Instructor:

- Mr. Nazeef Ul Haq

Registration No._____

Name: _____

Guide Lines/Instructions:

- Use of Spyder IDE/Anaconda in this lab.
- Create meaningful variable names. Add comments for readability. Indent each line of your code.
- Plagiarism/Cheating is highly discouraged by penalizing to both who tried and one who shared his/her code.

Today's Task:

- Python Environment Setup using Spyder IDE
- Get comfortable with the Python Syntax Specifically Arrays
- Learn to write recursive tasks

Installation Guideline:

- Go to the website Home --- Spyder IDE and click the download button at the bottom or download from the direct link Spyder Install (221 MB).
- Run the setup file according to Figure 1. Complete the installation with the emerging instructions.

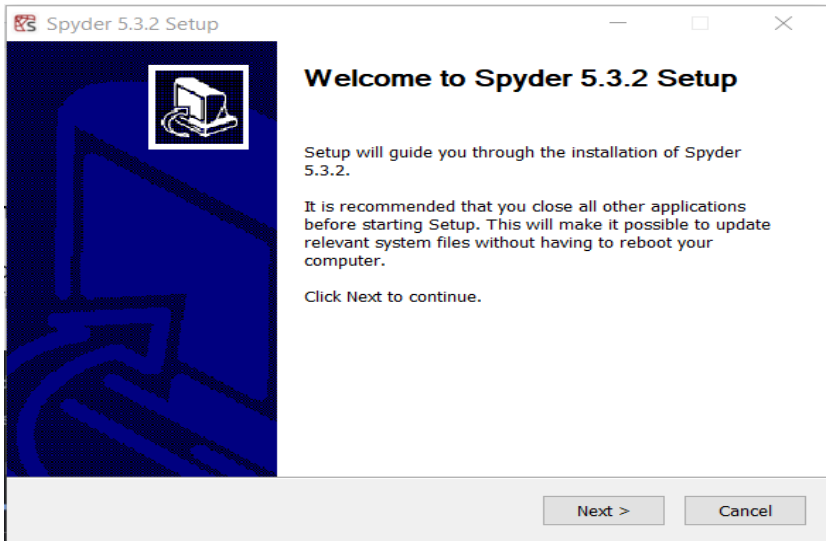

Figure 1 Spyder Installation Interface

- Launch the Spyder from the installation directory.
- You will see the interface according to Figure 2.
- Write your first program in python

```
print ("Hello to Data Structure and Algorithms Course")
```

- Note that python does not require the program template as required in C++ and C#.

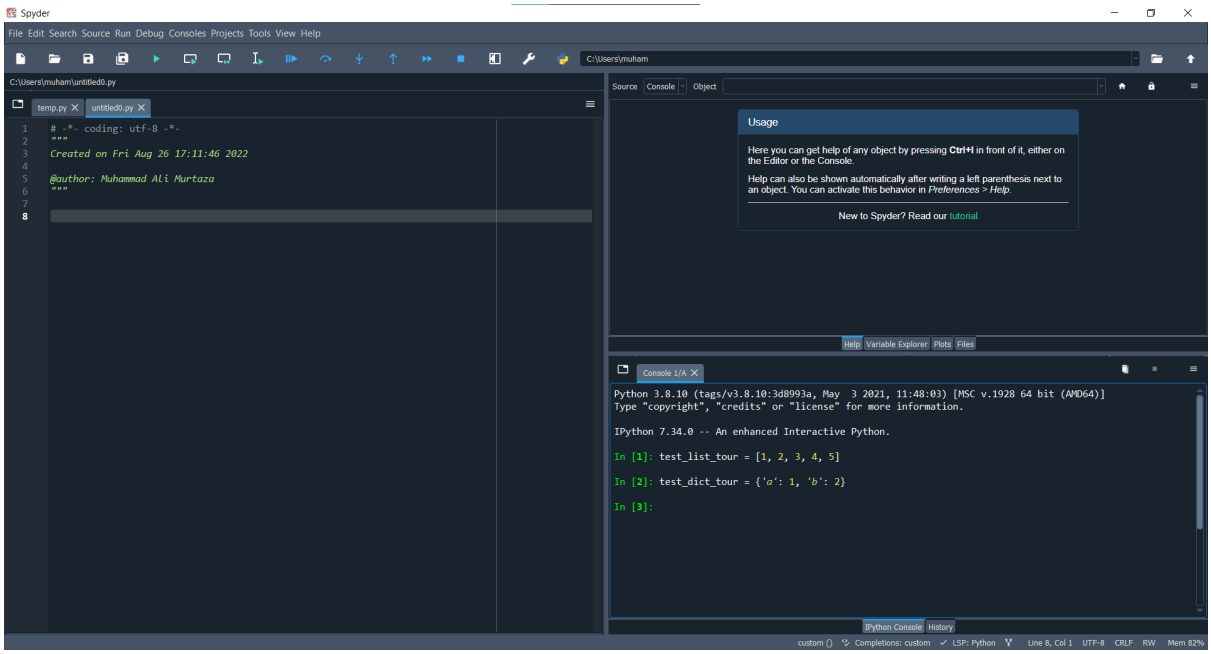

Figure 2 Spyder IDE Home Screen

Part1: Getting Comfortable with Python

Activity 1:

Run the above code using command prompt. Write below the process of running program from cmd (commands).

