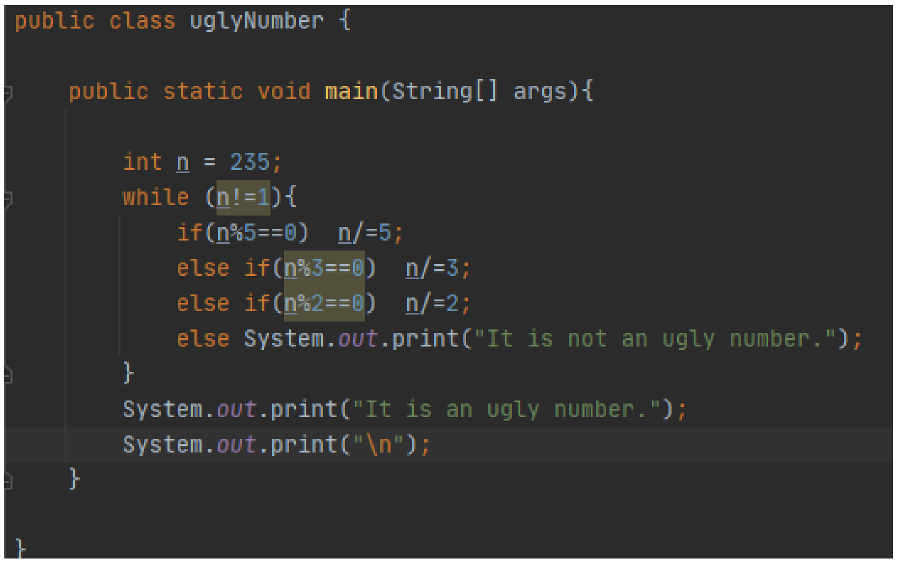
Faculty of Engineering, University of Jaffna
Department of Computer Engineering
EC5080: Software Construction
Lab – 01: Introduction to Java language features

