**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?")

It asks you to type your name in the black area, and then it says “Hi (your 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.

print("Type a number you want to square root between 1-10")

number=input()

if number==1:

print("1")

elif number==2:

print("1.41421356237")

elif number==3:

print("1.73205080757")

elif number==4:

print ("4")

elif number==5:

print("2.2360679775")

elif number==6:

print("2.44948974278")

elif number==7:

1. print("2.64575131106")
2. elif number==8:
3. print("2.82842712475")
4. elif number==9:
5. print("3")
6. elif number==10:
7. print("3.16227766017")

**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 .
3. pX="X"
4. pO="O"
5. pN=""
6. board=[pN,pN,pN,
   1. pN,pN,pN,
   2. pN,pN,pN]
7. def printBoard():
8. print()
9. print(board[0],"|",board[1],"|", board[2])
10. print("---------")
11. print(board[3],"|",board[4],"|", board[5])
12. print("---------")
13. print(board[6],"|",board[7],"|", board[8])
14. print()
15. while True:
16. move=int(input("Player X, Make a move:"))
17. board[move]=pX
18. printBoard()
19. if board[0]==pX and board[1]==pX and board[2]==pX:
20. print("X Wins")
21. break
22. if board[3]==pX and board[4]==pX and board[5]==pX:
23. print("X Wins")
24. break
25. if board[6]==pX and board[7]==pX and board[8]==pX:
26. print("X Wins")
27. break
28. if board[0]==pX and board[3]==pX and board[6]==pX:
29. print("X Wins")
30. break
31. if board[1]==pX and board[4]==pX and board[7]==pX:
32. print("X Wins")
33. break
34. if board[2]==pX and board[5]==pX and board[8]==pX:
35. print("X Wins")
36. break
37. if board[0]==pX and board[4]==pX and board[8]==pX:
38. print("X Wins")
39. break
40. if board[6]==pX and board[1]==pX and board[2]==pX:
41. print("X Wins")
42. break
43. if board[0]==pO and board[1]==pO and board[2]==pO:
44. print("O Wins")
45. break
46. if board[3]==pO and board[4]==pO and board[5]==pO:
47. print("O Wins")
48. break
49. if board[6]==pO and board[7]==pO and board[8]==pO:
50. print("O Wins")
51. break
52. if board[0]==pO and board[3]==pO and board[6]==pO:
53. print("O Wins")
54. break
55. if board[1]==pO and board[4]==pO and board[7]==pO:
56. print("O Wins")
57. break
58. if board[2]==pO and board[5]==pO and board[8]==pO:
59. print("O Wins")
60. break
61. if board[0]==pO and board[4]==pO and board[8]==pO:
62. print("O Wins")
63. break
64. if board[6]==pO and board[1]==pO and board[2]==pO:
65. print("O Wins")
66. break
67. move=int(input("Player O, Make a move:"))
68. board[move]=pO
69. printBoard()
70. if board[0]==pX and board[1]==pX and board[2]==pX:
71. print("X Wins")
72. break
73. if board[3]==pX and board[4]==pX and board[5]==pX:
74. print("X Wins")
75. break
76. if board[6]==pX and board[7]==pX and board[8]==pX:
77. print("X Wins")
78. break
79. if board[0]==pX and board[3]==pX and board[6]==pX:
80. print("X Wins")
81. break
82. if board[1]==pX and board[4]==pX and board[7]==pX:
83. print("X Wins")
84. break
85. if board[2]==pX and board[5]==pX and board[8]==pX:
86. print("X Wins")
87. break
88. if board[0]==pX and board[4]==pX and board[8]==pX:
89. print("X Wins")
90. break
91. if board[6]==pX and board[1]==pX and board[2]==pX:
92. print("X Wins")
93. break
94. if board[0]==pO and board[1]==pO and board[2]==pO:
95. print("O Wins")
96. break
97. if board[3]==pO and board[4]==pO and board[5]==pO:
98. print("O Wins")
99. break
100. if board[6]==pO and board[7]==pO and board[8]==pO:
101. print("O Wins")
102. break
103. if board[0]==pO and board[3]==pO and board[6]==pO:
104. print("O Wins")
105. break
106. if board[1]==pO and board[4]==pO and board[7]==pO:
107. print("O Wins")
108. break
109. if board[2]==pO and board[5]==pO and board[8]==pO:
110. print("O Wins")
111. break
112. if board[0]==pO and board[4]==pO and board[8]==pO:
113. print("O Wins")
114. break
115. if board[6]==pO and board[1]==pO and board[2]==pO:
116. print("O Wins")
117. break
118. 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?

The data structure that is used is print. Lines 1-3 are an example.

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

At the beginning, the players are defined. pX is player X, pO is player O. Whenever the person makes a move, move is the input and board[move]=pX is when it is recorded in the board. Lines 19-23 are an example.

* 1. How are free spaces recorded?

Free spaces already exist at the beginning. They were created by blanks. Lines 3-7 are an example.

