**Student Name:** Ritik Kumar **Roll No.:53**

**Student ID:** 11808486

**Email Address:** [**Ritikkumar2405199@gmail.com**](mailto:Ritikkumar2405199@gmail.com)

**GitHub Link:** <https://github.com/ritikkr/OSPROJECT>

**Code of Question No 18:**

import java.util.Scanner;  //Scanner class is use for taking user input

import java.util.Random; //Random class is use to produce random number between given range

class OSPROJECT{

        public static *void* main(*String* *args*[]){

            /\* Hey its me Main function, Your program is nothing without me \*/

            //this is main function which is intially load by JVM automatically

*Scanner* input=new Scanner(System.in);  //Scanner object refernce

        System.out.print("Enter No of user: ");

*int* no\_of\_user=input.nextInt();   //this variable will store no of user

            giftShop(no\_of\_user);  //calling giftShop function

        }

        static *void* giftShop(*int* *no\_of\_user*){

            /\* Hello there I'm giftShop function :-) \*/

            //I'm going to generate random number of gifts for given number of users

*Random* rand=new Random();             //Random class object refernce

*int* gifts[]=new *int*[no\_of\_user]; // Integer array to store no of gifts

            for(*int* i=0;i<no\_of\_user;i++){

                gifts[i]=rand.nextInt(50); //this will generate random no of gift between 0 to 50

            }

             billingCounter(gifts,no\_of\_user);  //calling billingCounter

        }

        static *void* billingCounter(*int* *gifts*[],*int* *no\_of\_user*){

            /\* Hey I'm billingCounter function :-) \*/

            /\*this function will set the users according to their number of gift

            i.e., highest the number highest the priority \*/

            /\* In this function we're making a copy of our original gift array so that we won't loose our original data

            and after we're applying sorting on gift\_copy array \*/

            /\* here we used bubble sorting method in which we're generating pass for 'n-1' person  and

            in each pass one person gets its original position \*/

*int* gift\_copy[]=gifts.clone();

            for(*int* pass=0; pass<no\_of\_user-1;pass++){

*int* flag=0;

                for(*int* i=0;i<no\_of\_user-1-pass;i++){

                    if(gift\_copy[i]<gift\_copy[i+1]){

*int* temp=gift\_copy[i];

                        gift\_copy[i]=gift\_copy[i+1];

                        gift\_copy[i+1]=temp;

                        flag=+1;

                    }

                }

                if(flag==0){

                    /\* if flag becomes 0 means our array is now sorted so need to go further it is used to reduce the complexity \*/

                    break;

                }

            }

                display(gifts,gift\_copy,no\_of\_user);  // calling the display function for displaying the output

        } //end of billingCounter function

       static *void* display(*int* *gifts*[],*int* *gift\_copy*[],*int* *no\_of\_user*) {

            /\* Hey I'm Display function :-) \*/

            /\* I'm going to display output for your program \*/

            /\* here we have two arrays one is sorted and another one that is our original data so we're going to match

            one by one data of sorted array with original data \*/

           for(*int* i=0;i<no\_of\_user;i++){

               for(*int* j=0;j<no\_of\_user;j++){

                   if(gift\_copy[i]==gifts[j]){

                    try {

                        // thread to sleep for 1000 milliseconds

                        Thread.sleep(1000);

                     } catch (*Exception* *e*) {

                        System.out.println(e);

                     }

                       System.out.println((j+1)+" person have "+(i+1)+" priority with gift no: "+gifts[j]+" \n");

                       break;

                   }

               }  //end of j

           } //end of i

        System.out.println("------------------------------------------");

        System.out.println("\tExecution Completed ");

        System.out.println("------------------------------------------");

       }//end of display function

}

**Questions:**

**Question 1:** Explain the problem in terms of operating system concept? (Max 200 word)

**Answer:** This algorithm can be use in priority based scheduling algorithm where each process are assigned with some priority. Priority scheduling is a method of scheduling processes based on priority. In this method, the scheduler chooses the tasks to work as per the priority, which is different from other types of scheduling, for example, a simple round robin.

Priority scheduling involves priority assignment to every process, and processes with higher priorities are carried out first, whereas tasks with equal priorities are carried out on a first-come-first-served (FCFS) or round robin basis. An example of a general-priority-scheduling algorithm is the shortest-job-first (SJF) algorithm.

