Expt. 14 – 21/12/2024

21/12

Expt. 14:

Aim: Congestion Control using leaky Bucket Algorithm.

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#define NOF_PACKETS 5
int rand(int a) {
    int rn = (random()·1·10)·1·a;
    return rn == 0?1:rn;
}

long int random(void);
int main() {
    int packt_sy[NOF_PACKETS], i·clk, b_size, o_rate,
        p_size_rm = 0, p_sy, p_time, op;
    for (i=0; i<NOF_PACKETS; i++)
        packet_sy[i] = random()·1·100;
    for (i=0; i<NOF_PACKETS; i++)
        printf("\n packet [·1·d]·:·1·d bytes \t", i,
               packet_sy[i]);
    printf("\n Enter the output rate");
    scanf("·1·d, &_arate);
    printf("enter bucket size:");
    scanf("·1·d", &b_size);
    for(i=0; i<NOF_PACKETS; i++);
        if(packet_sy[i] + p_sy_rml > bsize)
            if(packt_sy[i] > bsize)
                printf("Incoming packet size
```

04.01.25

```c
(·1·d bytes) is greater than Bucket capacity
(·1·d bytes) – PACKET- REJECTED packet_sy[i] bsize)
else
    printf("\n\n Bucket capacity exceeds - d_PACKETS
           REJECTED !!");
else {
    p_sy_rmt = packet_sy[i];
    printf("\n\n Incoming packet cant ·1·d",
           packet_sy[i]);
    printf("\n Bytes remaining to transmit: ·1·d",
           p_sy_rm);
    while(s_sy_rm>0) {
        sleep(1);
        if(p_sy_rm);
            if(p_sy_rm <= o_rate)
                op = p_sy_rm, p = sy_rm = 0
        else
            op = o_rate, p_sy_rm-o-r ct,
            printf(Packet size, op);
    else
        printf(" No packets to transmit);
    }
}
}
```
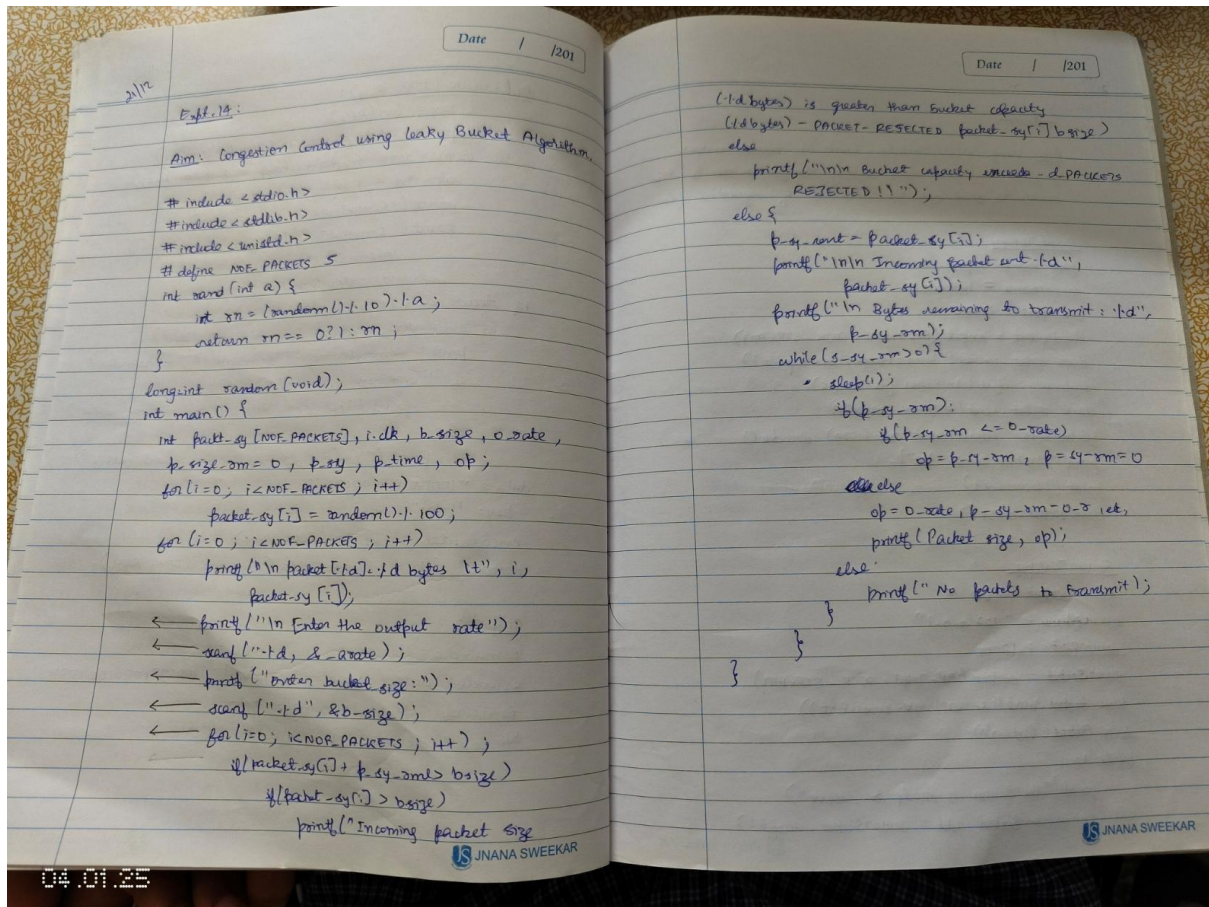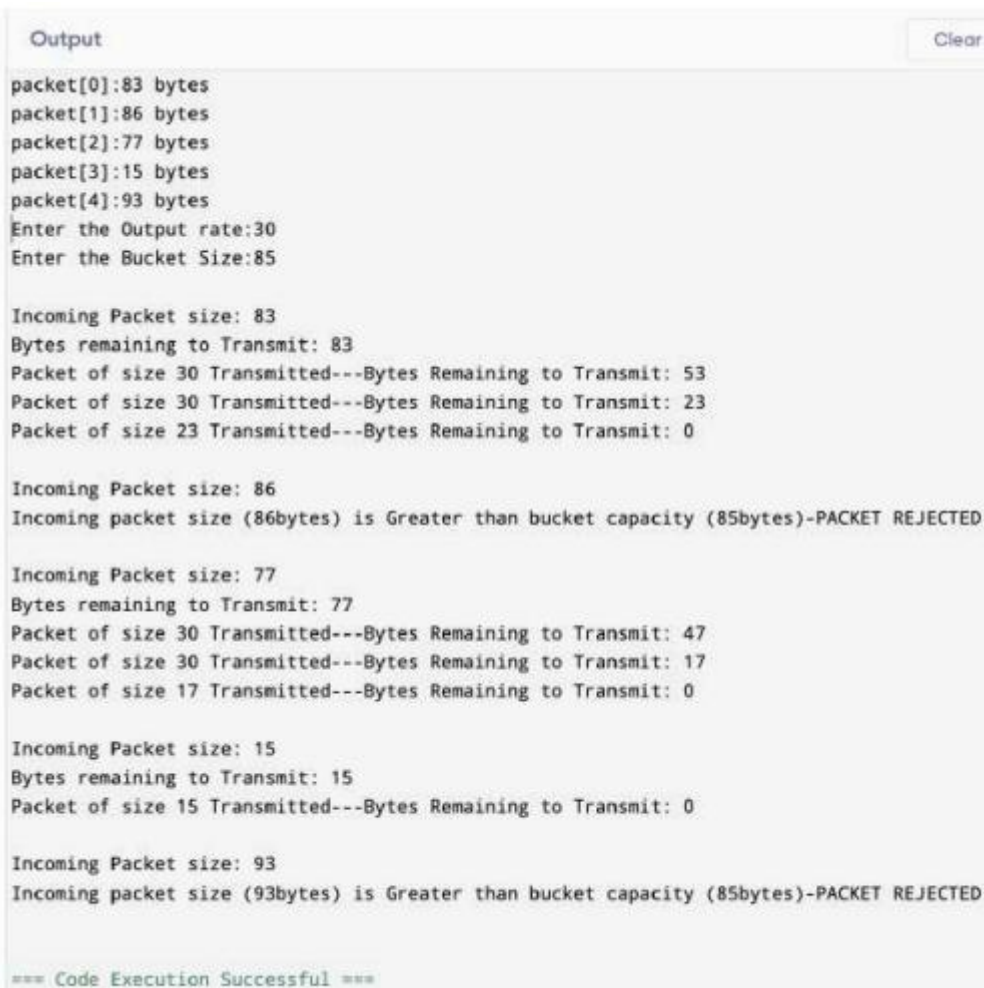
Code:

```
#include <stdio.h>

#include <stdlib.h>

#include <unistd.h> // for sleep function

#define NOF_PACKETS 5

// Function to simulate sending packets

void send_packet(int packet_size, int output_rate) {

while (packet_size > 0) {

int sent = (packet_size < output_rate) ? packet_size : output_rate;

printf("Packet of size %d Transmitted---", sent);

packet_size -= sent;

printf("Bytes Remaining to Transmit: %d\n", packet_size);

sleep(1); // Simulate time delay between packets

}

}

int main() {

int output_rate, bucket_size, incoming_packet_size;

int i, packet_size[NOF_PACKETS];

// Input number of packets and their sizes

for(i = 0; i < NOF_PACKETS; i++) {

packet_size[i] = rand() % 100; // Random packet size between 0 and 99

printf("packet[%d]:%d bytes\n", i, packet_size[i]);

}

printf("Enter the Output rate:");

scanf("%d", &output_rate);

printf("Enter the Bucket Size:");

scanf("%d", &bucket_size);

for(i = 0; i < NOF_PACKETS; i++) {

printf("\nIncoming Packet size: %d\n", packet_size[i]);

if(packet_size[i] > bucket_size) {
```

```c
        printf("Incoming packet size (%dbytes) is Greater than bucket capacity
(%dbytes)-PACKET REJECTED\n", packet_size[i], bucket_size);
55
        continue;
    }
    printf("Bytes remaining to Transmit: %d\n", packet_size[i]);
    send_packet(packet_size[i], output_rate);
    }
    return 0;
}
```

Output:

```
Output                                                          Clear

packet[0]:83 bytes
packet[1]:86 bytes
packet[2]:77 bytes
packet[3]:15 bytes
packet[4]:93 bytes
Enter the Output rate:30
Enter the Bucket Size:85

Incoming Packet size: 83
Bytes remaining to Transmit: 83
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 53
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 23
Packet of size 23 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 86
Incoming packet size (86bytes) is Greater than bucket capacity (85bytes)-PACKET REJECTED

Incoming Packet size: 77
Bytes remaining to Transmit: 77
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 47
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 17
Packet of size 17 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 15
Bytes remaining to Transmit: 15
Packet of size 15 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 93
Incoming packet size (93bytes) is Greater than bucket capacity (85bytes)-PACKET REJECTED

=== Code Execution Successful ===
```