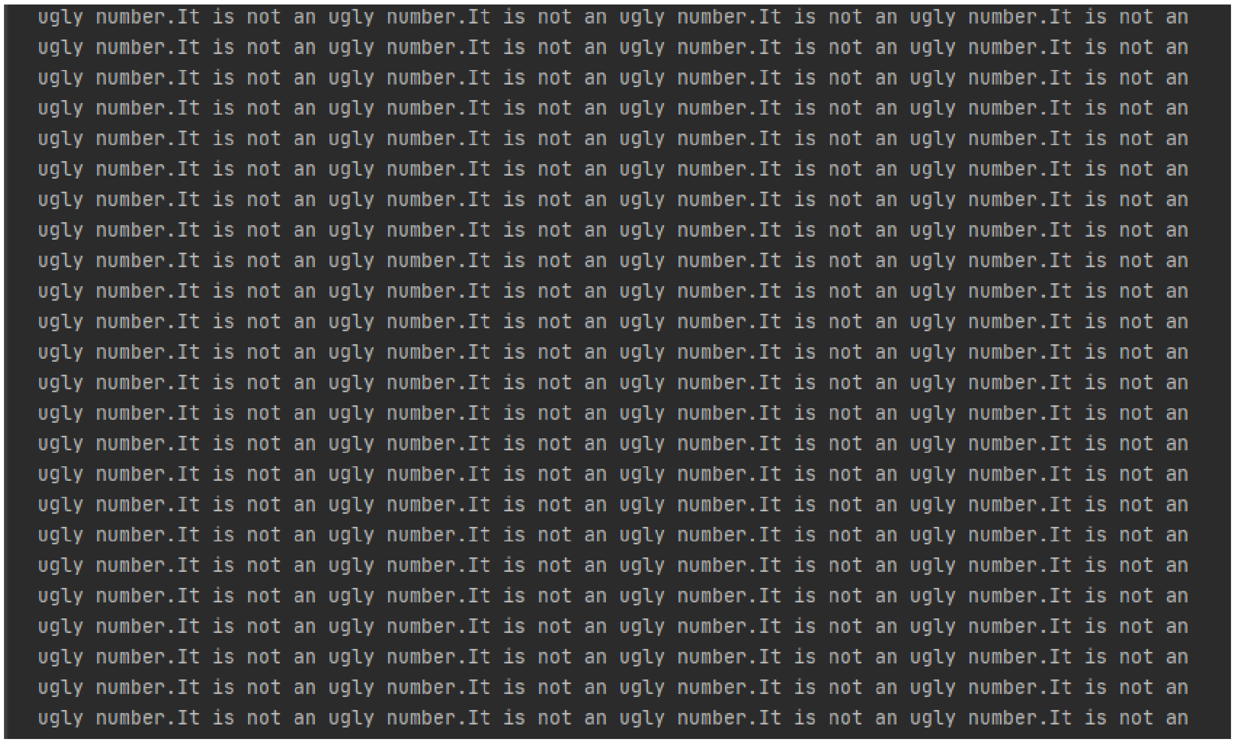
Date: 27th April 2021

1.

I.

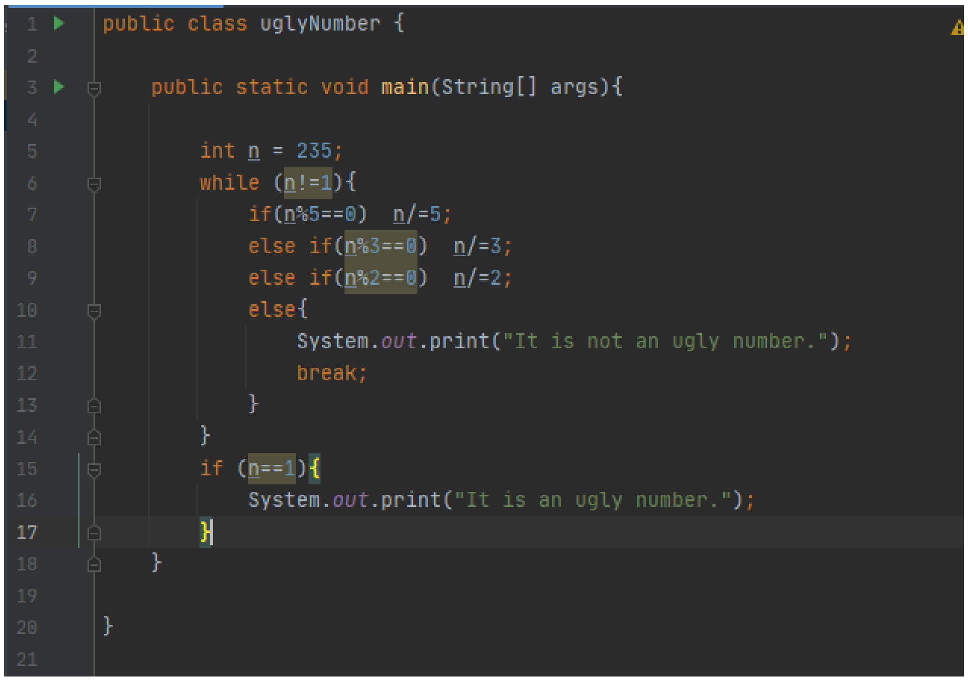


```java
public class uglyNumber {

    public static void main(String[] args){

        int n = 235;
        while (n!=1){
            if(n%5==0)  n/=5;
            else if(n%3==0)  n/=3;
            else if(n%2==0)  n/=2;
            else System.out.print("It is not an ugly number.");
        }
        System.out.print("It is an ugly number.");
        System.out.print("\n");
    }

}
```

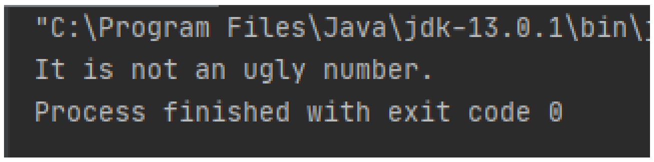


```
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
ugly number.It is not an ugly number.It is not an ugly number.It is not an ugly number.It is not an
```

II.



