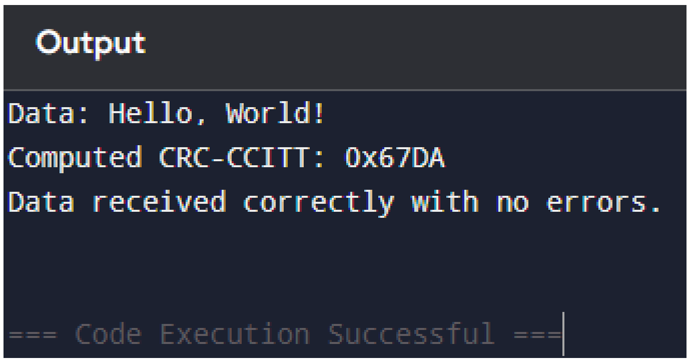
LAB-09

1.Write a program for error detecting code using CRC-CCITT.

Code:

```
#include <stdio.h>
#include <stdint.h>
#define CRC_POLY 0x11021
#define INITIAL_CRC 0xFFFF
uint16_t compute_crc(uint8_t *data, size_t length) {
    uint16_t crc = INITIAL_CRC;
    for (size_t i = 0; i < length; i++) {
        crc ^= (data[i] << 8);
        for (int j = 0; j < 8; j++) {
            if (crc & 0x8000) {
                crc = (crc << 1) ^ CRC_POLY;
            } else {
                crc <<= 1;
            }
        }
    }
    return crc & 0xFFFF;
}
int check_crc(uint8_t *data, size_t length, uint16_t expected_crc) {
    uint16_t computed_crc = compute_crc(data, length);
    return (computed_crc == expected_crc);
}
int main() {
    uint8_t data[] = "Hello, World!";
    size_t data_length = sizeof(data) - 1;
    printf("Data: %s\n", data);
    uint16_t crc = compute_crc(data, data_length);
    printf("Computed CRC-CCITT: 0x%04X\n", crc);
    uint8_t received_data[] = "Hello, World!";
    size_t received_length = sizeof(received_data) - 1;
    if (check_crc(received_data, received_length, crc)) {
        printf("Data received correctly with no errors.\n");
    } else {
        printf("Error detected in received data!\n");
    }
    return 0;
}
```

Output:

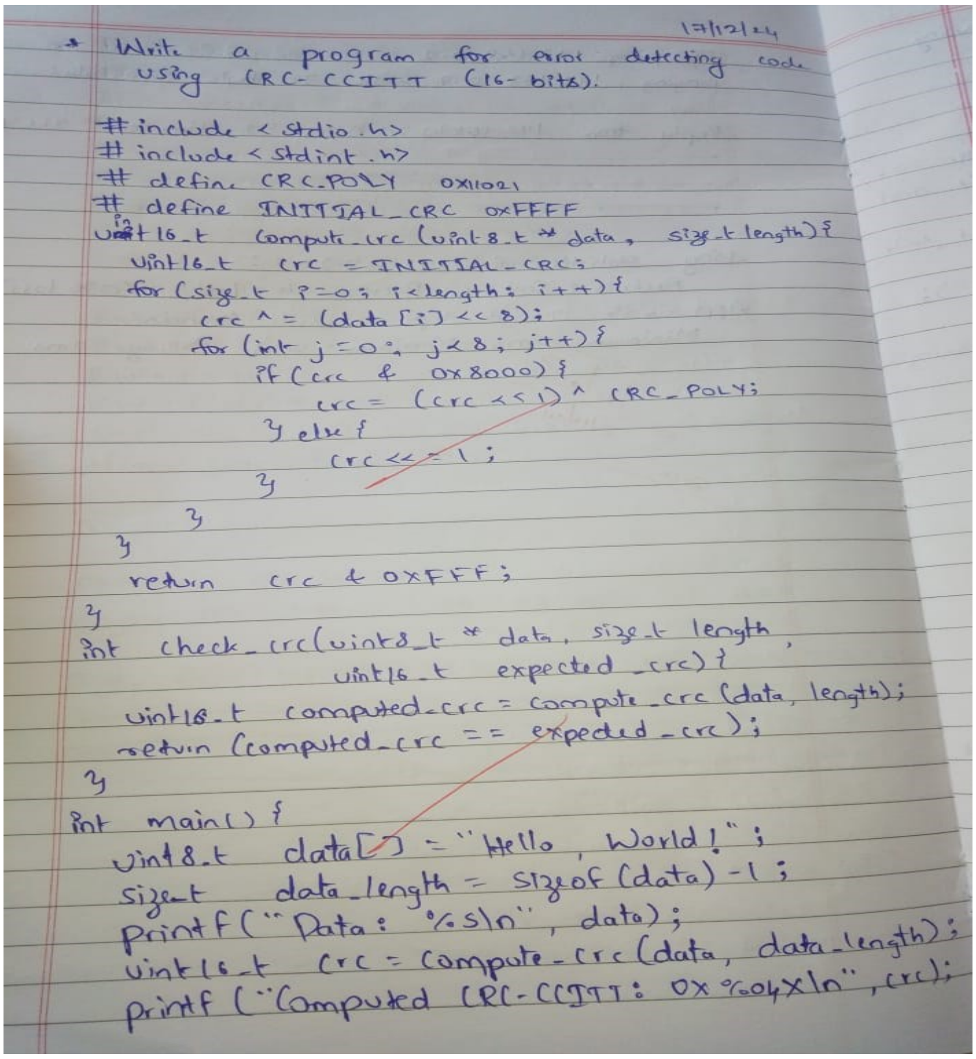


Output

```
Data: Hello, World!
Computed CRC-CCITT: 0x67DA
Data received correctly with no errors.


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

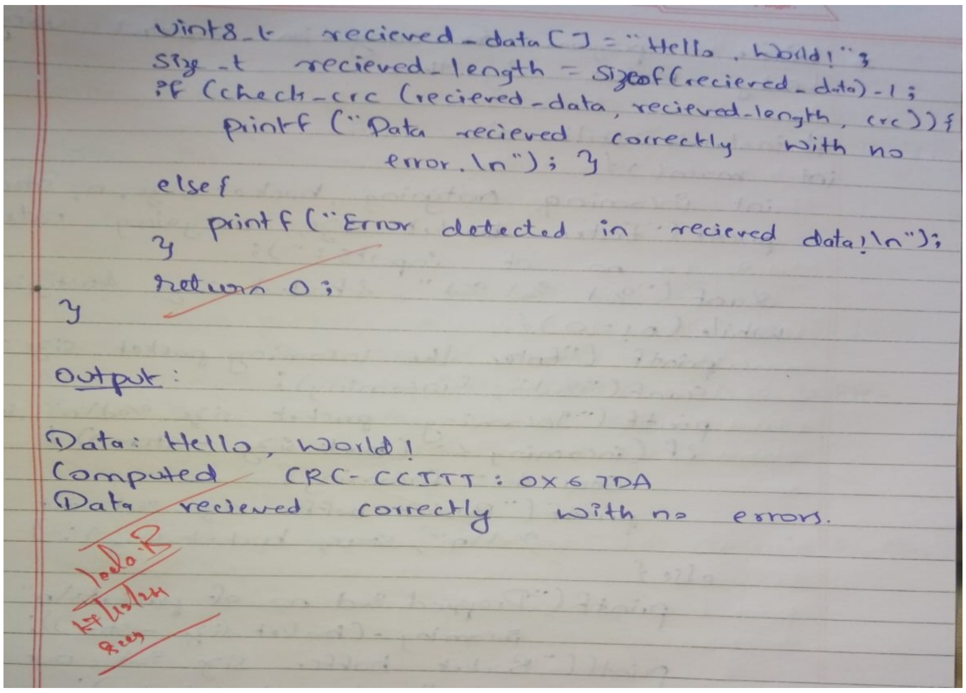
Observation:



17/12/24

* Write a program for error detecting code using CRC-CCITT (16-bits).

```
#include <stdio.h>
#include <stdint.h>
#define CRC_POLY 0x1021
#define INITIAL_CRC 0xFFFF
uint16_t compute_crc(uint8_t *data, size_t length){
  uint16_t crc = INITIAL_CRC;
  for(size_t i=0; i<length; i++){
      crc ^= (data[i] << 8);
      for(int j=0; j<8; j++){
          if(crc & 0x8000){
              crc = (crc << 1) ^ CRC_POLY;
          } else {
              crc <<= 1;
          }
      }
  }
  return crc & 0xFFFF;
}
int check_crc(uint8_t *data, size_t length,
                  uint16_t expected_crc){
  uint16_t computed_crc = compute_crc(data, length);
  return (computed_crc == expected_crc);
}
int main(){
  uint8_t data[] = "Hello, World!";
  size_t data_length = sizeof(data) - 1;
  printf("Data: %s\n", data);
  uint16_t crc = compute_crc(data, data_length);
  printf("Computed CRC-CCITT: 0x %04X\n", crc);
```