Activity 2:

Describe the process of code compilation and execution in python. How the byte code will be generated? How the process is different from C#. Write in your own words.

Here are some examples that will provide you the roadmap of Conversion from C# to Python.

Example 1.1: Write a Program to display value

C#	Python

using System;
public class ConsoleApp1
{
public static void Main()
{
// declare variable with int datatype
int a = 5;
// prints the string
System.Console.WriteLine("The value of a is: " + a);
}
}
	# No compulsory library is required for simple programs
	
	a = 5
	
	# no terminator restriction
	
	print("The value of a is: ", a)

Example 1.2: Input value from User

```
a = input (“Enter the value:”) #a is string

#conversion of string to int data type
b = int(a)

print (“Entered value is:” + str(b))

#We need to convert int type variable to string. Because in python concatenation of int and string type variables is not possible without conversion.
```

Example 1.3: Array Declaration of 1D and 2D arrays

```
#1D array declaration
array = []
#Also initialize at the time of declaration
array = [1,2,6,10,4]

#2D array
arr = [[1,3,2], [1,5,6]]
```

Activity 3:

We do not specify the data type of variable in python. How python will infer the data type. How will you verify the data type of variable in python. Give convincing justification.

Example 1.4: Array of Zeros

```
array = 0 * 10 #array of length 10 having all zeros

#2D array having all zeros
array1 = [[0 for x in range(4)] for y in range(3)]

#we can perform the same task more easily using numpy library
TODO–add numpy code for zeros
```

Example 1.5: 1D array of Random ints

```
import random
array = []
min = 0
max = 20
n = 5
for i in range (0, n):
      num = random. randint (min, max)
    array. append (num)

#Process the same using numpy
--to-do add numpy code
```

Example 1.6: Traversal of an array

```
#Traverse in forward direction using
for loop
str = [“U”, “E”, “T”]
  for x in range(len(str)):
    print(str[x])

array = [32, 1, 9, 31, 12, 22]

# Reverse by using a slice
# slice (start, end, step)
print(array[::-1])
```

```
#Traverse in backward direction using reverse
method
array.reverse()
print(array)

#Traverse through an array using for loop
for i in range(len(array)-1, -1, -1):
      print(array[i])
```

Example 1.7: Slicing of Arrays—Extracting subarrays

Slicing in Python is a feature that enables accessing parts of sequences like strings, tuples, and lists. Here are some examples on arrays that would explain slicing.

Starting subarray	Middle subarray	Ending subarray

>>> arr = [1,2,3,4,5]
>>> arr [:2]
[1, 2]
	>>> arr = [1,2,3,4,5]
	>>> arr [1:3]
	[2, 3]
		>>> arr = [1,2,3,4,5]
		>>> arr [2:]
		[3, 4, 5]

Important

```
In python, subarrays can also be extracted through negative indices.
>>> arr = [1,2,3,4,5]
>>> arr [-2:]
[4, 5]
```

Here -1 means the first element from last. -2 means second element from last.

Example 1.8: Read data from File

Let say we have single file for this code testing named **test.txt** and we have single line written in it as:

```
given_file = open (file = 'test.txt', mode = 'r')
lines = given_file. read ()

numbers = []
arr = lines.split()
for s in arr:
    num = int(s)
    numbers.append(num)

print(numbers)
```

test.txt

```
1
2
3
4
5
```

Note:

Example modes for opening file in python are:

w → Write mode

r → Read Mode

a → Append Mode

x → Open for exclusive creation, failing if the file already exists

Example 1.9: Write data to File

```
#Write array elements one per line to file

arr = ['Hello world', 'UET']
f = open (file="test.txt", mode="w")
for i in arr:
    f.write (i + "\n")
```

Output:

```
Hello World
UET
```

Example 1.10: Play with functions

Pass an array to function

```
def display(arr):
    for i in arr:
        print(i)

array = [1, 2, 3, 4, 5, 6, 7, 8, 9]
display(array)
```

Return array from function

```
def get_name ():
    names = ['Ali', 'Ahmad', 'Hassan']
    return names

names= get_name ()
```

Activity 4:

What are mutable and immutable data types in python. Give at least three examples for each.