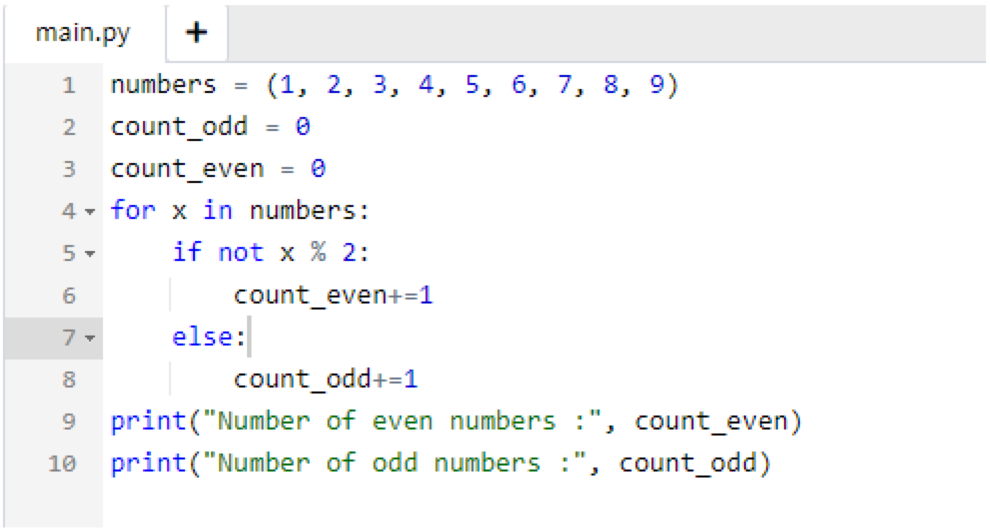
```java
public class uglyNumber {

    public static void main(String[] args){

        int n = 235;
        while (n!=1){
            if(n%5==0)  n/=5;
            else if(n%3==0)  n/=3;
            else if(n%2==0)  n/=2;
            else{
                System.out.print("It is not an ugly number.");
                break;
            }
        }
        if (n==1){
            System.out.print("It is an ugly number.");
        }
    }

}
```



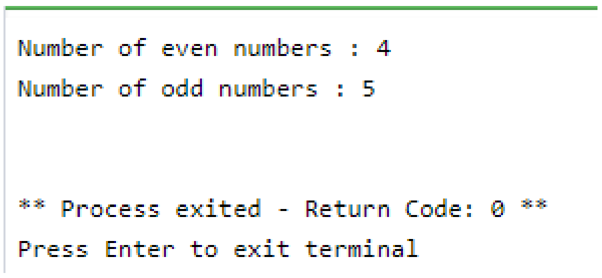
```
"C:\Program Files\Java\jdk-13.0.1\bin\
It is not an ugly number.
Process finished with exit code 0
```

2.

I.



```
main.py

numbers = (1, 2, 3, 4, 5, 6, 7, 8, 9)
count_odd = 0
count_even = 0
for x in numbers:
    if not x % 2:
        count_even+=1
    else:
        count_odd+=1
print("Number of even numbers :", count_even)
print("Number of odd numbers :", count_odd)
```

