



UNIVERSITY OF MISSOURI-KANSAS CITY

Python and Deep Learning

Lab 1 Assignment

Team Members:

Pragathi Thammaneni

Sridevi Mallipudi

Introduction:

This Lab assignment deals with the Python Programming which is an -interpreted programming language. In this lab assignment it primarily focuses on the basic topics of Python programming.

Objectives:

To code for the 6 questions by implementing the below concepts.

Python inbuilt functions

Operators

Iterative Loops

Conditional/Decision making loops

Usage of List and Set Functions

Usage of Regular Expressions

Approaches /Methods:

Using Python 3.6, PyCharm (Community edition)

Workflow &Datasets/Parameters and Evaluation:

The below each question will follow different approaches to solve. Coding is done to perform the evaluation of each individual snippet to execute the datasets which are provided as the input parameters.

Question 1:

1. For any web application login, the user password need to be validated against database rules. For My UMKC web application following are the criteria for valid password:
 - a. The password length should be in range 6-16 characters
 - b. Should have at least one number
 - c. Should have at least one special character in [\$@!*
 - d. Should have at least one lowercase and at least one uppercase character

Solution:

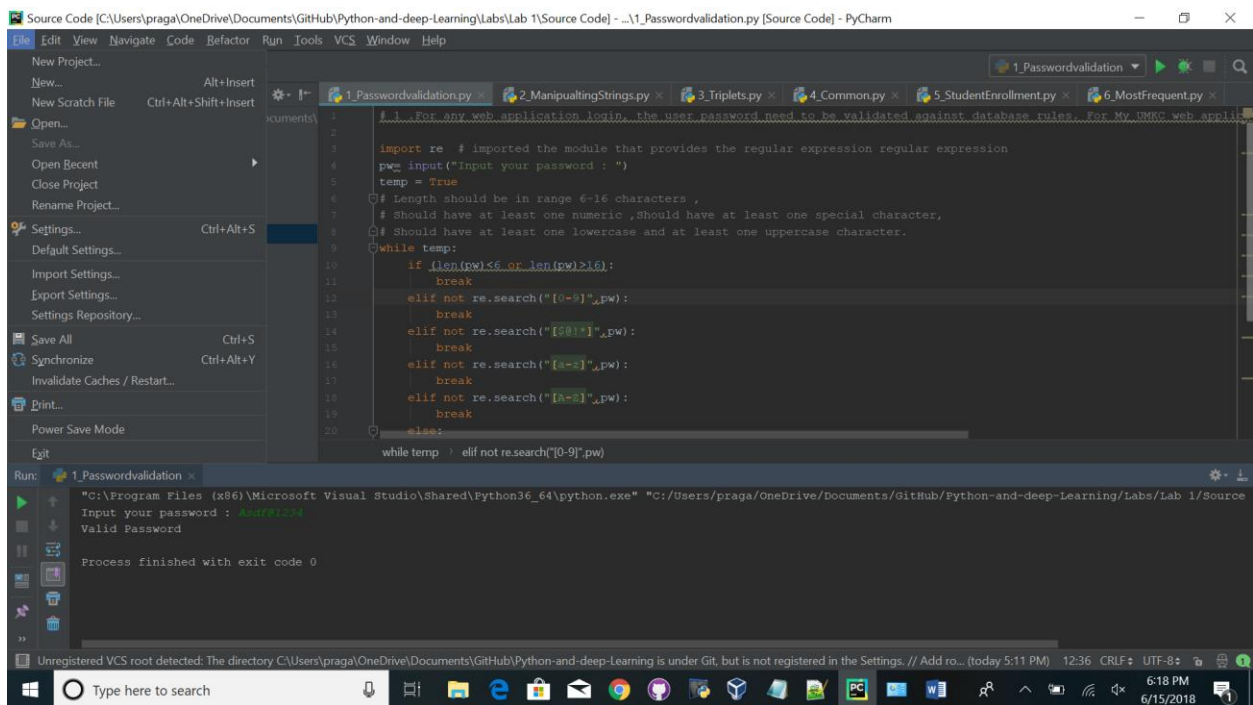
This snippet code provides the above implementation to validate a password for the above conditions. The steps that are followed for the workflow are, the input is taken from the user and looped conditions which matches the password criteria. If the password matches the above criteria it prints the valid password or else it will print the invalid password.