Part 2: Think Recursively

Example 2.1: Calculate sum of integers

Iterative	Recursive

`sum = 0`
`for i in range (11):`
`    sum += i`
 	
`print(sum)`
	`def sum(n):`
	`    if n == 0:`
	`        return n`
	`    else:`
	`        return n + sum(n-1)`
	`print (sum (10))`

Example 2.2: Print array of elements

Iterative	Recursive

`arr = [1,2,3,4,5,6,7,8,9,10]`
 	
`for i in arr:`
`    print(i)`
	`def printArray (arr, start, end):`
	`    if start == end:`
	`        print(arr[start])`
	`    else:`
	`        print(arr[start])`
	`        printArray (arr, start+1, end)`
	
	`arr = [1,2,3,4,5,6,7,8,9,10]`
	`printArray (arr, 0, len(arr)-1)`

Example 2.3: Calculating power function through recursion.

Iterative	Recursive

`num = 2`
`power = 5`
`result = 1`
`for i in range(power):`
`    result = result * num`
 	
`print(result)`
	`def power (n, k):`
	`    if k == 1:`
	`        return n`
	`    else:`
	`        return n * power (n, k-1)`

Example 2.4: Factorial of Number using recursion

```
def recur_factorial(num):
    if num < 0:
       return -1
    elif num == 0 || num == 1:
         return 1
    else:
       return n*recur_factorial(n-1)
num = int(input("Enter a number: "))
print("The factorial of",num,"is",recur_factorial(num))
```

Activity 5:

What is recursion? Give some prose and cons of recursion.

Activity 6:

How recursive function is evaluated in memory. Give some details

Note:

1. Whenever you are asked to read array, you are required to load array from the file, reading each element per line.
2. Do not take input from console for array.

Problems

1. Look for the index of the given element x in the given array:
`X = [22,2,1,7,11,13,5,2,9]`
 	
`SearchA(Arr, x) – return array of indices`
 	
`Arr: Array`
`x: element to be searched`
	Input: Enter the number: 2
	Output: `Index: 1,7`

2. Answer question 1 in the scenario where the input array is already sorted.
How much elements you need to check in sorted array.
 	
`SearchB(Arr, x)-- return array of indices`
 	
`Arr: Array`
`x: element to be searched`
	Input: Enter the number: 2
	Output: `Index: 1,7`

3. Write a function that takes an array as input, starting and ending index and return the index of minimum element from start to ending index in the array. `Minimum(Arr, starting, ending)– return integer`	For example, you are given the following inputs Array: [3,4,7,8,0,1,23,-2,-5] StartingIndex: 4 EndingIndex: 7 Output: (Return index of minimum element) 7

4. Sort an array X using the above generated function.
 	
Hint: Find the smallest element from the unsorted part of the array repeatedly and place it at the start of the array.
 	
`Sort4(Arr)–return array`
`Arr: Array to be sorted`
	Output: `X = [-5, -4, -3, 0, 1, 1, 4, 35, 100, 101]`

5. Extract the relevant portion and print it in the reverse direction from the string s = **"University of Engineering and Technology Lahore"**. Without using any loop and reverse () method. `StringReverse(str, starting, ending)–returns string`	**Output**: "ygolonhceT dn"

6. Given a number, the task is to find the sum of its digits using an iterative and recursive method.
 	
`SumIterative(number) – returns integer`
 	
`SumRecursive(number)-- returns integer`
	Input: 1524
	Output: `Sum of digits is: 12`

7. Find the sum of the given matrix both column- and row-wise.
 	
$A = \begin{bmatrix} 1 & 13 & 13 \\ 5 & 11 & 6 \\ 4 & 4 & 9 \end{bmatrix}$
 	
`ColumnWiseSum(Mat) – returns 1d array`
`RowWiseSum(Mat) – returns 1d array`
	Output: `Row-wise:` 27
	22
	17
	
	`Column-wise:` 10 28 28

8. Without using any sorting methods, combine two sorted arrays keeping the resultant array sorted in ascending order.
 	
`A = [0,3,4,10,11]`
`B = [1,8,13,24]`
 	
`SortedMerge(Arr1, Arr2) – returns sorted array`
	Output: `[0,1,3,4,8,10,11,13,24]`

9. Write a recursive function that takes a string and returns if the string is palindrome or not. `PalindromRecursive(str)- returns a boolean`	**Input:** `"radar"` **Output:** Palindrome
10. Sort the given array so that the elements are arranged in the following way while taking ascending order into consideration `Sort10(Arr)–returns array`	**Input:** `[10, -1, 9, 20, -3, -8, 22, 9, 7]` **Output:** `[-8, 7, -3, 9, -1, 9, 10, 20, 22]`

What to Submit:

1. Only .py files are allowed.
2. For Lab1, you are required to write all functions in single file, funcs.py
3. For each problem, create a driver .py file
    a. Lab1.py
4. Functions names input and output should be exactly same.
5. Zip all files, and submit on eduko