BCSE308 P -COMPUTER NETWORKS LAB

Register Number	24BCE1282	Subject Code	BCSE308P
Name of the Student	Kunsh Jain	Subject Name	Computer Networks Lab
Programme	B.Tech		
Date	23/7/25	Exp No	4

Aim :

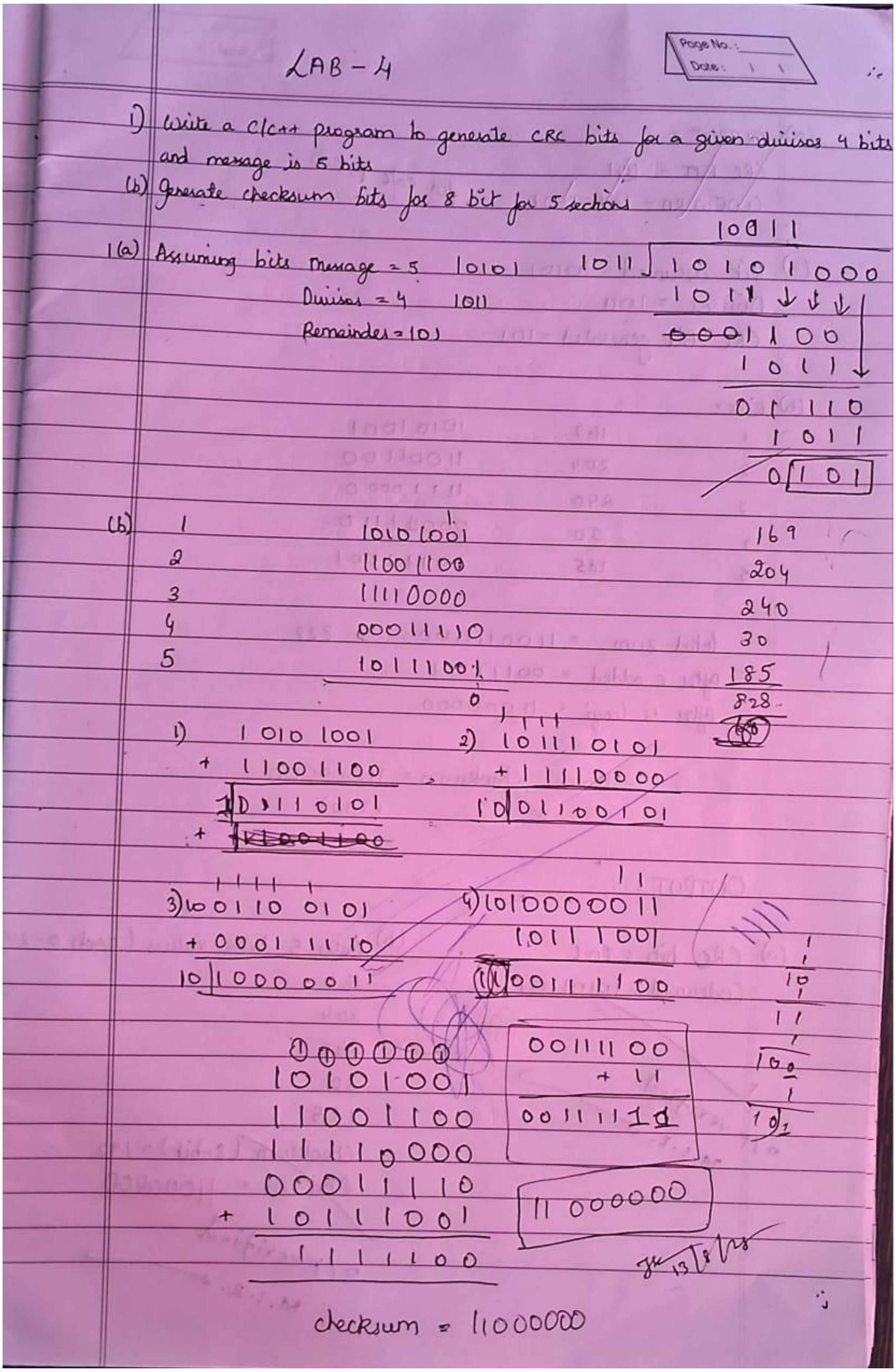
(a) To write a c/cpp program to generate CRC lists for the given divisor is 4 bits+message is 5 bits.

