Lab session 8 - Recursion
CSC 113
King Saud University
College of Computer and Information Sciences

- Do **not** use loops in any method except `main`.
- Do **not** use global variables
- Do **not** give the class `ArrayRecursor` any attributes
- Do **not** use **static** variables in any method.

1 Array manipulation

You will write the class `ArrayRecursor` which implements some recursive and `static` methods, as well as a main method to test them. Your program will maintain an integer array of maximum size 10 and offer a menu of the following choices to the user *until* the user chooses to quit:

```
1) Fill new array.
2) Count elements.
3) Calculate sum of elements.
4) Print the array.
5) Print the array in reverse order.
6) Quit
Enter a choice:
```

You may use method **overloading** to provide a cleaner interface for each method. In this case, your recursive helper method should be `private`. For example:

```
public static int sum(int array[]) {        //Overloaded, interface method
  return sum(array, 0);
}
private static int sum(int array[], int start) {   //Helper recursive method
  .
  .
  .
}
```

Write the following `static`, recursive methods.

1. `int fill(int [], int )`: which receives an array of `int` and a starting index x. This method will fill the array from x with numbers given by the user untill the user enters -1 or there is no more room in the array. The method will return the number of integers entered by the user.

```
1) Fill new array.
2) Count elements.
3) Calculate sum of elements.
4) Print the array.
5) Print the array in reverse order.
6) Quit
Enter a choice: 1
Enter number 1: 3
Enter number 2: 4
Enter number 3: 1
Enter number 4: 0
Enter number 5: -1
You entered 4 numbers.
```

- You may need to reset the int array in your main function before filling it each time.

2. `int count(int [], int)` which receives an array of `int` and a starting index x, it returns a count of the array's length.

   - Do *not* use the attribute **length** of the array.

```
1) Fill new array.
2) Count elements.
3) Calculate sum of elements.
4) Print the array.
5) Print the array in reverse order.
6) Quit
Enter a choice: 2
The array is of size 4.
```

3. `int sum(int [], int)` which receives an array and a starting index , then recursively calculates the sum of the array's elements beginning at the given index.

```
1) Fill new array.
2) Count elements.
3) Calculate sum of elements.
4) Print the array.
5) Print the array in reverse order.
6) Quit
Enter a choice: 3
The array has sum of :8
```

4. `void printArray(int [], int)` which receives an array and a starting index, then recursively prints the elements in the array.

```
1) Fill new array.
2) Count elements.
3) Calculate sum of elements.
4) Print the array.
5) Print the array in reverse order.
6) Quit
Enter a choice: 4
The array is : [3,4,1,0]
```

5. `void printReverse(int [], int)` which receives an array of integers and returns it in reverse order.

```
1) Fill new array.
2) Count elements.
3) Calculate sum of elements.
4) Print the array.
5) Print the array in reverse order.
6) Quit
Enter a choice: 5
The array in reverse order is [0,1,4,3]
```