**Question 2:** Write the algorithm for proposed solution of the assigned problem.

**Answer:**

**=>** giftshop(no\_of\_user)

1. Initialize gifts[no\_of\_user];

2. Repeat While i<no\_of\_user

2.1 Assigned gift[i] with random number

3.Exit

=>billingCounter(gifts,no\_of\_user)

1.Intialise pass=0, i=0

2.Copy gifts Array into gift\_copy array

3. Repeat while pass<no\_of\_user-1

3.1 Intialise flag=0

3.2 Repeat while i<no\_of\_user-1-pass

3.2.1 check if gift\_copy[i] < gift\_copy[i+1] then swap gifts[i] with gifts[i+1]

3.2.2 flag=flag+1

3.3 chech if flag=0 then break

4.Exit

=>Display(int gifts[],int gift\_copy[],int no\_of\_user)

1.Initilize i=0 and j=0

2. Repeat while i<no\_of\_user

2.1 Repeat while j<no\_of\_user

2.1.1 check if (gift\_copy[i]=gifts[j]) then

Print details of person and sleep for some seconds

2.1.2 break

3. Print “Execution Completed”

4.Exit

**Question 3:** Calculate complexity of implemented algorithm. (Student must specify complexity of each line of code with overall complexity)

**Answer:**

The Main Function have complexity of O(1) time.

The Function giftshop has complexity of O(N) where N is no of user

The Function billingCounter has complexity of O(N2) where again N is no of user

The Function Display have complexity of O(N2) where N is no of user

**Question 4:** Explain all the constraint given in the problem. Attach the code snippet pf the implemented constraint.

**Answer:** As given problem is totally depend upon no of users so only constraint yhat canbe there is no of users should not be more than 32,767

Code Snippet:



**Question 5:** If you have implemented any additional algorithm to support the solution explain the need and usage of the same.

**Answer:** Yes I have implemented a One algorithm that is

import java.io.\*;

public class random{

public static class p{

}

static *long* reg=0;

static *long* lfsr()

{

    if(reg==0)

    {

        reg=145896027340307l;

    }

*long* bit=(reg>>0^reg>>2^reg>>3^reg>>5)&1;

    reg=reg>>1|bit<<62;

    return reg;

}

static *long* getRand()

{

*String* s=String.valueOf(new p());

    //System.out.println(s);

*long* n=0;

    lfsr();

    for(*int* i=0;i<s.length();i++)

    {

        n=n<<8|+s.charAt(i);

    }

    System.out.print(n+" "+System.currentTimeMillis()+" "+reg+" ");

    n=n^System.currentTimeMillis()^reg;

    return n;

}

public static *void* main(*String* *args*[])throws *IOException*

{

    for(*int* i=0;i<400;i++)

    {

        System.out.println(getRand());

    }

}

This is a random number generator where it is guaranteed that the sequence never repeats itself. I have paired time with object value (randomly put by java) with LFSR.

Advantages:

* The sequence doesn't repeat itself
* The sequence is new on every run

Disadvantages:

* Only compatible with java. In C++, new object that is created is same on every run.
* But there too time and LFSR parameters would put in enough randomness
* It is slower than most PRNGs as an object needs to be created everytime a number is needed

**Question 6:** Explain the boundary condition of the implemented code.

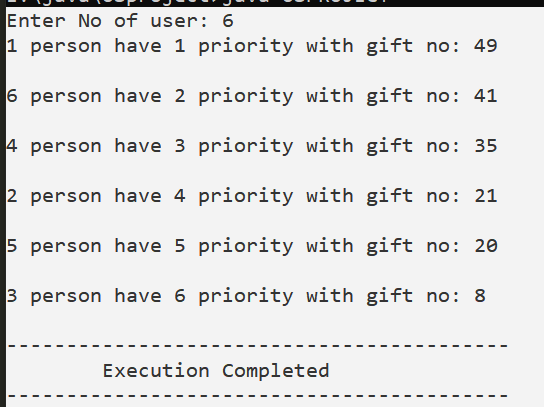
**Answer:** All the Variables can store values between 32,767, but I have prohibited using negative values as no of user can’t be negative. So the values can range from 0 to

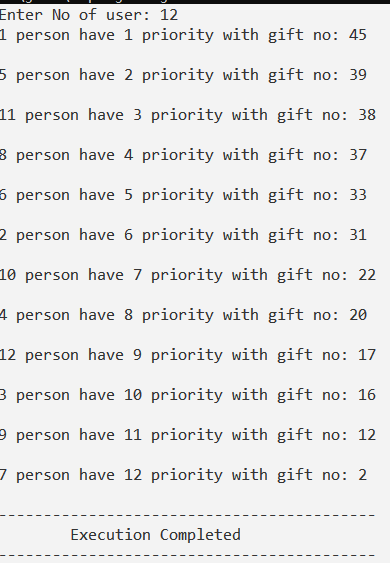
32,767.

**Question 7:** Explain all the test cases applied on the solution of assigned problem.

**Answer:**

**Input:** No of users: 6





**Question 8:** Have you made minimum 5 revision of solution on GitHub?

**Answer: Yes.**

**GitHub Link:** <https://github.com/ritikkr/OSPROJECT>