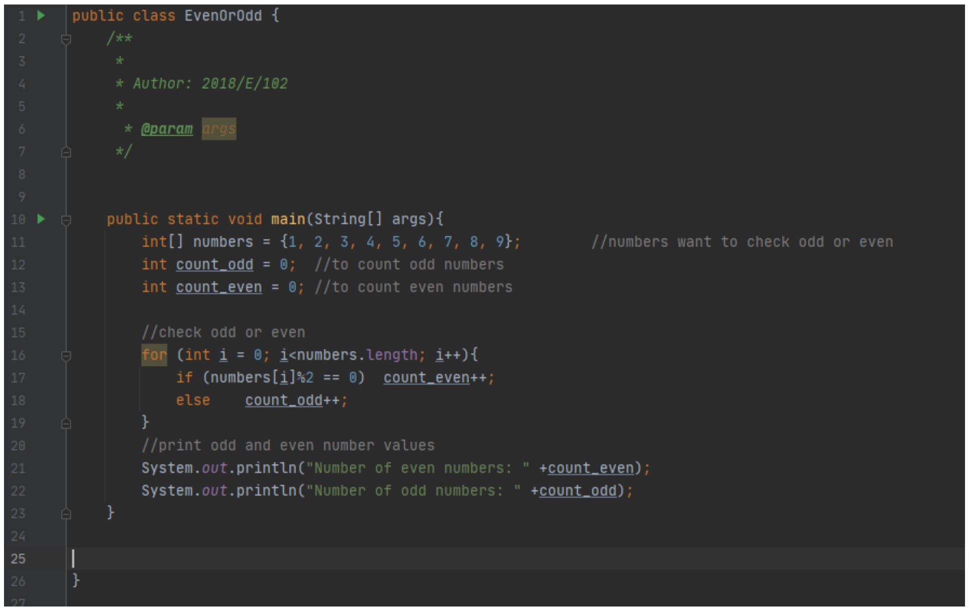


```
Number of even numbers : 4
Number of odd numbers : 5


** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

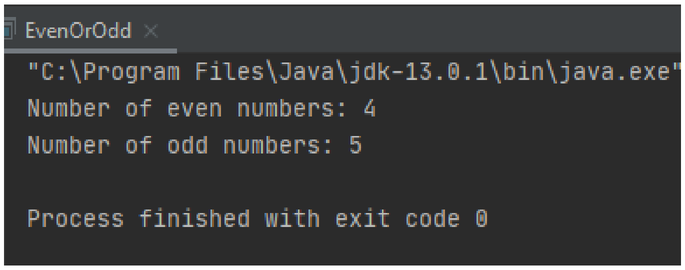
II.

Python is a Dynamically typed language.
It doesn't know about the type of the variable until the code is run. So declaration is of no use. What it does is, it stores