RAMAIAH UNIVERSITY OF APPLIED SCIENCES

ASSIGNMENT

Course Code	19CSC303A
Course Name	Computer Networks
Programme	B. Tech
Department	Computer Science and Engineering
Faculty	Engineering and Technology

Name of the Student	Deepak R
Reg. No	18ETCS002041
Semester/Year	5th/2020
Course Leader/s	Dr. Rinki Sharma

Declaration Sheet			
Student Name	Deepak R		
Reg. No	18ETCS002041		
Programme	B. Tech	Semester/Year	5th/2020
Course Code	19CSC303A		
Course Title	Computer Networks		
Course Date		to	
Course Leader	Dr. Rinki Sharma		

Declaration

The assignment submitted herewith is a result of my own investigations and that I have conformed to the guidelines against plagiarism as laid out in the Student Handbook. All sections of the text and results, which have been obtained from other sources, are fully referenced. I understand that cheating and plagiarism constitute a breach of University regulations and will be dealt with accordingly.

Signature of the Student		Date	05/12/2020
Submission date stamp (by Examination & Assessment Section)			

Signature of the Course Leader and date	Signature of the Reviewer and date

Faculty of Engineering & Technology			
Ramaiah University of Applied Sciences			
Department	Computer Science and Engineering	Programme	B. Tech.
Semester	5th		
Course Code	CSC303A	Course Title	Computer Networks
Course Leader	Dr. Rinki Sharma		

Assignment - 1					
Register No.	18ETCS002041	Name of Student	Deepak R		
Sections		Marking Scheme	Max Marks	First Examiner Marks	Second Examiner Marks
Q.1	1.1	Disadvantages of the protocol	02		
	1.2	Modifications to overcome the disadvantages	03		
		Max Marks	05		
Q.2	2.1	Program to compute checksum at the transmitter	10		
	2.2	Program to check for error free data transmission at the receiver	10		
		Max Marks	20		
		Total Assignment Marks	25		

Course Marks Tabulation				
Component- 1(B) Assignment	First Examiner	Remarks	Second Examiner	Remarks
Q 1				
Q 2				
Marks (Max 25)				

Signature of First Examiner	Signature of
Second Examiner	

Please note:

1. **Documental evidence for all the components/parts of the assessment such as the reports, photographs, laboratory exam / tool tests are required to be attached to the assignment report in a proper order.**
2. **The First Examiner is required to mark the comments in RED ink and the Second Examiner's comments should be in GREEN ink.**
3. **If the variation between the marks awarded by the first examiner and the second examiner lies within +/- 3 marks, then the marks allotted by the first examiner is considered to be final. If the variation is more than +/- 3 marks then both the examiners should resolve the issue in consultation with the Chairman BoE.**

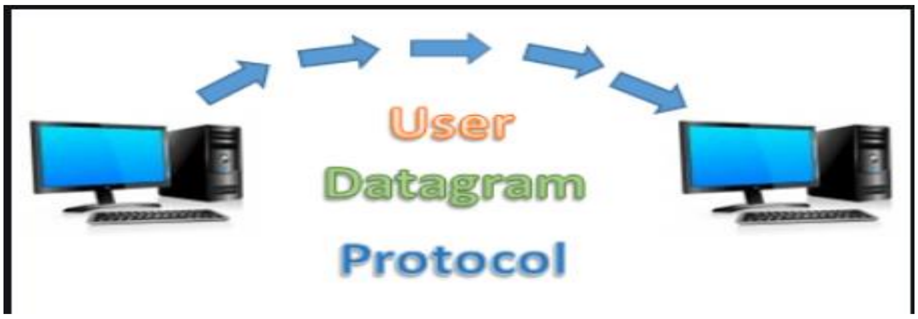
Assignment

Instructions to students:

1. **The assignment consists of 3 questions.**
2. **Maximum marks is 25.**
3. **The assignment has to be neatly word processed as per the prescribed format.**
4. **The maximum number of pages should be restricted to 9.**
5. **The printed assignment must be submitted to the course leader.**
6. **Submission Date: December 5th 2020**
7. **Submission after the due date is not permitted.**
8. **IMPORTANT: It is essential that all the sources used in preparation of the assignment must be suitably referenced in the text.**
9. **Marks will be awarded only to the sections and subsections clearly indicated as per the problem statement/exercise/question**

Solution for Question 1

Here a simple application-level protocol built on top of UDP that allows a client to retrieve a file from a remote server residing at a well-known address



Let consider a simple application-level protocol is build on the top of the UDP that allows a client to retrieve a file from a remote server exist in a well-known address. Here, UDP is User Datagram Protocol.The client first sends with a file name and the server responds with a sequence of data packets that contain the requested file.