Code Snippet:

The below is the code for executing the above workflow

```
1 # 1 .For any web application login, the user password need to be validated against database rules. For My UMKC web application following are the rules
2
3 import re # imported the module that provides the regular expression regular expression
4 pw= input("Input your password : ")
5 temp = True
6 # Length should be in range 6-16 characters ,
7 # Should have at least one numeric ,Should have at least one special character,
8 # Should have at least one lowercase and at least one uppercase character.
9 while temp:
10     if (len(pw)<6 or len(pw)>16):
11         break
12     elif not re.search("[0-9]",pw):
13         break
14     elif not re.search("[!@#$%^&*]",pw):
15         break
16     elif not re.search("[a-z]",pw):
17         break
18     elif not re.search("[A-Z]",pw):
19         break
20     else:
21         print("Valid Password")
22         temp=False
23         break
24
25 if temp:
26     print("Not a Valid Password")
```

Output Screenshot:



Question 2:

2. Write a Python function that accepts a sentence of words from user and display the following:

- a. Middle word
- b. Longest word in the sentence
- c. Reverse all the words in sentence

Sample input:

My name is Jacqueline Fernandez Dsouza

Sample output:

Middle words are: [is,Jacqueline]

Longest word is: Jacqueline

Sentence with reverse words is: ym eman si enileuqcaj zednanref azuosd

Solution:

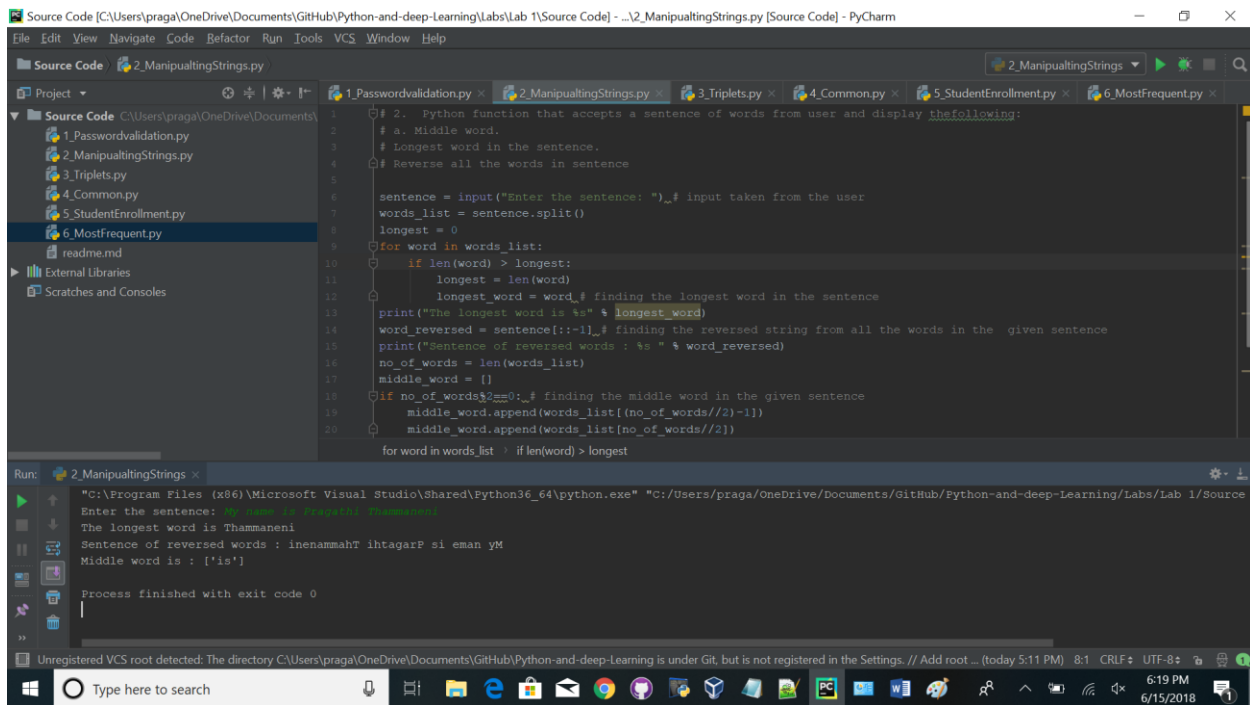
This snippet code provides the above implementation to find out the middle/longest /reverse of the words from the given sentence

