LEAKY BUCKET ALGORITHM:

ALGORITHM

1. Initialize the bucket_size and output_size.
2. Input the values of bucket_size and output_size along with the simulation time (here) to keep track of the Algorithm working.
3. Take any random packets as incoming packets as the incoming packets have different sizes.
4. When packet size is greater than the bucket_size; discard or drop the extra packets and send the remaining to the bucket or buffer.
5. The output_size of packets will be sent out and the other will remain in the bucket.
6. The next incoming packet will be received according to the size of the packet
   left in the bucket.
7. Repeat the steps 4 and 5 until the packets left is less than the size of the bucket.
8. Send all the remaining packets as output.
9. Exit

SIMULATION IN JAVA

//File Name: LeakyBucket.java

```
import java.util.*;
class LeakyBucket{
   static Scanner s=new Scanner(System.in);
   static int i=0,buf_size=0,output_size=0,no_of_sec=0,bucket_size=0;
   static int[] packet_array=new int[100];
   public static void main(String [] args) {
      System.out.println("Enter the size of the bucket: ");
      bucket_size=s.nextInt();
      System.out.println("Enter the output size of the bucket: ");
      output_size=s.nextInt();
      System.out.println("Enter the simulation time (in sec): ");
      no_of_sec=s.nextInt();
      int packet_remaining=0,drop=0,min=0;
      Random r=new Random();
      for(i=0;i<no_of_sec;i++)
         packet_array[i]=((r.nextInt(9)+1)*10);
```

```
    for(i=0;i<no_of_sec;i++){
      packet_remaining+=packet_array[i];
      if(packet_remaining>bucket_size){
        drop=packet_remaining-bucket_size;
        packet_remaining=bucket_size;
        System.out.println("Number of seconds: "+ (i+1));
        System.out.println("Packets received: " + packet_array[i]);
        min=Math.min(packet_remaining, output_size);
        System.out.println("Packets sent to the buffer: " + min);
        packet_remaining-=min;
        System.out.println("Packets left: " + packet_remaining);
        System.out.println("Packets dropped: " + drop + "\n");
        drop=0;
      }
    }
    System.out.println("Simulating beyond the time: ");
    while(packet_remaining!=0){
      if(packet_remaining>bucket_size){
        drop=packet_remaining-bucket_size;
        packet_remaining=bucket_size;
      }
      min=Math.min(packet_remaining,output_size);
      System.out.println("Packets received: " + packet_remaining);
      System.out.println("Packets left: "+ min + "\n");
      packet_remaining-=min;
      System.out.println("Packets left: " + packet_remaining);
      System.out.println("Packets dropped: " + drop + "\n");
      drop=0;
    }
  }
}
```

DESCRIPTION

s → object for Scanner class
bucket_size → stores the size of the bucket
output_size → stores the output rate of data from bucket
no_of_sec → stores the number of seconds the simulation has to be done
packet_array → stores the packets received to the bucket
drop → stores the value of the dropped packets

min → stores the minimum value of the two values sent for comparision

Random → Function generates random values in the given bounds
r → object of "Random" class
r.nextInt(9)+1)*10 → (generates random numbers within the range (0-9))*10
Math → class for mathematical operations
min(a,b) → gives the minimum of a and b; method of Math class
More clarity with the below sample data!

REFERENCES AND RECOMMENDATIONS:

1. Transmission Control Protocol
2. Network Performance
3. Internet Backbone
4. Bandwidth and Speed