LAB 01: Revision of Basic Programming Constructs

CS211 – Data Structures and Algorithms
Usman Institute of Technology
Fall 2020

- **How to submit:**
  - Submit a **single** .py file on MS Teams. (No other format will be accepted)
  - File name should be saved with your roll number (e.g. 19b-001-SE.py)

Write functions in Python whose parameters and return value are given below.

1. Create a function **EvenList** that takes a parameter **n** to input **n** number from users and returns the list of only even numbers.

```
def EvenList(n):
    // your code goes here
```

Example: if user gives the input 1,2,3,4,5,6 then list contains only 2,4,6

2. Create a function **Max** that takes a list as a parameter and returns the maximum element in the list.

```
def Max(list):
    // your code goes here
```

Example: if given list contains [2,4,6] then function must return 6

3. Create a function **Min** that takes a list as a parameter and returns the minimum element in the list.

```
def Min(list):
    // your code goes here
```

Example: if given list contains [2,4,6] then function must return 2

4. Create a function **Last** that takes a list as a parameter and returns the last element in the list. Can you write this function without using the loop?

```
def Last(list):
    // your code goes here
```

Example: if given list contains [2,4,6] then function must return 6

5. Create a function **KElement** that takes a list and a number **k** as a parameter and returns the k^{th} element in the list. Can you write this function without using the loop?

```
def KElement(list,k):
    // your code goes here
```

Example: if given list contains [2,4,6] and k=1 then function must return 4

6. Create a function **SecondLast** that takes a list as a parameter and returns the second last element in the list. Can you write this function without using the loop?

```
def SecondLast(list):
    // your code goes here
```

Example: if given list contains [2,4,6,14] then function must return 6

7. Create a function **Reverse** that takes a list as a parameter and returns a list which is reverse of the original list.

```
def Reverse(list):
    // your code goes here
```

Example: if given list is [2,4,6] then function must return [6,4,2]

8. Create a function **Unique** that takes a list as a parameter and returns a list containing only unique elements i.e. duplicate elements should be removed.

```
def Unique(list):
    // your code goes here
```

Example: if given list contains [2,4,4,6,6] then function must return [2,4,6]

9. Create a function **UserNumbers** that takes 10 number as input from the users but stores only even numbers. The function also prints the following from given number.
   a. The last element of the list
   b. The maximum value
   c. The minimum value
   d. The second last element of the list

```
def UserNumbers(list):
    // your code goes here
```

10. Create a function **ShowExcitement** that returns the string "A quick brown fox jumps over the lazy dog" 5 times. Make sure to separate the sentence with space every time. Don't copy paste the sentence 5 times.

```
    def ShowExcitement(list):
        // your code goes here
```

11. Create a function **Greater** which takes three numbers as parameters and returns the largest numbers.

```
    def Greater(n1, n2, n3):
        // your code goes here
```

12. Create a function **Divide** that takes two numbers, *dividend* and *divisor*, as parameters and returns the *quotient* and *reminder*.

```
 def divide(dividend, divisor):
        // your code goes here
```

Quotient: In arithmetic, a quotient is the quantity produced by the division of two numbers.
Remainder: In mathematics, the remainder is the amount "left over" after performing some computation

Example: Divide(10,3) must return (3,1) because 3 is quotient and 1 is remainder.

13. Create a class **Person** which takes two parameters to initialize: name and age. The class should also have a function **birthday()** which increases the age by 1 year.

```
class Person:
     def_init_(self, name, age):
        // your code goes here

     def birthday():
        // your code goes here
```

Home Task

Polar Co-ordinates are an alternative way of representing Cartesian Co-ordinates or Complex Numbers. A complex number z is represented by:

$$z = x + yj$$

Where x is the real part and y is the imaginary part.
j is the imaginary unit.

A polar co-ordinate (r, φ) is completely represented by its modulus r and phase angle φ. If we convert complex number z to its polar co-ordinate, we find:

r: Distance from z to origin i.e. $\sqrt{x^2 + y^2}$
φ: Counter clockwise angle measured from the positive x -axis to the line segment that joins z to the origin.

Write a function **PolarCoordinates** that takes a complex number as input and converts it into its polar coordinates. The input and output format is given below:

(Hint: Python's ***cmath*** module gives access to the mathematical functions of complex numbers)

```
def PolarCoordinates(z):
      //your code goes here

z = 1+2j
PolarCoordinates(z)

Output:
r = 2.236
φ = 1.107
```