that value at some memory location and then binds that variable name to that memory container. And makes the contents of the container accessible through that variable name. So the data type does not matter. As it will get to know the type of the value at run-time. So Python can be called as a dynamically typed language

III.

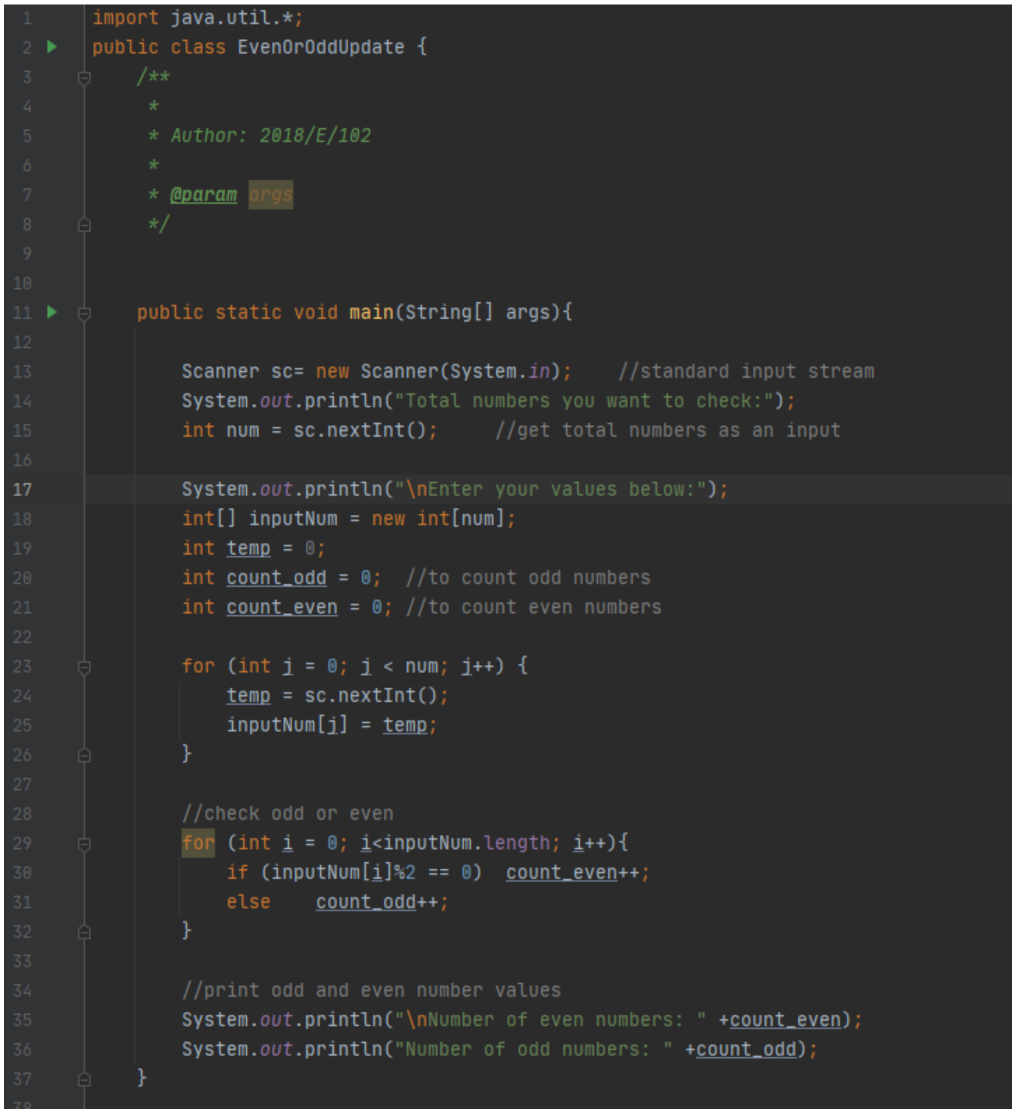


