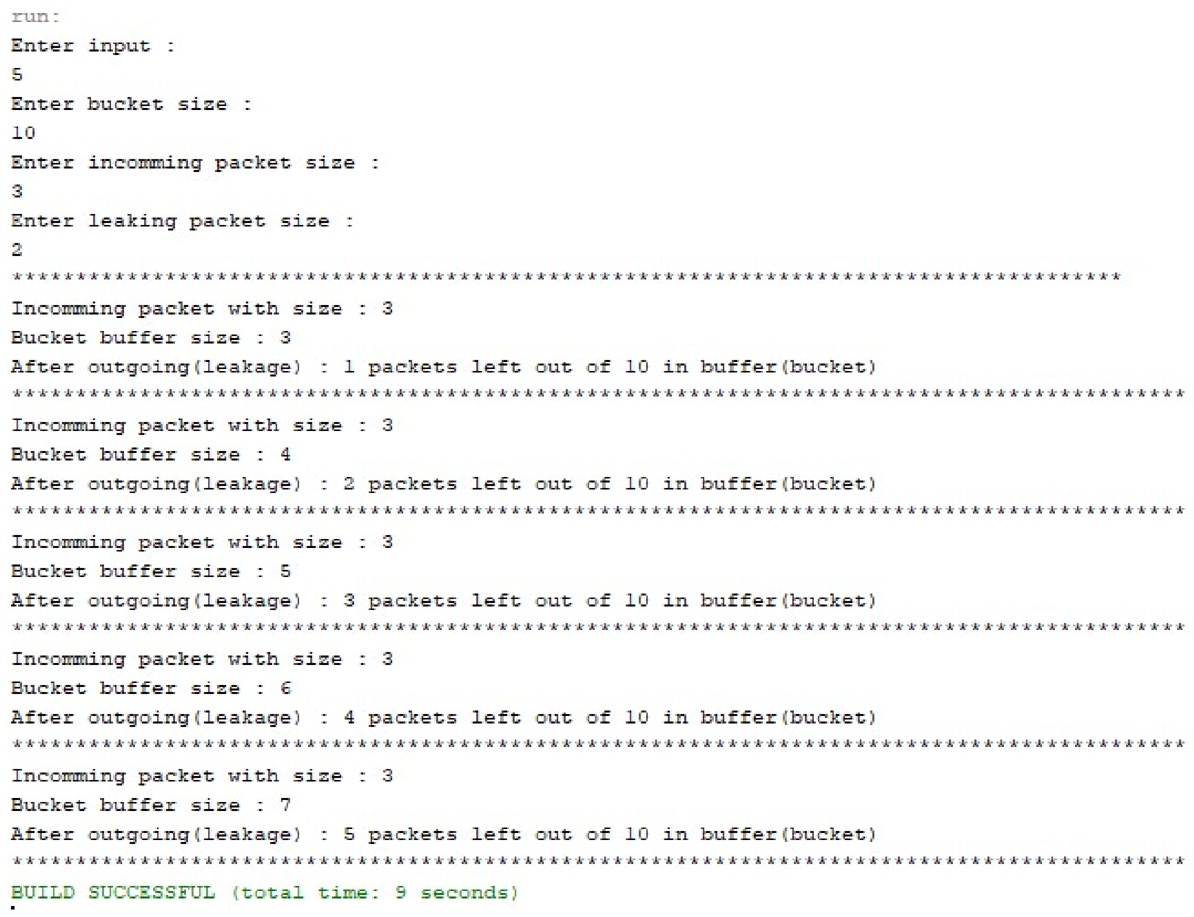

NAME : ATHARVA PALIWAL
ROLL NO : 40
EXPERIMENT NO : 07

AIM : Write a program to implement Leaky Bucket and Token Bucket algorithm.

<u>LEAKY BUCKET CODE -</u>

```
import java.util.Scanner;

/**
 *
 * @author Atharva Paliwal
 */

public class cnlab7 {

    public static void main(String[] args) {
        //Variable declaration
        int inPktSize , leakedPktSize , bucketSize;
        int count = 0; //amount of data in the bucket

        //Variable initialization
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter input : ");
        int t = sc.nextInt();
        System.out.println("Enter bucket size : ");
        bucketSize = sc.nextInt();

        System.out.println("Enter incomming packet size : ");

        inPktSize =sc.nextInt();

        System.out.println("Enter leaking packet size : ");

        leakedPktSize = sc.nextInt();


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

        while(t != 0)

        {

            System.out.println("Incomming packet with size : " + inPktSize);

            if(inPktSize <= (bucketSize - count))

            {

                // updating the store size

                count = count + inPktSize;

                System.out.println("Bucket buffer size : " + count);


            }

            else

            {

                System.out.println("Packet loss : " + (inPktSize -
(bucketSize - count)));

                // Incomming packet size - Storage of bucket left

                count = bucketSize;

                System.out.println("Bucket buffer size : " + count);


            }

            //Leakage will cause decrease in store

            count = count - leakedPktSize;

            System.out.println("After outgoing(leakage) : "+ count + "
packets left out of " + bucketSize + " in buffer(bucket)");
System.out.println("*********************************************************
***********************************");

            t--;

        }


    }


}
```