```
  uint8_t recieved_data[] = "Hello, World!";
  size_t recieved_length = sizeof(recieved_data) - 1;
  if (check_crc(recieved_data, recieved_length, crc)){
        printf("Data recieved correctly with no
                    error.\n"); }
  else{
        printf("Error detected in recieved data!\n");
  }
  return 0;
}
```

Output:

```
Data: Hello, World!
Computed CRC-CCITT: 0x67DA
Data recieved correctly with no errors.
```

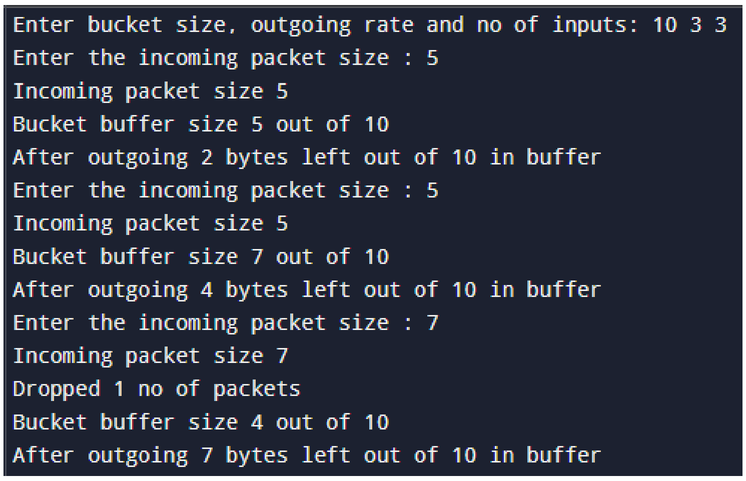
[illegible] 17/12/24

2. Write a program for congestion control using Leaky bucket algorithm.

Code:

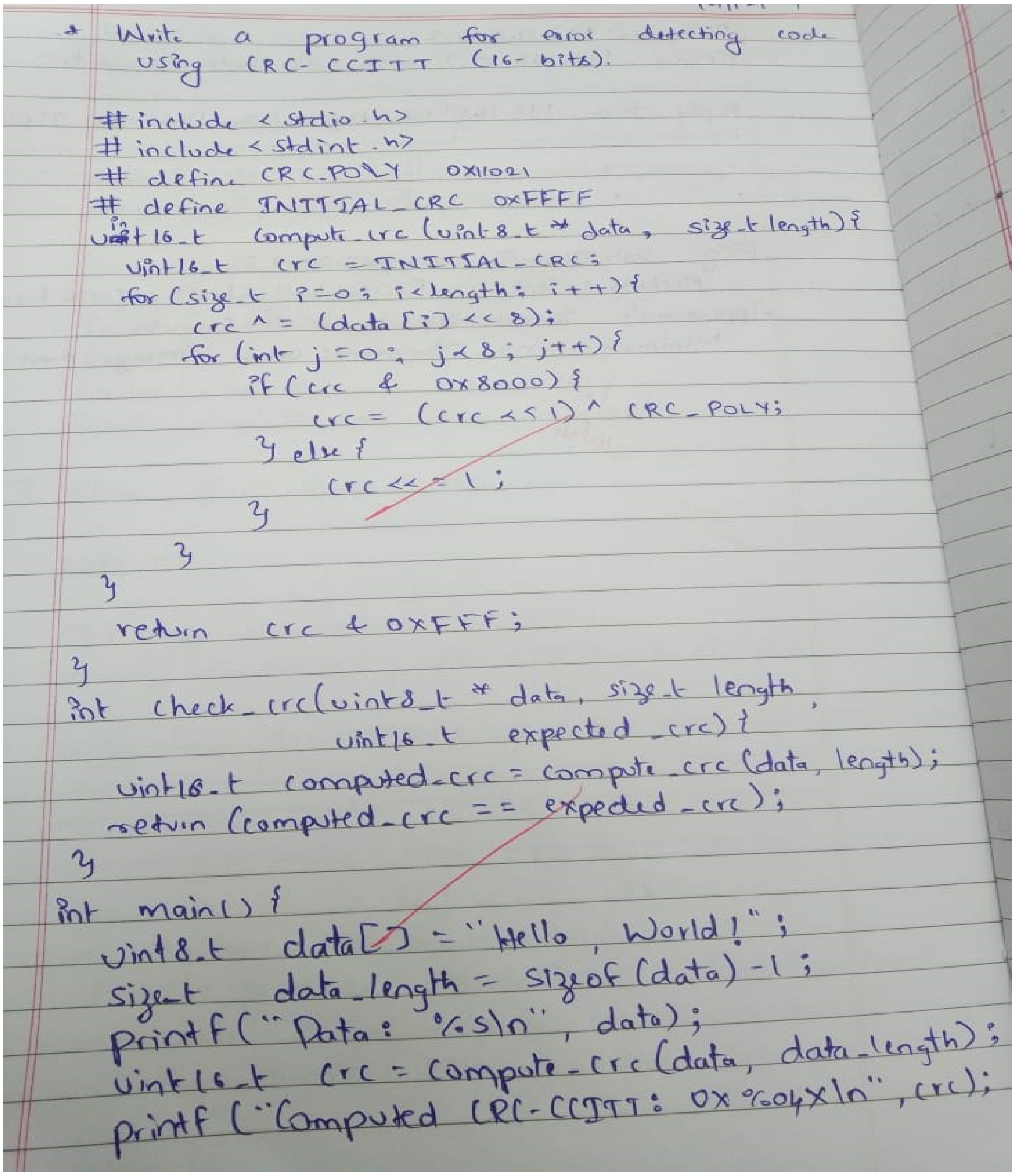
```c
#include<stdio.h>
int main(){
    int incoming, outgoing, buck_size, n, store = 0;
    printf("Enter bucket size, outgoing rate and no of inputs: ");
    scanf("%d %d %d", &buck_size, &outgoing, &n);
    while (n != 0) {
        printf("Enter the incoming packet size : ");
        scanf("%d", &incoming);
        printf("Incoming packet size %d\n", incoming);
        if (incoming <= (buck_size - store)){
            store += incoming;
            printf("Bucket buffer size %d out of %d\n", store, buck_size);
        } else {
            printf("Dropped %d no of packets\n", incoming - (buck_size - store));
            printf("Bucket buffer size %d out of %d\n", store, buck_size);
            store = buck_size;
        }
        store = store - outgoing;
        printf("After outgoing %d bytes left out of %d in buffer\n", store, buck_size);
        n--;
    }
}
```

Output:



```
Enter bucket size, outgoing rate and no of inputs: 10 3 3
Enter the incoming packet size : 5
Incoming packet size 5
Bucket buffer size 5 out of 10
After outgoing 2 bytes left out of 10 in buffer
Enter the incoming packet size : 5
Incoming packet size 5
Bucket buffer size 7 out of 10
After outgoing 4 bytes left out of 10 in buffer
Enter the incoming packet size : 7
Incoming packet size 7
Dropped 1 no of packets
Bucket buffer size 4 out of 10
After outgoing 7 bytes left out of 10 in buffer
```

Observation:



* Write a program for error detecting code using CRC- CCITT (16- bits).

```
#include <stdio.h>
#include <stdint.h>
#define CRC_POLY   0X11021
#define INITIAL_CRC  0xFFFF
uint16_t  compute_crc (uint8_t * data,  size_t length){
  uint16_t  crc  = INITIAL_CRC;
  for (size_t  i=0; i<length; i++){
      crc ^= (data [i] << 8);
      for (int j=0; j<8; j++){
          if (crc  &  0x8000){
              crc = (crc << 1) ^ CRC_POLY;
          } else {
              crc <<= 1;
          }
      }
  }
  return  crc & 0xFFF;
}
int  check_crc(uint8_t * data, size_t length,
                uint16_t  expected_crc){
  uint16_t  computed_crc = compute_crc (data, length);
  return (computed_crc == expected_crc);
}
int  main(){
  uint8_t  data[] = "Hello , World!";
  size_t  data_length = sizeof (data) -1;
  printf(" Data: %s\n", data);
  uint16_t  crc = compute_crc(data, data_length);
  printf ("Computed CRC-CCITT: 0x %04X\n", crc);
```

```
    uint8_t  recieved_data[] = "Hello, World!";
    size_t  recieved_length = sizeof(recieved_data) - 1;
    if (check_crc (recieved_data, recieved_length, crc)){
        printf ("Data recieved correctly with no
                    error.\n"); }
    else{
        printf("Error detected in recieved data!\n");
    }
    return 0;
}
```

Output:

```
Data: Hello, World!
Computed   CRC-CCITT : 0X 67DA
Data recieved correctly with no errors.
```

[illegible]