(b) Generate checksum bits of 8 bits for 5 sections.

Procedure:

1. Write the formula for the assumed calculations
2. Do dry calculations on your notebook
3. Check the output with the verified output
4. Write the c/cpp code for (a) and (b)
5. Get the ouput verified

Attestation:



LAB - 4

1) Write a C/C++ program to generate CRC bits for a given divisor 4 bits and message is 5 bits

(b) Generate checksum bits for 8 bit for 5 sections

1(a) Assuming bits message = 5 10101

Divisor = 4 1011

Remainder = 101

```
              10011
1011 )  1 0 1 0 1 0 0 0
        1 0 1 1 ↓ ↓ ↓
        0 0 0 1 1 0 0
              1 0 1 1 ↓
              0 1 1 1 0
                1 0 1 1
              0 [1 0 1]
```

(b)

1	10101001	169
2	11001100	204
3	11110000	240
4	00011110	30
5	10111001	185
		828

```
1)   1010 1001            2)   101110101
   + 11001100                + 11110000
   1|01110101                10|01100101
   + ~~11001100~~

3)  100110 0101           4)  1010000011
  + 0001 1110                  10111001
   10|10000011              (1)(1)|00111100
```

```
    ① ① ① ① ① ①
    1 0 1 0 1 0 0 1
    1 1 0 0 1 1 0 0
    1 1 1 1 0 0 0 0
    0 0 0 1 1 1 1 0
  + 1 0 1 1 1 0 0 1
    1 1 1 1 1 1 0 0
```

```
  00111100
  +     11
  00111111
```

```
  1
  1
  10
  1
  11
  1
  100
  1
  101₂
```

$\boxed{11000000}$

[illegible] 13/8/[illegible]

checksum = 11000000

Page No.: ______
Date: / /

(a) ~~OUTPUT:~~
~~CRC BIT - 101~~
~~CODEWORD - 10101101~~ for calc P.T.O

(a) Bit assumed = 10101
Divisor = 1011
CRC Bit generated = 101

(b) Bits:

1	169	10101001
2	204	11001100
3	240	11110000
4	30	00011110
5	185	10111001

total sum = 1100111100 = 828
After c added = 00111111
After 1's Comp = 11000000

Hence Checksum = 11000000
(192)

OUTPUT:

(a) CRC bits : 101
Codeword : 10101101

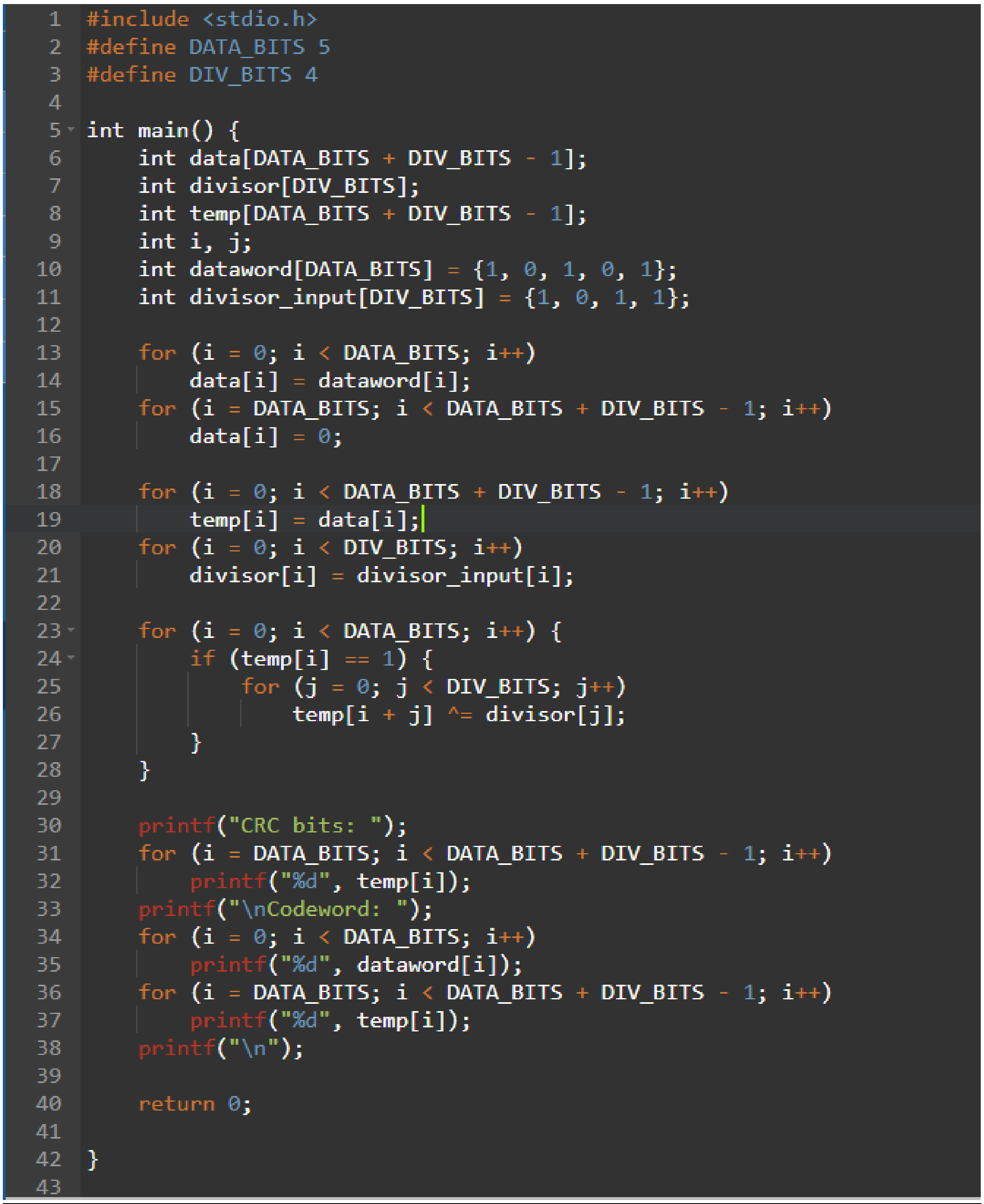
o/p verified
M.S.R.

(b) Enter 5 data values (each 0-255)
169
204
240
30
185
Checksum (8-bit) = 192
8 bit res = 11000000

o/p verified
M.S.R.

Code:

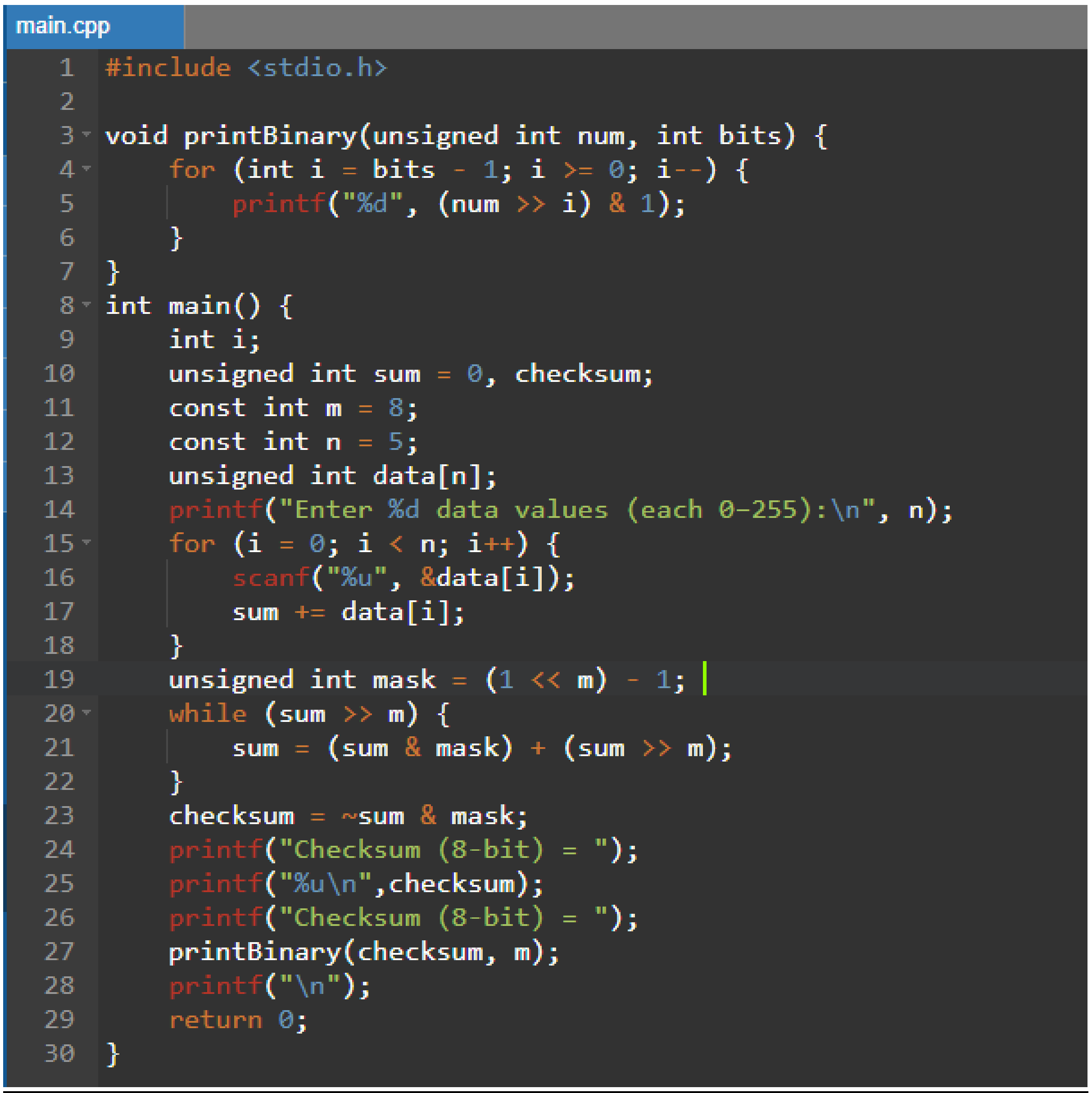
(a)



```c
#include <stdio.h>
#define DATA_BITS 5
#define DIV_BITS 4

int main() {
    int data[DATA_BITS + DIV_BITS - 1];
    int divisor[DIV_BITS];
    int temp[DATA_BITS + DIV_BITS - 1];
    int i, j;
    int dataword[DATA_BITS] = {1, 0, 1, 0, 1};
    int divisor_input[DIV_BITS] = {1, 0, 1, 1};

    for (i = 0; i < DATA_BITS; i++)
        data[i] = dataword[i];
    for (i = DATA_BITS; i < DATA_BITS + DIV_BITS - 1; i++)
        data[i] = 0;

    for (i = 0; i < DATA_BITS + DIV_BITS - 1; i++)
        temp[i] = data[i];
    for (i = 0; i < DIV_BITS; i++)
        divisor[i] = divisor_input[i];

    for (i = 0; i < DATA_BITS; i++) {
        if (temp[i] == 1) {
            for (j = 0; j < DIV_BITS; j++)
                temp[i + j] ^= divisor[j];
        }
    }

    printf("CRC bits: ");
    for (i = DATA_BITS; i < DATA_BITS + DIV_BITS - 1; i++)
        printf("%d", temp[i]);
    printf("\nCodeword: ");
    for (i = 0; i < DATA_BITS; i++)
        printf("%d", dataword[i]);
    for (i = DATA_BITS; i < DATA_BITS + DIV_BITS - 1; i++)
        printf("%d", temp[i]);
    printf("\n");

    return 0;

}
```

```c
#include <stdio.h>

#define DATA_BITS 5

#define DIV_BITS 4



int main() {

    int data[DATA_BITS + DIV_BITS - 1];

    int divisor[DIV_BITS];
    int temp[DATA_BITS + DIV_BITS - 1];

    int i, j;

    int dataword[DATA_BITS] = {1, 0, 1, 0, 1};

    int divisor_input[DIV_BITS] = {1, 0, 1, 1};


    for (i = 0; i < DATA_BITS; i++)

        data[i] = dataword[i];

    for (i = DATA_BITS; i < DATA_BITS + DIV_BITS - 1; i++)

        data[i] = 0;


    for (i = 0; i < DATA_BITS + DIV_BITS - 1; i++)

        temp[i] = data[i];

    for (i = 0; i < DIV_BITS; i++)

        divisor[i] = divisor_input[i];


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

        if (temp[i] == 1) {

            for (j = 0; j < DIV_BITS; j++)

                temp[i + j] ^= divisor[j];

        }

    }


    printf("CRC bits: ");

    for (i = DATA_BITS; i < DATA_BITS + DIV_BITS - 1; i++)

        printf("%d", temp[i]);

    printf("\nCodeword: ");

    for (i = 0; i < DATA_BITS; i++)

        printf("%d", dataword[i]);

    for (i = DATA_BITS; i < DATA_BITS + DIV_BITS - 1; i++)

        printf("%d", temp[i]);

    printf("\n");


    return 0;


}
```