Code Snippet:

The below is the code for executing the above workflow

```
23 lines (22 sloc) | 991 Bytes
Raw Blame History
1 # 2. Python function that accepts a sentence of words from user and display thefollowing:
2 # a. Middle word.
3 # Longest word in the sentence.
4 # Reverse all the words in sentence
5
6 sentence = input("Enter the sentence: ") # input taken from the user
7 words_list = sentence.split()
8 longest = 0
9 for word in words_list:
10     if len(word) > longest:
11         longest = len(word)
12         longest_word = word # finding the longest word in the sentence
13 print("The longest word is %s" % longest_word)
14 word_reversed = sentence[::-1] # finding the reversed string from all the words in the given sentence
15 print("Sentence of reversed words : %s " % word_reversed)
16 no_of_words = len(words_list)
17 middle_word = []
18 if no_of_words%2==0: # finding the middle word in the given sentence
19     middle_word.append(words_list[(no_of_words//2)-1])
20     middle_word.append(words_list[no_of_words//2])
21 else:
22     middle_word.append(words_list[no_of_words//2])
23 print("Middle word is : %s" % middle_word)
```

Output Screenshot:



```
1 # 2. Python function that accepts a sentence of words from user and display the following:
2 # a. Middle word.
3 # Longest word in the sentence.
4 # Reverse all the words in sentence
5
6 sentence = input("Enter the sentence: ") # input taken from the user
7 words_list = sentence.split()
8 longest = 0
9
10 for word in words_list:
11     if len(word) > longest:
12         longest = len(word)
13         longest_word = word # finding the longest word in the sentence
14
15 word_reversed = sentence[::-1] # finding the reversed string from all the words in the given sentence
16 print("The longest word is %s" % longest_word)
17 print("Sentence of reversed words : %s" % word_reversed)
18 no_of_words = len(words_list)
19 middle_word = []
20
21 if no_of_words % 2 == 0: # finding the middle word in the given sentence
22     middle_word.append(words_list[(no_of_words//2)-1])
23     middle_word.append(words_list[(no_of_words//2)])
24
25 for word in words_list:
26     if len(word) > longest
```

Run: 2_ManipulatingStrings x

"C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36_64\python.exe" "C:/Users/praga/OneDrive/Documents/GitHub/Python-and-deep-Learning/Labs/Lab 1/Source Code/2_ManipulatingStrings.py"

Enter the sentence: The longest word is Tammaneni

The longest word is Tammaneni

Sentence of reversed words : inenammaht ihtagarP si eman yM

Middle word is : ['is']

Process finished with exit code 0

Question 3:

3. Given a list of n number, write a Python program to find triplets in the list which gives the sum of zero.

Sample input: [1, 3, 6, 2, -1, 2, 8, -2, 9]

Sample output: [(3, -1, -2)]

Solution:

This snippet code provides the above implementation to find the triplets from the given list which forms the summation as zero. User is allowed to give the input then the below code is executed to get the set which forms the sum as zero when added up.