To ensure reliability and sequenced delivery,here the client and server are using the stop-and-wait protocol.

Problems occur with this protocol:-

The given statement makes the possibility for a client system to get the wrong file.

Let us assume that there are two clients A and B available, and the client A sends a request for a file

"F1" and then it gets crashed.

The client B sends same protocol to request some other file "F2";

In this case, the client B is running on the same machine of where the client A is running and then both files F1 and F2 are using the same IP addresses.

The client B binds its User Datagram Protocol socket to the same port of the client A was using previously.

In case the client B's request is lost when the server's reply for the client A arrive.

And also, client B receives it and assumes that it is a reply for its own request.

Hence, the client system gets the wrong file, when using this approach.

Steps that can be taken to tackle above Problems are:

To ensure reliability and sequenced delivery, client and server use a stop-and-wait protocol.

After timeout timer expires, sender will assume that the data is lost but actually the acknowledgment is lost. By assuming this it will send the data packet again but according to receiver it is a new data packet, hence it will give rise to duplicate packet problem.

To eliminate duplicate packet problem sequence number is added to the data packet. So using packet numbers it can easily determine the duplicate packets.

According to sender, the acknowledgement of packet 1 is delayed and packet 2 has been lost. But the receiver assumes that the acknowledgement that has been received was of packet 2. This problem is called missing packet problem.
Missing packet problem can be solved if acknowledgements also have numbers.

Solution for Question 2

Check sum stepwise Explanation/Algorithm

Step-01:

At sender side,

- If m bit checksum is used, the data unit to be transmitted is divided into segments of m bits.
- All the m bit segments are added.
- The result of the sum is then complemented using 1's complement arithmetic.
- The value so obtained is called as checksum.

Step-02:

- The data along with the checksum value is transmitted to the receiver.

Step-03:

At receiver side,

- If m bit checksum is being used, the received data unit is divided into segments of m bits.
- All the m bit segments are added along with the checksum value.
- The value so obtained is complemented and the result is checked.

Then, following two cases are possible-

Case-01: Result = 0

If the result is zero,

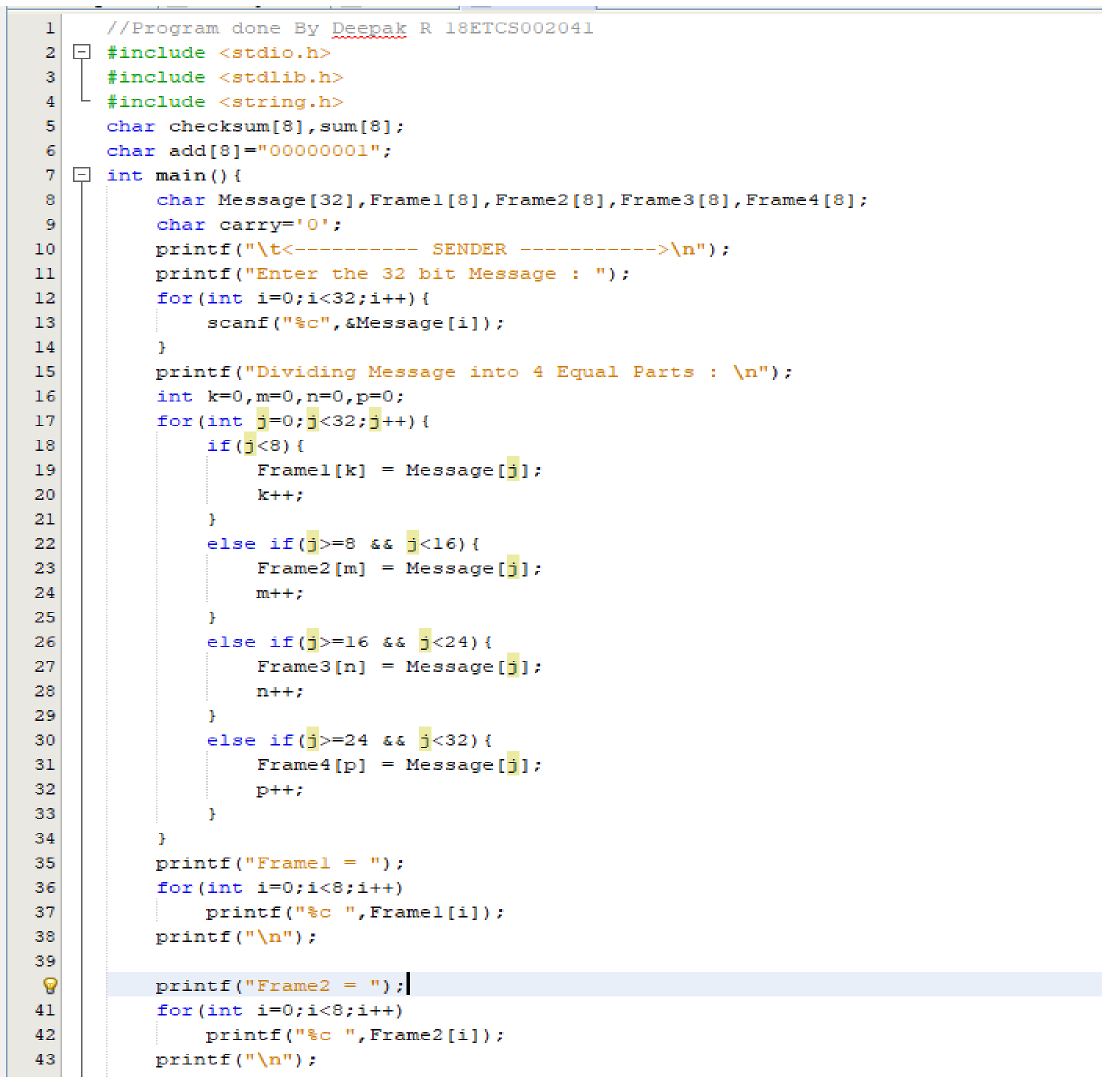
- Receiver assumes that no error occurred in the data during the transmission.
- Receiver accepts the data.