OUTPUT-



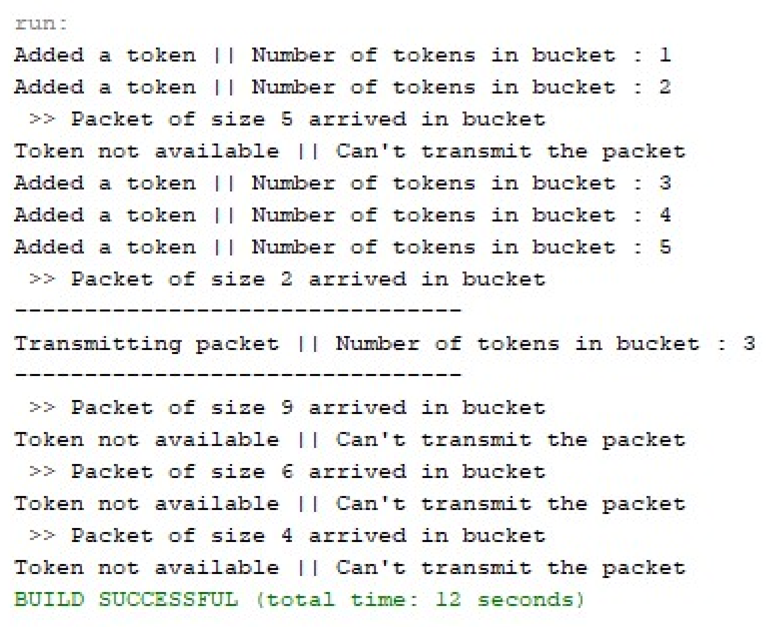
```
run:
Enter input :
5
Enter bucket size :
10
Enter incomming packet size :
3
Enter leaking packet size :
2
***************************************************************************************
Incomming packet with size : 3
Bucket buffer size : 3
After outgoing(leakage) : 1 packets left out of 10 in buffer(bucket)
********************************************************************************************
Incomming packet with size : 3
Bucket buffer size : 4
After outgoing(leakage) : 2 packets left out of 10 in buffer(bucket)
********************************************************************************************
Incomming packet with size : 3
Bucket buffer size : 5
After outgoing(leakage) : 3 packets left out of 10 in buffer(bucket)
********************************************************************************************
Incomming packet with size : 3
Bucket buffer size : 6
After outgoing(leakage) : 4 packets left out of 10 in buffer(bucket)
********************************************************************************************
Incomming packet with size : 3
Bucket buffer size : 7
After outgoing(leakage) : 5 packets left out of 10 in buffer(bucket)
********************************************************************************************
BUILD SUCCESSFUL (total time: 9 seconds)
```

TOKEN BUCKET CODE -

```
/**
 *
 * @author Atharva Paliwal
 */

class Bucket{

    public int tokens, maxsize;

    Bucket(int max){
            tokens = 0;
            maxsize = max;
    }

    synchronized void addToken(int n)
    // adding to the quantity of tokens that are generated after time t
    {
            if(tokens >= maxsize) return;
            tokens = tokens + 1;
            System.out.println("Added a token || Number of tokens in bucket :
" + tokens);
    }

    synchronized void addPacket(int n){
            System.out.println(" >> Packet of size " + n + " arrived in
bucket");
            if(n > tokens){ // n : number of packet
                System.out.println("Token not available || Can't transmit the
packet");
            }
            else{
                    tokens = tokens - n;
//while transmitting the packet the host capture and destroy one token
System.out.println("--------------------------------");
System.out.println("Transmitting packet || Number of tokens in bucket : " +
tokens);
                    System.out.println("--------------------------------");
            }
    }


}


class generateToken extends Thread{
      Bucket b;
        int count = 0; //No. of tokens
      generateToken(Bucket b){
            this.b = b;
      }
        @Override
      public void run(){
            while(count != 5)
//Here we have set limit to 5 to avoid infinite loop for demonstration
            {
                b.addToken(1);
                try
                {
                    Thread.sleep(1000);
                } catch(Exception e){}


                count ++;
            }
        }
    }

    class generatePacket extends Thread{
        Bucket b;
          int count = 0; //No of packets
        generatePacket(Bucket b){
            this.b = b;
        }

          @Override
        public void run(){
            while(count != 5)
            //Here we have set limit to 5 to avoid infinite loop for
demonstration
            {
                try
                {
                    Thread.sleep(500 + (int) (Math.random()*3000));
                //Setting the random time of arrival
                }
                catch(Exception e){}
                b.addPacket(1 + (int) (Math.random()*9));
                //Random generation of size of packet
                count++;
            }
        }
    }
```

```
class TokenBucket{
      public static void main(String args[]){
            Bucket b = new Bucket(10);
            Thread tokens = new generateToken(b);
            Thread packets = new generatePacket(b);
            try{
                    tokens.start();
                    packets.start();
            }
            catch(Exception e){}


      }
}
```

OUTPUT-



```
run:
Added a token || Number of tokens in bucket : 1
Added a token || Number of tokens in bucket : 2
 >> Packet of size 5 arrived in bucket
Token not available || Can't transmit the packet
Added a token || Number of tokens in bucket : 3
Added a token || Number of tokens in bucket : 4
Added a token || Number of tokens in bucket : 5
 >> Packet of size 2 arrived in bucket
--------------------------------
Transmitting packet || Number of tokens in bucket : 3
--------------------------------
 >> Packet of size 9 arrived in bucket
Token not available || Can't transmit the packet
 >> Packet of size 6 arrived in bucket
Token not available || Can't transmit the packet
 >> Packet of size 4 arrived in bucket
Token not available || Can't transmit the packet
BUILD SUCCESSFUL (total time: 12 seconds)
```