number being shown, so it is easy to play moves. The game board is updated and shown after every computer and Player move.

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)?
   2. How does the program verify that the move location is valid?
   3. How does the program verify that the space is free?
   4. What does the program do if there is something wrong with the move?

The player is given choice of picking a number between 1 and 9, all these are spots on the Tic-

Tac-Toe grid. When the player chooses the number, the number is then converted

into an integer using the help of int(input()), so it can be identified as an index of

Board. The number then goes through a check which checks if the number/index

that is selected is a valid move. The move inputted by the player can only be valid if it is between 1 and 9. If the number typed is bigger than 9 and smaller than 1, it won’t work.

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?
   2. How does the program keep asking for moves?
   3. How does the program decide when to stop asking for moves?

The program for moves is in a while loop which always true unless it is stopped with “break;”. This means that the moves switch between player x and o and thee loop keeps on

going. For example, player o goes then player x goes until a winner is decided or when they’re no moves left to make.