Case-02: Result $\neq$ 0

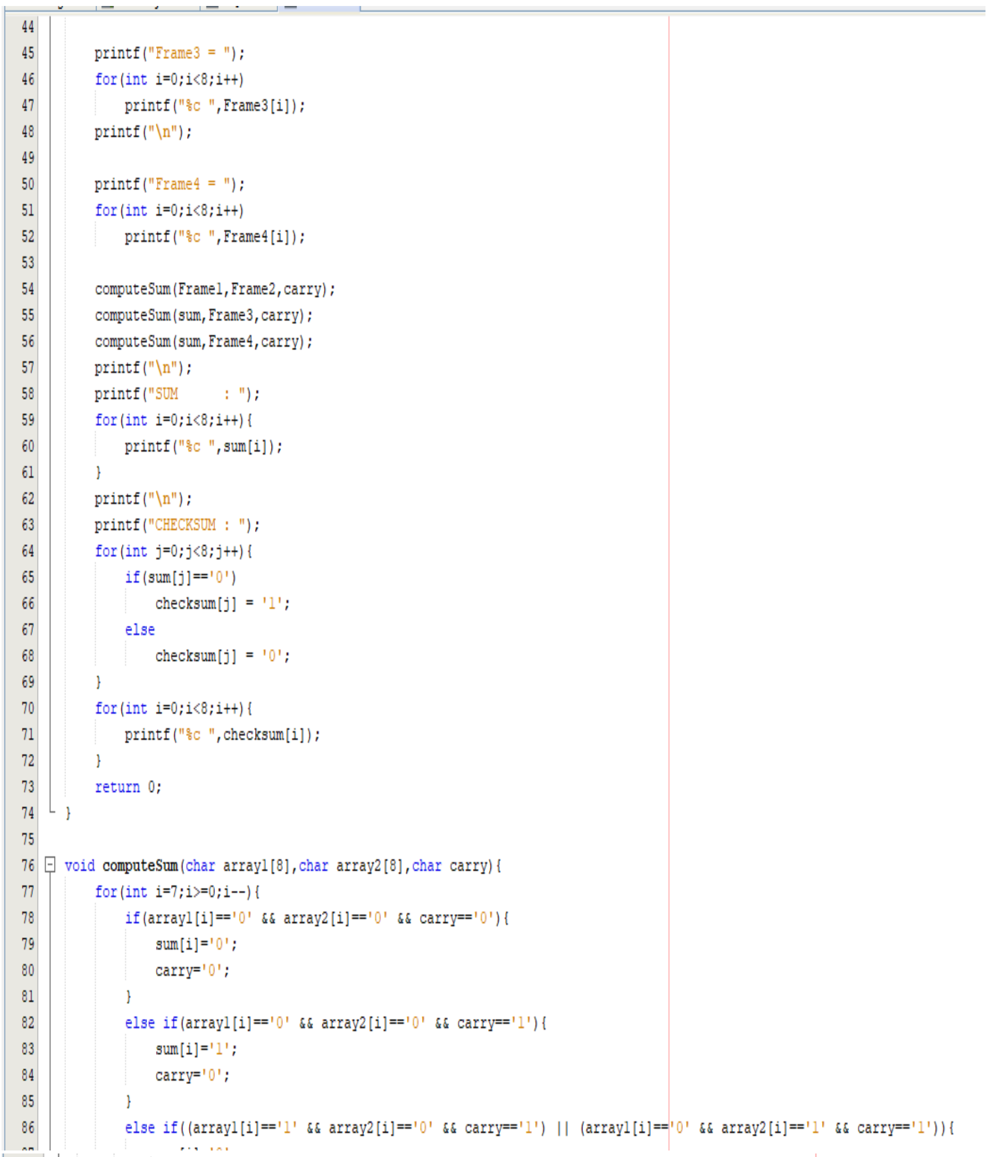
If the result is non-zero,

- Receiver assumes that error occurred in the data during the transmission.
- Receiver discards the data and asks the sender for retransmission.

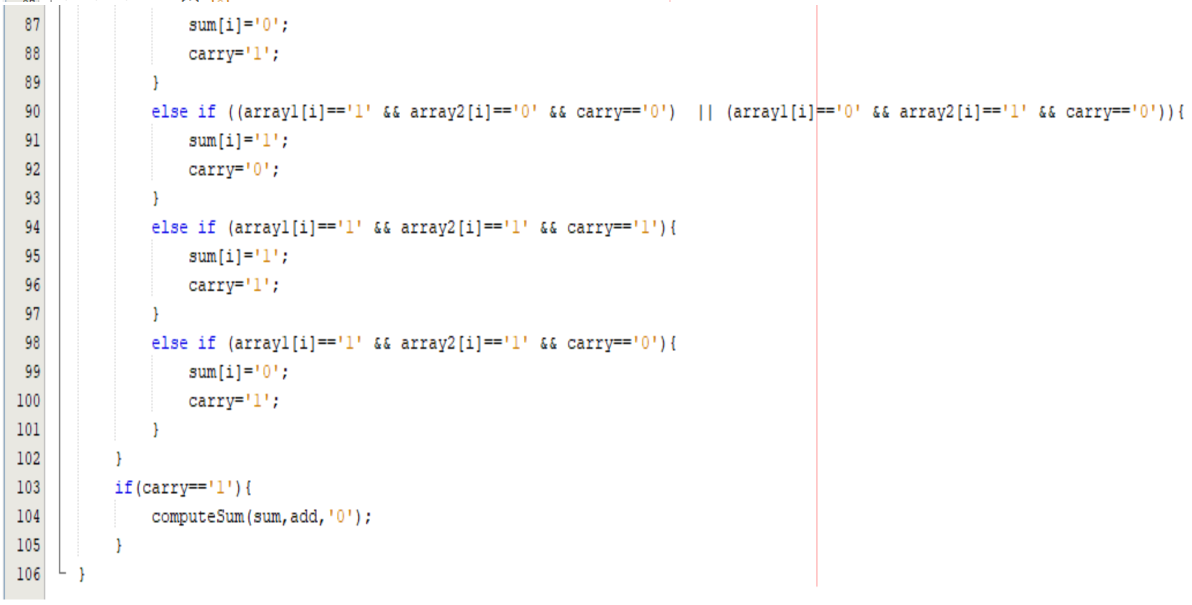
Sender.c



```c
//Program done By Deepak R 18ETCS002041
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
char checksum[8],sum[8];
char add[8]="00000001";
int main(){
    char Message[32],Frame1[8],Frame2[8],Frame3[8],Frame4[8];
    char carry='0';
    printf("\t<---------- SENDER ----------->\n");
    printf("Enter the 32 bit Message : ");
    for(int i=0;i<32;i++){
        scanf("%c",&Message[i]);
    }
    printf("Dividing Message into 4 Equal Parts : \n");
    int k=0,m=0,n=0,p=0;
    for(int j=0;j<32;j++){
        if(j<8){
            Frame1[k] = Message[j];
            k++;
        }
        else if(j>=8 && j<16){
            Frame2[m] = Message[j];
            m++;
        }
        else if(j>=16 && j<24){
            Frame3[n] = Message[j];
            n++;
        }
        else if(j>=24 && j<32){
            Frame4[p] = Message[j];
            p++;
        }
    }
    printf("Frame1 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame1[i]);
    printf("\n");

    printf("Frame2 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame2[i]);
    printf("\n");
```