(b)



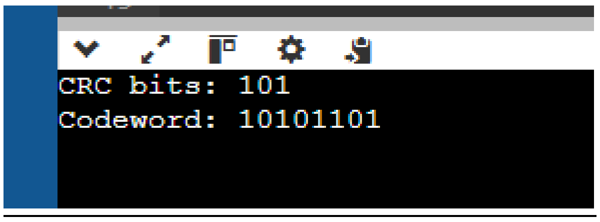
```
#include <stdio.h>

void printBinary(unsigned int num, int bits) {
    for (int i = bits - 1; i >= 0; i--) {
        printf("%d", (num >> i) & 1);
    }
}
int main() {
    int i;
    unsigned int sum = 0, checksum;
    const int m = 8;
    const int n = 5;
    unsigned int data[n];
    printf("Enter %d data values (each 0-255):\n", n);
    for (i = 0; i < n; i++) {
        scanf("%u", &data[i]);
        sum += data[i];
    }
    unsigned int mask = (1 << m) - 1;
    while (sum >> m) {
        sum = (sum & mask) + (sum >> m);
    }
    checksum = ~sum & mask;
    printf("Checksum (8-bit) = ");
    printf("%u\n",checksum);
    printf("Checksum (8-bit) = ");
    printBinary(checksum, m);
    printf("\n");
    return 0;
}
```

```
#include <stdio.h>


void printBinary(unsigned int num, int bits) {

  for (int i = bits - 1; i >= 0; i--) {

    printf("%d", (num >> i) & 1);

  }

}

int main() {

  int i;

  unsigned int sum = 0, checksum;
```

```
    const int m = 8;

    const int n = 5;

    unsigned int data[n];

    printf("Enter %d data values (each 0–255):\n",   n);

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

        scanf("%u", &data[i]);

        sum += data[i];

    }

    unsigned int mask = (1 << m) - 1;

    while (sum >> m) {

        sum = (sum & mask) + (sum >> m);

    }

    checksum = ~sum & mask;

    printf("Checksum (8-bit) = ");

    printf("%u\n",checksum);

    printf("Checksum (8-bit) = ");

    printBinary(checksum, m);

    printf("\n");

    return 0;

}
```

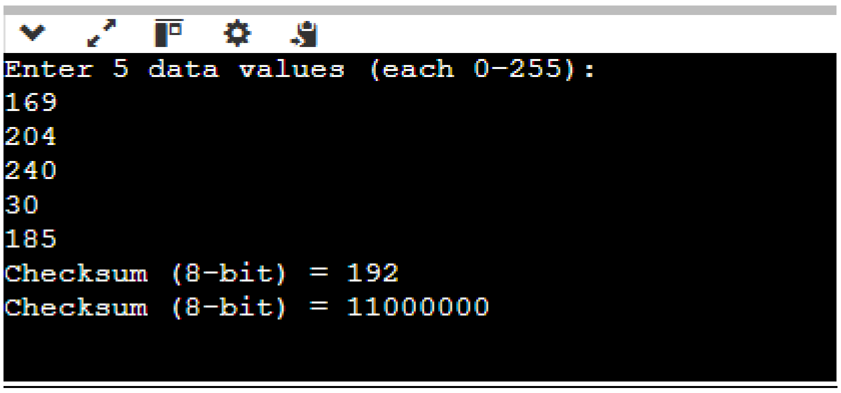
Output:

(a)



```
CRC bits: 101
Codeword: 10101101
```

(b)



```
Enter 5 data values (each 0-255):
169
204
240
30
185
Checksum (8-bit) = 192
Checksum (8-bit) = 11000000
```

Result:

The output of the code is verified and the following written code runs on the system c/cpp compiler.