**Learning Journey-02** **Functional Programming in SCALA**
**Instructor: Engr. Farhan Ahmed Karim** **Assignment-02**

## Task-1 Do the following

**Exercise 1:** In Scala, there are many ways to do something, similarly, there are many ways to make an Array. Think of other ways to make an Array. Write them down and test them to see if they work.

**Exercise 2:** Set and Map collections can also be mutable. Figure out how to make them mutable.

**Exercise 3:** Figure out how to use Option type in a function parameter.

**Exercise 4:** In this session we saw some methods like .head and .tail that can be applied on Lists and Arrays. Some of these methods are called higher order methods and we can pass a function in these method as parameter. Explore the higher order methods available in scala for arrays. This will be helpful in doing task 4 in assignments.

## Task-2 Solve the following

**Task 1:** Generate a list of 15 integer numbers generated randomly from 50 - 500. After making the complete list check if each element is prime or not, if its a prime number then put it into an iterator. Finally sort them in ascending order and put them into a Map. Where each key should be the element location of the number. Bonus point for anyone who does not uses for loop.

**Task 2:** Write a function that returns a List[Char] that contains 'a'-'z' using tail recursion. The only argument which is passed to the method is the start alphabet array in ascii i.e 98. Hint: Use toChar to make this work.

```
def CharArray ( start : Int ) : List [ Char ]={

// code here

}
```

**Task 3:** Given two Array[Double] values of the same length, write a function that returns the element-wise sum. This is a new Array where each element is the sum of the values from the two input arrays at that location. So if you have Array(1,2,3) and Array(4,5,6) you will get back Array(5,7,9).

**Task 4:** Code different techniques that will take an Array[Int] and return number of even values in the Array. Each one will use a different technique. To test this on a larger array you can make one using

```
Array . fill (100) ( util . Random . nextInt (100) )
```

1. Use a recursive function.

2. Use the count higher-order method.

**Task 5:** Implement the following function that will build a Map from any sequence of a type with a

function that can make keys from values.

```
def buildMap [ A , B ]( data : Seq [ A ] , f : A = > B ) : Map [ B , A ]{

// code here

}
```

where ‘f’ is an user defined function which is passed as parameter. Below is an example of how we can use buildMap method to make a Map collection.

```
// Example

val lst = Array (1 ,2 ,3 ,4 ,5)

def func ( x : Int ) : Boolean = x %2 == 0 // entry is even or not

val result = buildMap ( lst , func )

// Output

// result : Map [ Int , Boolean ] = Map (1 -> false , 2 -> true , 3 -> false, 4 ->True)
```

**Note:** Submission Date is 25th April 2021 by 11:59 P.M.