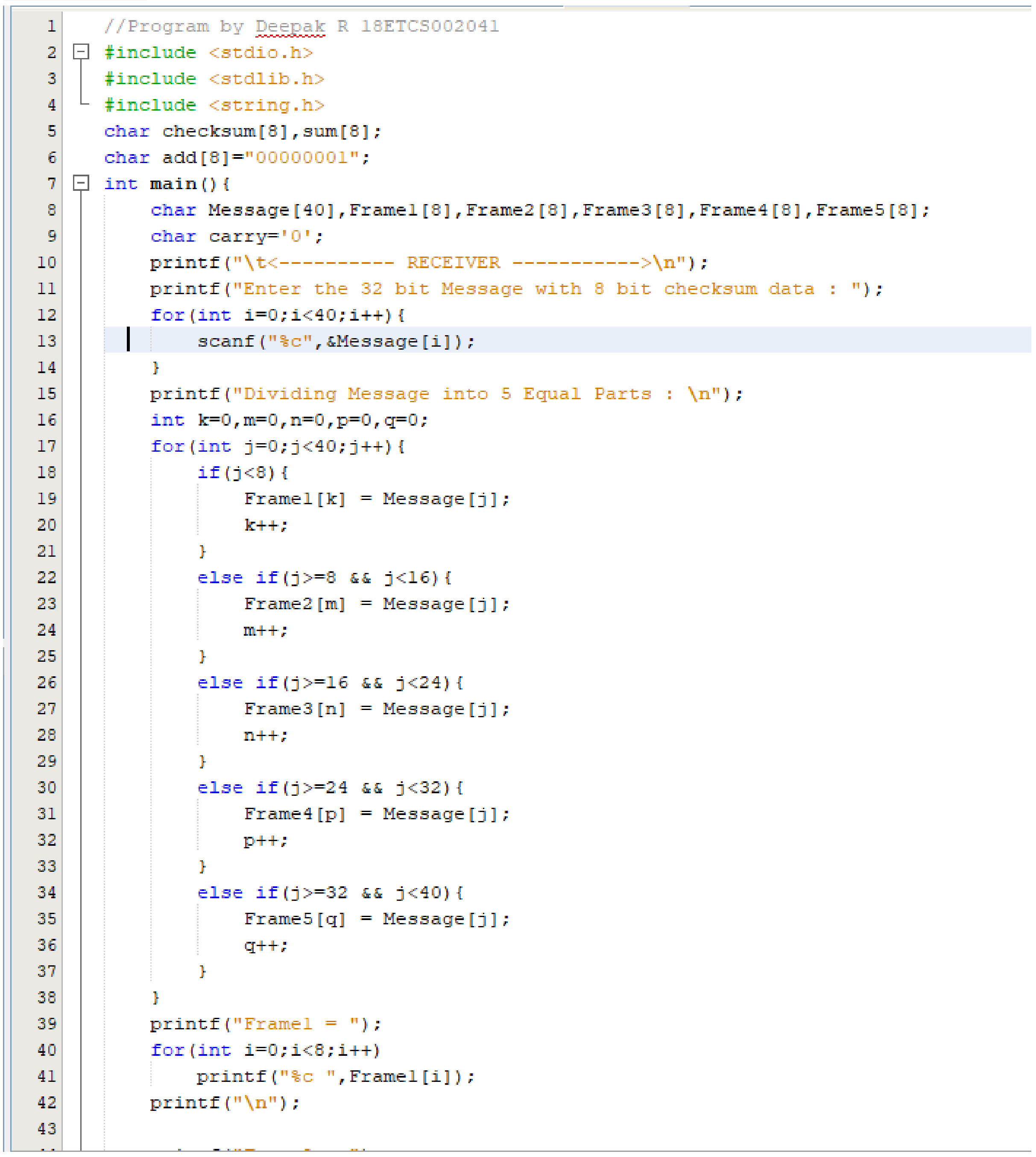


```c
    printf("Frame3 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame3[i]);
    printf("\n");

    printf("Frame4 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame4[i]);

    computeSum(Frame1,Frame2,carry);
    computeSum(sum,Frame3,carry);
    computeSum(sum,Frame4,carry);
    printf("\n");
    printf("SUM      : ");
    for(int i=0;i<8;i++){
        printf("%c ",sum[i]);
    }
    printf("\n");
    printf("CHECKSUM : ");
    for(int j=0;j<8;j++){
        if(sum[j]=='0')
            checksum[j] = '1';
        else
            checksum[j] = '0';
    }
    for(int i=0;i<8;i++){
        printf("%c ",checksum[i]);
    }
    return 0;
}

void computeSum(char array1[8],char array2[8],char carry){
    for(int i=7;i>=0;i--){
        if(array1[i]=='0' && array2[i]=='0' && carry=='0'){
            sum[i]='0';
            carry='0';
        }
        else if(array1[i]=='0' && array2[i]=='0' && carry=='1'){
            sum[i]='1';
            carry='0';
        }
        else if((array1[i]=='1' && array2[i]=='0' && carry=='1') || (array1[i]=='0' && array2[i]=='1' && carry=='1')){
```

