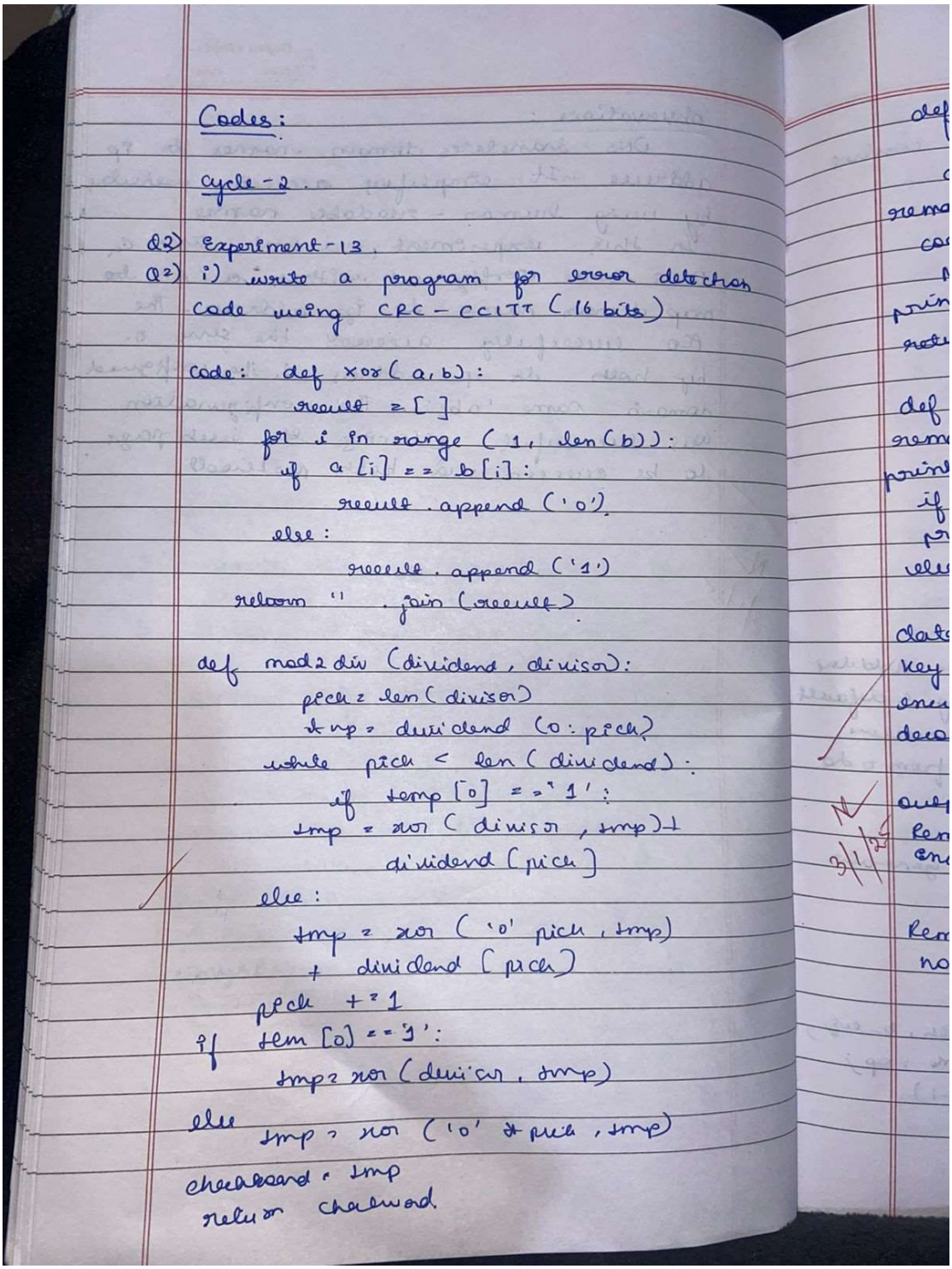
CYCLE-2

Avani.A (1BM22CS059)

EXPERIMENT-13

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

Code and Output:



Codes:

cycle-2.

Q2) Experiment-13.

Q2) i) write a program for error detection code using CRC-CCITT (16 bits)

```
Code: def xor(a,b):
        result = []
        for i in range(1, len(b)):
            if a[i] == b[i]:
                result.append('0')
            else:
                result.append('1')
        return ''.join(result)

def mod2div(dividend, divisor):
    pick = len(divisor)
    tmp = dividend[0:pick]
    while pick < len(dividend):
        if tmp[0] == '1':
            tmp = xor(divisor, tmp)+
                dividend[pick]
        else:
            tmp = xor('0'*pick, tmp)
            + dividend[pick]
        pick += 1
    if tem[0] == '1':
        tmp = xor(divisor, tmp)
    else
        tmp = xor('0'*pick, tmp)
    checkword = tmp
    return checkword
```

Bafna Gold
Date:

```
def encode (data, key):
    l.key = len(key)
    appended-data = data + '0' + (l.key - 1)
remainder = mod2div (append-data, key)
    codeword = data + remainder
    print ("Remainder:", remainder)
print("encoded data [data + remainder codeword)
return codeword

def decode data (encoded-data, key)
remainder = mod2 div (encoded - data, key)
print ("Remainder after decoding:", remainder)
    if '1' not in remainder
    print ("No error detected in recived data")
else print ("Error " detected in received data")

data = "100100100010010"
key = "1101"
encoded-data = encodedData (data, key)
decoded-data = encodeData (data)
```

output:
Remainder = 111
Encoded-Data (data + Remainder) =
100100100010010111
Remainder after decoding = 000
no error detected in received data

3/1/25

Codes:

code:

```
#include <stdio.h>
#include <string.h>
#define N strlen(gen_poly)

int data[28],check[28],gen_poly[10];
int data_len,i,j;

void XOR(){
   for(j = 0; j<N;j++){
      if(check[j] == gen_poly[j]){
         check[j] = '0';
      } else{
         check[j] = '1';
      }
   }
}

void crc(){
   for(i = 0; i<N;i++){
      check[i] = data[i];
   }
   do{
      if(check[j] == '1'){
         XOR();
      }
      for(j = 0; j<N-1; i++){
         check[j] = check[j+1];
      }
      check[j] = data[i++];
   }
   while(i <= data_len + N - 1);
}

void reciever(){
   printf("\nData recieved: ");
   scanf("%s",data);

   crc();
   for(i = 0; i<N-1;i++){
      if (check[i] == '1'){
         break;
      }
   if (i < N-1){
      printf("\nERROR!");
      }else{
         printf("\nNO ERROR!");
      }
```

```
    }
}

int main(){
    printf("\nEnter data:");
    scanf("%s",data);
    printf("\nEnter generator:");
    scanf("%s",gen_poly);
    data_len = strlen(data);
    for(i = 0; i<data_len+N-1;i++){
        data[i] = '0';
    }
    printf("\nData with padded 0's: %s",data);
    crc();
    printf("\nCheck sum: %s",check);

    for(i = data_len; i<data_len+N-1; i++){
        data[i] = check[i-data_len];
    }
    reciever();
}
```

Output: