Optional Assignments

- Try only if you are very comfortable with classwork and compulsory assigments.
- Try only if time is available.

1. Create a functional `interface Check<T>` with single abstract method `boolean compare(T x, T y)`. Create a static method in main class to test array elements `static <T> int countIf(T[] arr, T key, Check<T> c)`. This method should return count of elements in the array for which given check is satisfied. The check will be given as lambda expression. Example call to countIf() from main() will be as follows.

```
Integer [] arr = {44, 77, 99, 22, 55, 66};
Integer key = 50;
int cnt = countIf(arr, key, (x,y)-> x > y);
System.out.println("Count = " + cnt); // 4 (because 4 elements in array are greater than given key i.e. 50)
```

2. In above assignment, create one more array of Double (constant values) where few elements are repeated. Input a key from user and check how many times key is repeated in the array using appropriate lambda expression.

3. Create a user-defined class to create stack of Integers. Provide push() and pop() methods that follows LIFO behaviour. Write a forEach() method in the class that inputs a `Consumer<Integer>` and consumes each element in the stack. From main class, invoke all methods and test them.

```
class IntStack {
    private Integer[] arr;
    private int top;
    // constructor
    public void push(Integer i) {
        // ...
    }
    public Integer pop() {
        // ...
    }
    public void forEach(Consumer<Integer> c) {
        // ...
```

SUNBEAM INFOTECH

```
    }
}
```

```
class Main {
    public static void main(String[] args) {
        IntStack s = new IntStack();
        s.push(10);
        s.push(20);
        s.push(30);
        s.push(40);
        s.forEach(i -> System.out.println(i));
        Integer e = s.pop();
        // ...
    }
}
```

SUNBEAM INFOTECH