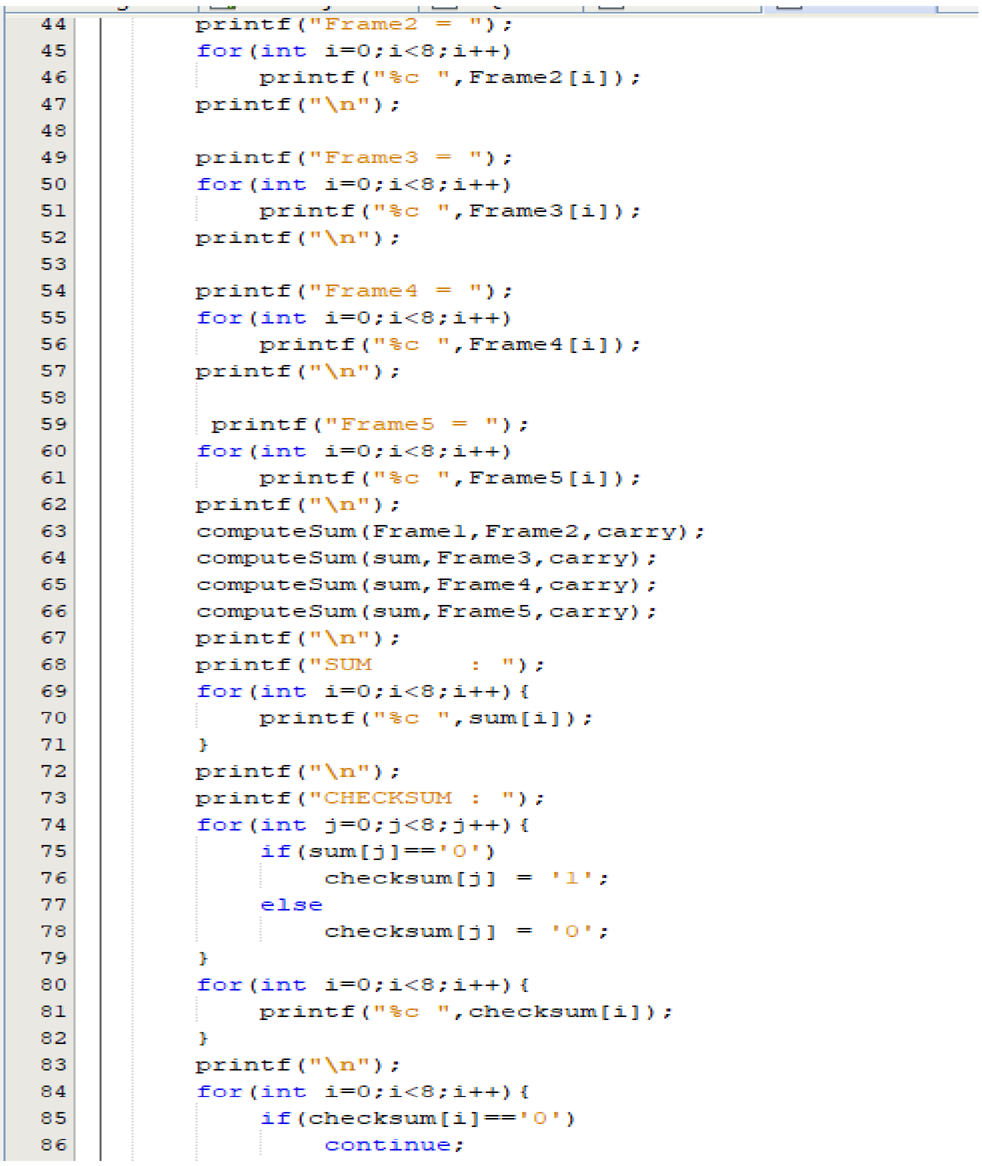


```c
            sum[i]='0';
            carry='1';
        }
        else if ((array1[i]=='1' && array2[i]=='0' && carry=='0')  || (array1[i]=='0' && array2[i]=='1' && carry=='0')){
            sum[i]='1';
            carry='0';
        }
        else if (array1[i]=='1' && array2[i]=='1' && carry=='1'){
            sum[i]='1';
            carry='1';
        }
        else if (array1[i]=='1' && array2[i]=='1' && carry=='0'){
            sum[i]='0';
            carry='1';
        }
    }
    if(carry=='1'){
        computeSum(sum,add,'0');
    }
}
```