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)?

The player tells the program a number from 0-9. Then, the program does the following program if the input is between 0-8. Lines 19-23 are an example for player X.

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

If the person prints anything else than a number between 0 and 8, it will give the message “Choose another place between 0-8”. Lines 20 and 21 are an example for player X.

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

It doesn’t. The players will know which place is free and which isn’t.

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

It will do nothing.

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?

When player X has gone, it checks for any win. Then it goes onto player O. Lines 19 to 87 is an example of how the program takes in the input and checks for any win.

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

It is set on repeat. Line 18 is an example.

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

It stops after someone has won. Lines 25 to 87 are an example.

**Level 3: Basic Enhancements**

1. Explain, in plain words, a strategy for determining if player “x” or player “O” has won the game after a move is made.

To determine if player “x” or player “o” has won, you could check if player X has gone in three spots in a row. This would require the program to check the board spots.

1. Provide a function called “checkWinForX” that returns the Boolean value of “True” if player “x” won the game.

The following program does a check for both X and O.

def checkWinForX():

if board[0]==pX and board[1]==pX and board[2]==pX:

print("X Wins")

elif board[3]==pX and board[4]==pX and board[5]==pX:

print("X Wins")

elif board[6]==pX and board[7]==pX and board[8]==pX:

print("X Wins")

elif board[0]==pX and board[3]==pX and board[6]==pX:

print("X Wins")

elif board[1]==pX and board[4]==pX and board[7]==pX:

print("X Wins")

elif board[2]==pX and board[5]==pX and board[8]==pX:

print("X Wins")

elif board[0]==pX and board[4]==pX and board[8]==pX:

print("X Wins")

elif board[6]==pX and board[1]==pX and board[2]==pX:

print("X Wins")

def checkWinForO():

if board[0]==pO and board[1]==pO and board[2]==pO:

print("O Wins")

elif board[3]==pO and board[4]==pO and board[5]==pO:

print("O Wins")

elif board[6]==pO and board[7]==pO and board[8]==pO:

print("O Wins")

elif board[0]==pO and board[3]==pO and board[6]==pO:

print("O Wins")

elif board[1]==pO and board[4]==pO and board[7]==pO:

print("O Wins")

elif board[2]==pO and board[5]==pO and board[8]==pO:

print("O Wins")

elif board[0]==pO and board[4]==pO and board[8]==pO:

print("O Wins")

elif board[6]==pO and board[1]==pO and board[2]==pO:

print("O Wins")

1. Modify your program to check and print a message, and stop the game of player “x” or player “O” wins the game.

def checkWinForX():

if board[0]==pX and board[1]==pX and board[2]==pX:

print("X Wins")

break

elif board[3]==pX and board[4]==pX and board[5]==pX:

print("X Wins")

break

elif board[6]==pX and board[7]==pX and board[8]==pX:

print("X Wins")

break

elif board[0]==pX and board[3]==pX and board[6]==pX:

print("X Wins")

break

elif board[1]==pX and board[4]==pX and board[7]==pX:

print("X Wins")

break

elif board[2]==pX and board[5]==pX and board[8]==pX:

print("X Wins")

break

elif board[0]==pX and board[4]==pX and board[8]==pX:

print("X Wins")

break

elif board[6]==pX and board[1]==pX and board[2]==pX:

print("X Wins")

break

def checkWinForO():

if board[0]==pO and board[1]==pO and board[2]==pO:

print("O Wins")

break

elif board[3]==pO and board[4]==pO and board[5]==pO:

print("O Wins")

break

elif board[6]==pO and board[7]==pO and board[8]==pO:

print("O Wins")

break

elif board[0]==pO and board[3]==pO and board[6]==pO:

print("O Wins")

break

elif board[1]==pO and board[4]==pO and board[7]==pO:

print("O Wins")

break

elif board[2]==pO and board[5]==pO and board[8]==pO:

print("O Wins")

break

elif board[0]==pO and board[4]==pO and board[8]==pO:

print("O Wins")

break

elif board[6]==pO and board[1]==pO and board[2]==pO:

print("O Wins")

break

1. Demonstrate your enhanced game to Mr. Nestor for credit for this level.

**Level 4: AI Enhancements**

1. Explain, in plain words, a strategy for suggesting the best move for player “x” or player “O” to make when it is their turn to move.

For the human playing, I would suggest anywhere where it would block the AI. For AI, I would set up different spots to go. For example, I would make the AI bock the player first, and then make the AI make a smart move.

1. Create a function to implement your strategy for suggesting the best move.

If board[0] and board[1]==pX:

Board[2]==pO

The program above would be repeated anywhere someone would be trying to win.

1. Modify your program to print a suggested move when it is each player’s turn to move.

If board[0] and board[1]==pX:

Print(“Recommendation: Go for place 3”)

The above program would be repeated anywhere someone would be trying to win.

1. Demonstrate your AI enhanced game to Mr. Nestor for credit for this level.