```
public class EvenOrOdd {
    /**
     *
     * Author: 2018/E/102
     *
      * @param args
     */


    public static void main(String[] args){
        int[] numbers = {1, 2, 3, 4, 5, 6, 7, 8, 9};        //numbers want to check odd or even
        int count_odd = 0;  //to count odd numbers
        int count_even = 0; //to count even numbers

        //check odd or even
        for (int i = 0; i<numbers.length; i++){
            if (numbers[i]%2 == 0)  count_even++;
            else    count_odd++;
        }
        //print odd and even number values
        System.out.println("Number of even numbers: " +count_even);
        System.out.println("Number of odd numbers: " +count_odd);
    }


}
```

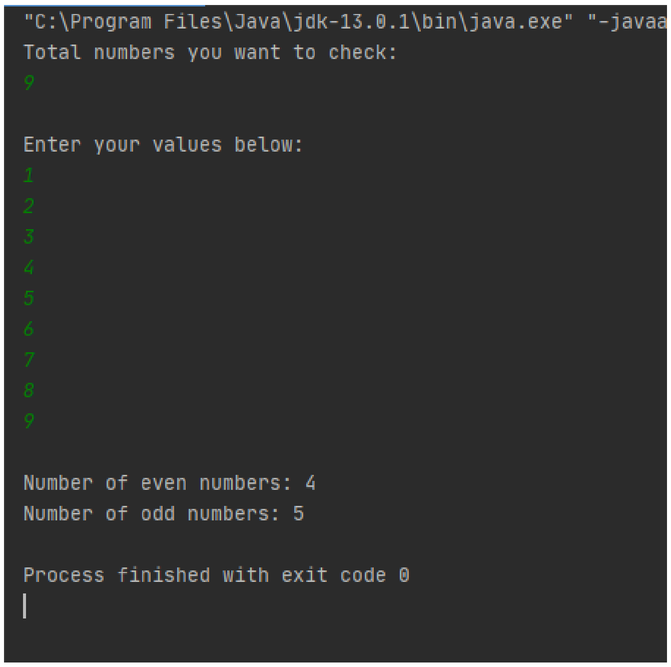


```
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe"
Number of even numbers: 4
Number of odd numbers: 5

Process finished with exit code 0
```

IV.



```java
import java.util.*;
public class EvenOrOddUpdate {
    /**
     *
     * Author: 2018/E/102
     *
     * @param args
     */


    public static void main(String[] args){

        Scanner sc= new Scanner(System.in);    //standard input stream
        System.out.println("Total numbers you want to check:");
        int num = sc.nextInt();     //get total numbers as an input

        System.out.println("\nEnter your values below:");
        int[] inputNum = new int[num];
        int temp = 0;
        int count_odd = 0;  //to count odd numbers
        int count_even = 0; //to count even numbers

        for (int j = 0; j < num; j++) {
            temp = sc.nextInt();
            inputNum[j] = temp;
        }

        //check odd or even
        for (int i = 0; i<inputNum.length; i++){
            if (inputNum[i]%2 == 0)  count_even++;
            else    count_odd++;
        }

        //print odd and even number values
        System.out.println("\nNumber of even numbers: " +count_even);
        System.out.println("Number of odd numbers: " +count_odd);
    }
```



```
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe" "-javaa
Total numbers you want to check:
9

Enter your values below:
1
2
3
4
5
6
7
8
9

Number of even numbers: 4
Number of odd numbers: 5

Process finished with exit code 0
```

3.

I.

Both stack and heap memory are important memory areas, but used for different purposes. The **major difference between Stack memory and heap memory** is that the stack is used to store the order of method execution and local variables while the heap memory stores the objects and it uses dynamic memory allocation and deallocation. In this section, we will discuss the differences between stack and heap in detail.

Stack memory:
The stack memory is a physical space (in RAM) allocated to each thread at run time. It is created when a thread creates. Memory management in the stack follows LIFO (Last-In-First-Out) order because it is accessible globally. It stores the variables, references to objects, and partial results. Memory allocated to stack lives until the function returns

Heap memory:
It stores objects and JRE classes. Whenever we create objects it occupies space in the heap memory while the reference of that object creates in the stack. It does not follow any order like the stack. It dynamically handles the memory blocks. It means, we need not to handle the memory manually. For managing the memory automatically, java provides the garbage collector that deletes the objects which are no longer being used. Memory allocated to heap lives until any one event, either program terminated or memory free does not occur. The elements are globally accessible in the application. It is a common memory space shared with all the threads

II.

temp – Stack memory (Primitive data type variable)
card – Heap memory (Object of CreditCard class)

III.

Java garbage collection is the process by which Java programs perform automatic memory management. Java programs compile to bytecode that can be run on a Java Virtual Machine, or JVM for short. When Java programs run on the JVM, objects are created on the heap, which is a portion of memory dedicated to the program. Eventually, some objects will no longer be needed. The garbage collector finds these unused objects and deletes them to free up memory.

IV.

Advantages of Garbage Collection:

- It makes java **memory efficient** because garbage collector removes the unreferenced objects from heap memory.
- It is **automatically done** by the garbage collector (a part of JVM) so we don't need to make extra efforts.

Disadvantages of Garbage Collection:

- Consumes computing resources in deciding what memory is to be freed
- Reconstructing facts that may have been known to the programmer often leading to decreased or uneven performance.