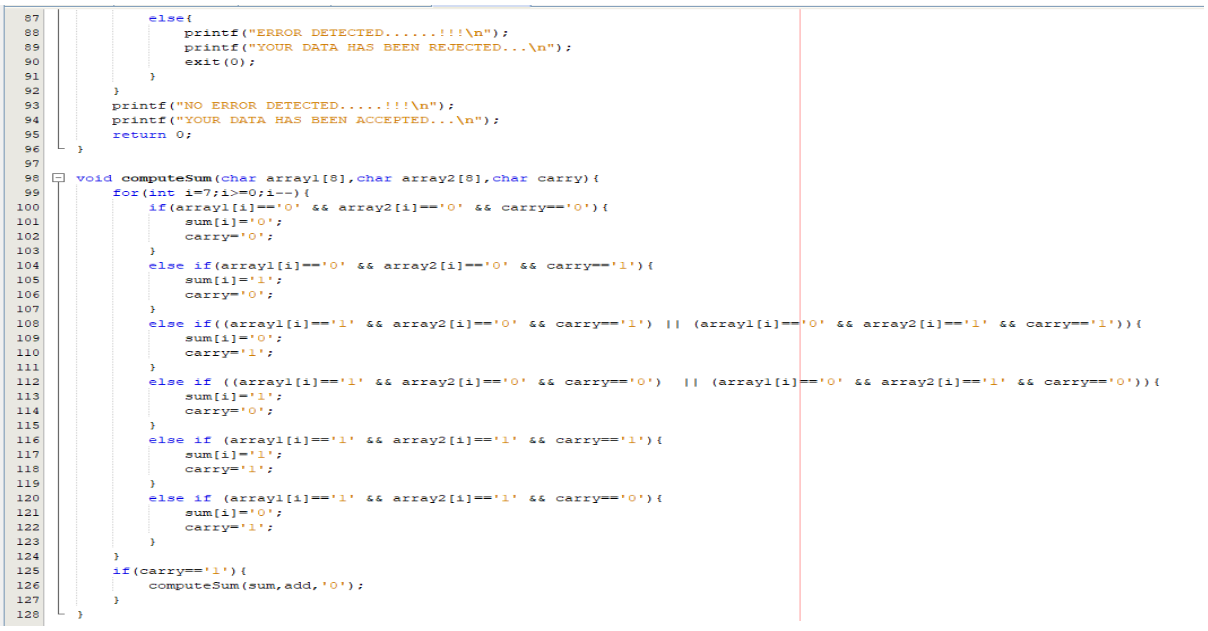
Receiver.c



```c
//Program by Deepak R 18ETCS002041
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
char checksum[8],sum[8];
char add[8]="00000001";
int main(){
    char Message[40],Frame1[8],Frame2[8],Frame3[8],Frame4[8],Frame5[8];
    char carry='0';
    printf("\t<---------- RECEIVER ----------->\n");
    printf("Enter the 32 bit Message with 8 bit checksum data : ");
    for(int i=0;i<40;i++){
        scanf("%c",&Message[i]);
    }
    printf("Dividing Message into 5 Equal Parts : \n");
    int k=0,m=0,n=0,p=0,q=0;
    for(int j=0;j<40;j++){
        if(j<8){
            Frame1[k] = Message[j];
            k++;
        }
        else if(j>=8 && j<16){
            Frame2[m] = Message[j];
            m++;
        }
        else if(j>=16 && j<24){
            Frame3[n] = Message[j];
            n++;
        }
        else if(j>=24 && j<32){
            Frame4[p] = Message[j];
            p++;
        }
        else if(j>=32 && j<40){
            Frame5[q] = Message[j];
            q++;
        }
    }
    printf("Frame1 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame1[i]);
    printf("\n");

```



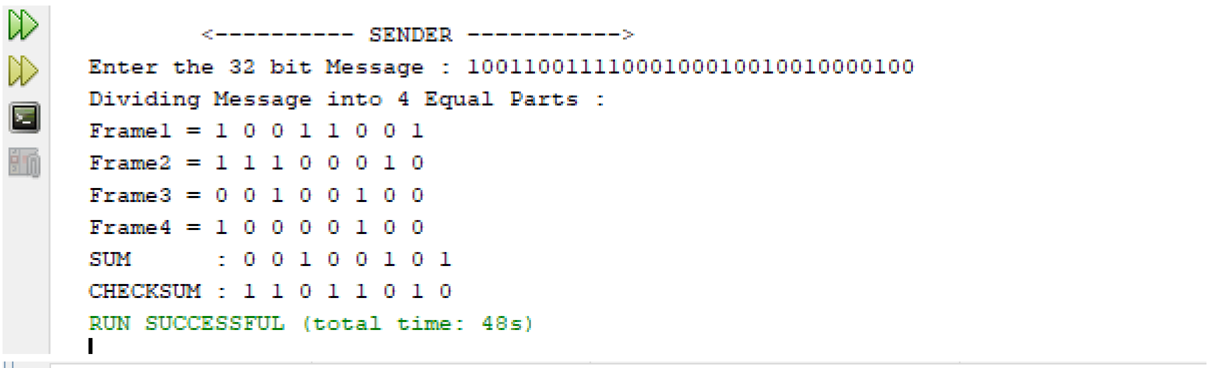
```c
    printf("Frame2 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame2[i]);
    printf("\n");

    printf("Frame3 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame3[i]);
    printf("\n");

    printf("Frame4 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame4[i]);
    printf("\n");

     printf("Frame5 = ");
    for(int i=0;i<8;i++)
        printf("%c ",Frame5[i]);
    printf("\n");
    computeSum(Frame1,Frame2,carry);
    computeSum(sum,Frame3,carry);
    computeSum(sum,Frame4,carry);
    computeSum(sum,Frame5,carry);
    printf("\n");
    printf("SUM        :  ");
    for(int i=0;i<8;i++){
        printf("%c ",sum[i]);
    }
    printf("\n");
    printf("CHECKSUM :  ");
    for(int j=0;j<8;j++){
        if(sum[j]=='0')
            checksum[j] = '1';
        else
            checksum[j] = '0';
    }
    for(int i=0;i<8;i++){
        printf("%c ",checksum[i]);
    }
    printf("\n");
    for(int i=0;i<8;i++){
        if(checksum[i]=='0')
            continue;
```