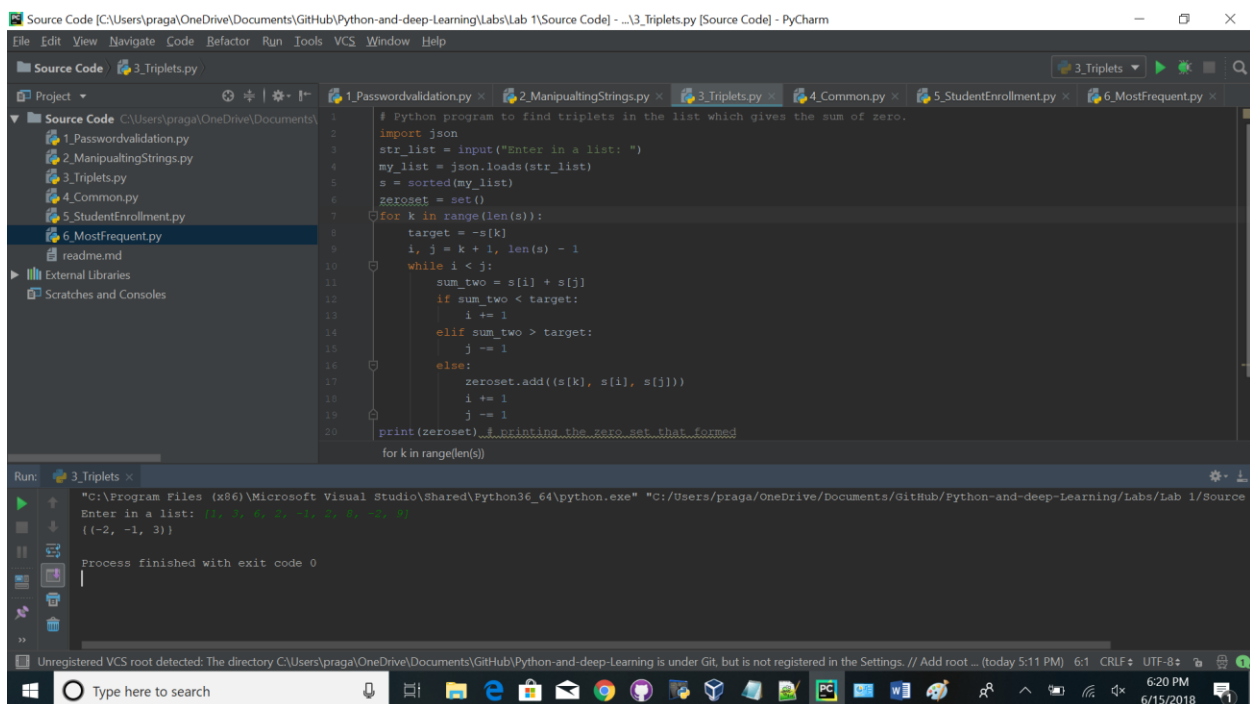
Code Snippet:

The below is the code for executing the above workflow

```
20 lines (20 sloc) | 574 Bytes
Raw Blame History

1 # Python program to find triplets in the list which gives the sum of zero.
2 import json
3 str_list = input("Enter in a list: ")
4 my_list = json.loads(str_list)
5 s = sorted(my_list)
6 zeroset = set()
7 for k in range(len(s)):
8     target = -s[k]
9     i, j = k + 1, len(s) - 1
10    while i < j:
11        sum_two = s[i] + s[j]
12        if sum_two < target:
13            i += 1
14        elif sum_two > target:
15            j -= 1
16        else:
17            zeroset.add((s[k], s[i], s[j]))
18            i += 1
19            j -= 1
20    print(zeroset) # printing the zero set that formed
```

Output Screenshot:



Question 4:

4. Consider the following scenario. You have a list of students who are attending class "Python" and another list of students who are attending class "Web Application". Find the list of students who are attending both the classes. Also find the list of students who are not common in both the classes. Print the both lists.

Solution:

This snippet code provides the above implementation to find out the common intersection and the unique items of two lists. To solve this created two lists which is one for who attends the python class and one for who attends the web application. The below code is executed to find out the common & not common students which is in both lists.

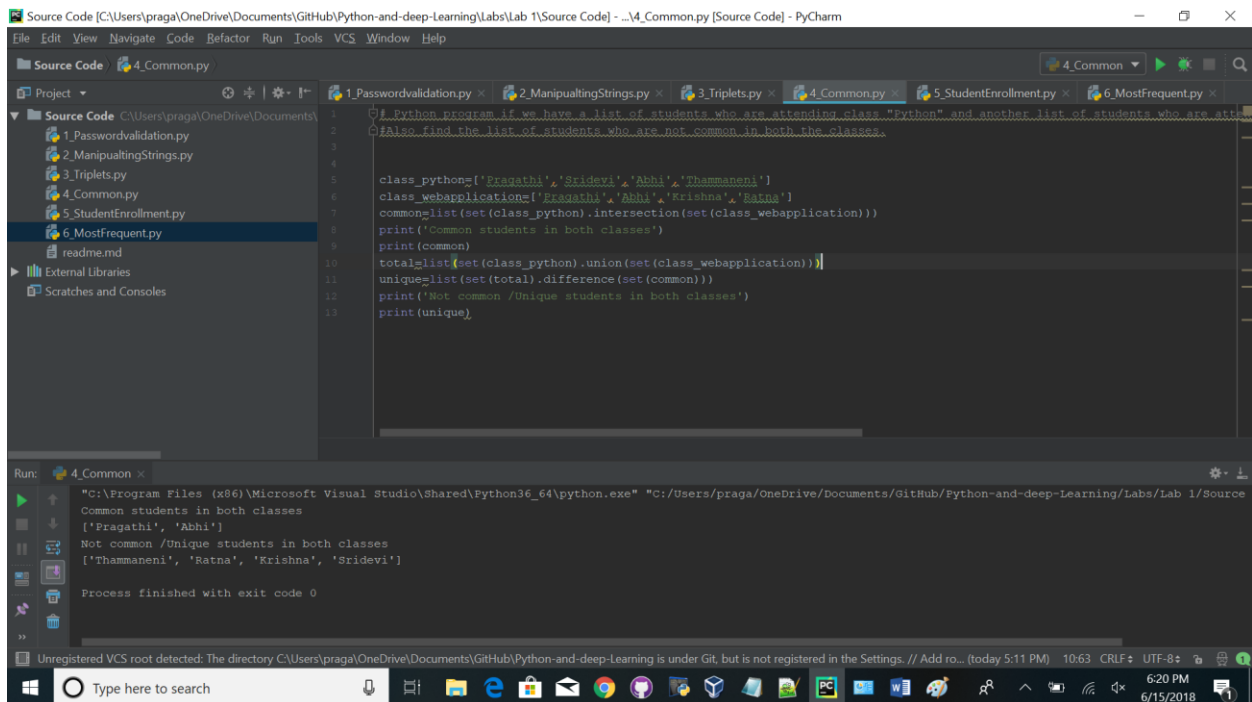
Code Snippet:

The below is the code for executing the above workflow

```
13 lines (11 sloc) | 717 Bytes
Raw Blame History

1 # Python program if we have a list of students who are attending class "Python" and another list of students who are attending class "Web A
2 #Also find the list of students who are not common in both the classes.
3
4
5 class_python=['Pragathi','Sridevi','Abhi','Thammaneni']
6 class_webapplication=['Pragathi','Abhi','Krishna','Ratna']
7 common=list(set(class_python).intersection(set(class_webapplication)))
8 print('Common students in both classes')
9 print(common)
10 total=list(set(class_python).union(set(class_webapplication)))
11 unique=list(set(total).difference(set(common)))
12 print('Not common /Unique students in both classes')
13 print(unique)
```

Output Screenshot:



Question 5:

5. Write a python program to create any one of the following management systems. You can also pick one of your own.

- a. Library Management System (e.g. classes Person, Student, Librarian, Book etc.)
- b. Airline Booking Reservation System (e.g. classes Flight, Person, Employee, Passenger etc.)
- c. Hotel Reservation System (e.g. classes Room, Occupants, Employee etc.)
- d. Student Enrollment System (e.g. classes Student, System, Grades etc.)
- e. Expense Tracker System (e.g. classes Expense, Transaction Category etc.)

Solution:

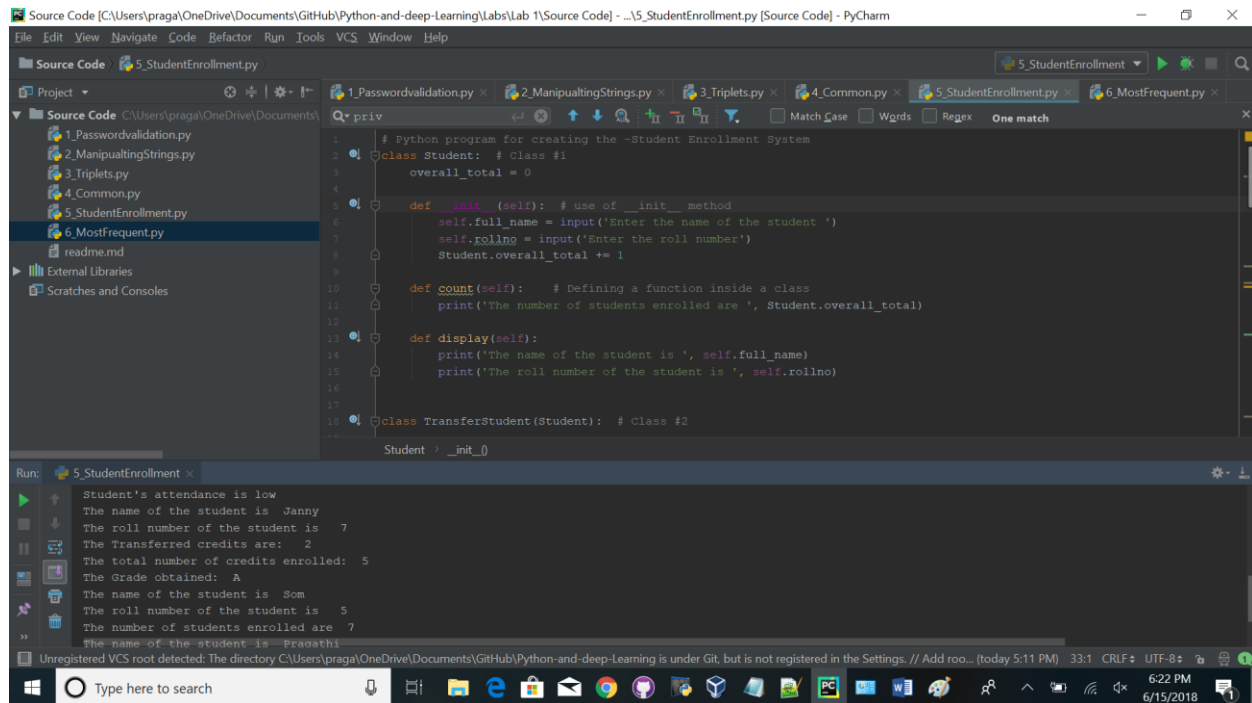
This snippet code provides the above implementation for student enroll system. The below snippet consist of 5 different classes, `__init__` constructor is implemented and also includes the inheritance, super call, private data member which also implements the multiple inheritance. Finally created the instances which shows the relationship between the classes.

Code Snippet:

The below is the code for executing the above workflow

```
1  # Python program for creating the -Student Enrollment System
2  class Student: # Class #1
3      overall_total = 0
4
5      def __init__(self): # use of __init__ method
6          self.full_name = input('Enter the name of the student ')
7          self.rollno = input('Enter the roll number')
8          Student.overall_total += 1
9
10     def count(self): # Defining a function inside a class
11         print('The number of students enrolled are ', Student.overall_total)
12
13     def display(self):
14         print('The name of the student is ', self.full_name)
15         print('The roll number of the student is ', self.rollno)
16
17
18     class TransferStudent(Student): # Class #2
19
20         def __init__(self):
21             super(TransferStudent, self).__init__() # use of super() call
22             self.TransferredCredits = input('Enter the number of credits that are transferred')
23
24
25     class System: # Class #3
26
27         def __init__(self):
28             self.TypeOfSystem = input('Enter the system online or inclass: ')
29
30         def display(self):
31             print('The system the student enrolled is: ', self.TypeOfSystem)
32
33
34     class Grades(TransferStudent): # Class #4
35
36         def __init__(self, grade, credits):
```

Output Screenshot:



Question 6:

6. Using NumPy create random vector of size 15 having only Integers in the range 0-20. Write a program to find the most frequent item/value in the vector list.

Sample input: [1,2,16,14,6,5,9,9,20,19,18]

Sample output: Most frequent item in the list is 9

Solution:

This snippet code provides the above implementation to create a random vector with size 15 which consist only the integers in the range of 0-25. The steps that are followed for the workflow are, the numpy package is installed and imported and generated the random numbers then printed the frequent item in the generated vector list.

Code Snippet:

The below is the code for executing the above workflow

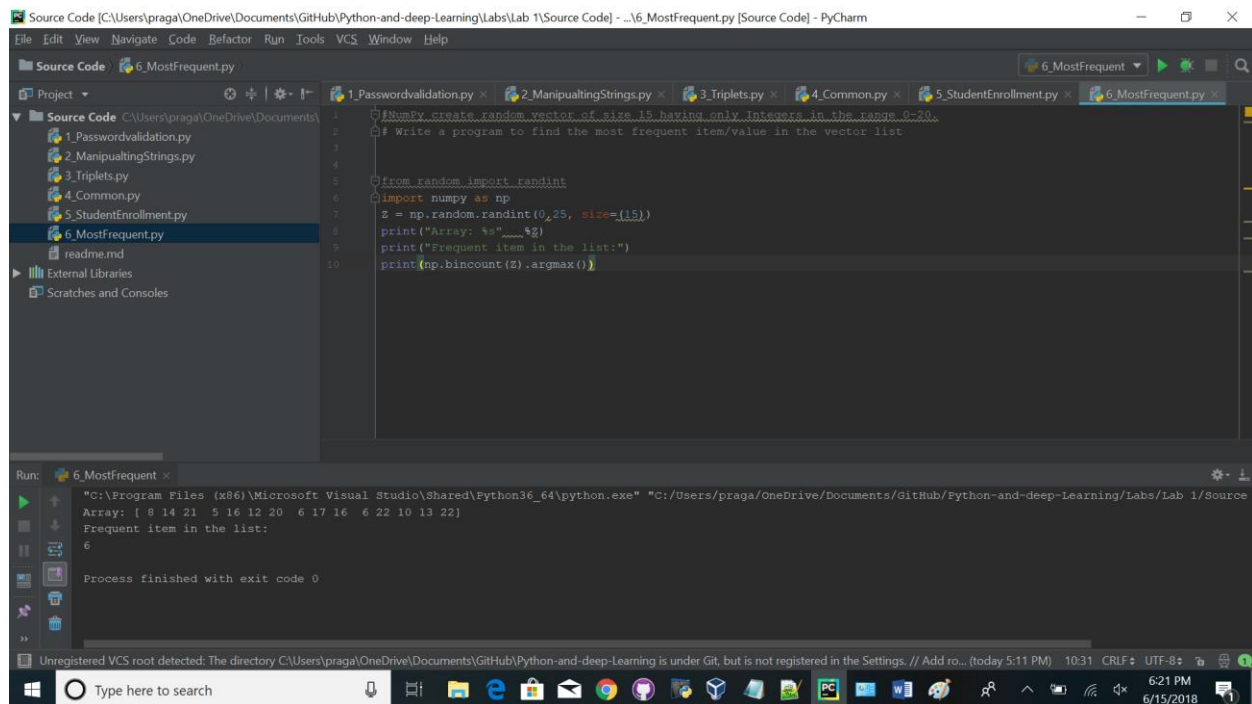
```

10 lines (8 sloc) | 339 Bytes
Raw Blame History

1 #NumPy create random vector of size 15 having only Integers in the range 0-20.
2 # Write a program to find the most frequent item/value in the vector list
3
4
5 from random import randint
6 import numpy as np
7 Z = np.random.randint(0,25, size=(15))
8 print("Array: %s" %Z)
9 print("Frequent item in the list:")
10 print(np.bincount(Z).argmax())

```

Output Screenshot:



The screenshot displays the PyCharm IDE interface. The top pane shows the source code for `6_MostFrequent.py`. The code generates a random vector of size 15 and finds the most frequent item. The bottom pane shows the output of the program.

```
1 #Numpy create random vector of size 15 having only integers in the range 0-20.
2 # Write a program to find the most frequent item/value in the vector list
3
4
5 from random import randint
6 import numpy as np
7 Z = np.random.randint(0,25, size=(15))
8 print("Array: %s"%Z)
9 print("Frequent item in the list:")
10 print(np.bincount(Z).argmax())
```

Run: 6_MostFrequent x

```
"C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36_64\python.exe" "C:\Users\praga\OneDrive\Documents\GitHub\Python-and-deep-Learning\Labs\Lab 1\Source
Array: [ 8 14 21  5 16 12 20  6 17 16  6 22 10 13 22]
Frequent item in the list:
6
Process finished with exit code 0
```

Conclusion:

As stated the above workflow with certain set of parameters is followed in solving the execution by implementing the core and basic concepts of the python programming.

Source code link:

<https://github.com/PragathiThammaneni/Python-and-deep-Learning/tree/master/Labs/Lab%201/Source%20Code>

Video Link: <https://youtu.be/54wLDxGpyOg>

Wiki Link:

<https://github.com/PragathiThammaneni/Python-and-deep-Learning/wiki/Lab-1-Assignment>