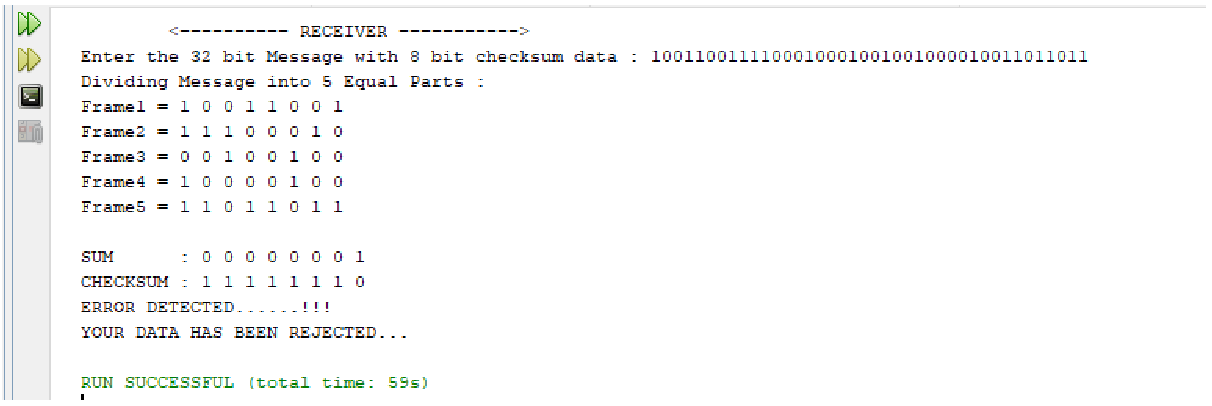
```c
        else{
            printf("ERROR DETECTED......!!!\n");
            printf("YOUR DATA HAS BEEN REJECTED...\n");
            exit(0);
        }
    }
    printf("NO ERROR DETECTED.....!!!\n");
    printf("YOUR DATA HAS BEEN ACCEPTED...\n");
    return 0;
}

void computeSum(char array1[8],char array2[8],char carry){
    for(int i=7;i>=0;i--){
        if(array1[i]=='0' && array2[i]=='0' && carry=='0'){
            sum[i]='0';
            carry='0';
        }
        else if(array1[i]=='0' && array2[i]=='0' && carry=='1'){
            sum[i]='1';
            carry='0';
        }
        else if((array1[i]=='1' && array2[i]=='0' && carry=='1') || (array1[i]=='0' && array2[i]=='1' && carry=='1')){
            sum[i]='0';
            carry='1';
        }
        else if ((array1[i]=='1' && array2[i]=='0' && carry=='0')  || (array1[i]=='0' && array2[i]=='1' && carry=='0')){
            sum[i]='1';
            carry='0';
        }
        else if (array1[i]=='1' && array2[i]=='1' && carry=='1'){
            sum[i]='1';
            carry='1';
        }
        else if (array1[i]=='1' && array2[i]=='1' && carry=='0'){
            sum[i]='0';
            carry='1';
        }
    }
    if(carry=='1'){
        computeSum(sum,add,'0');
    }
}
```

Output with Explanation for Errored output

Test case1



```
         <---------- SENDER ----------->
Enter the 32 bit Message : 10011001111000100010010010000100
Dividing Message into 4 Equal Parts :
Frame1 = 1 0 0 1 1 0 0 1
Frame2 = 1 1 1 0 0 0 1 0
Frame3 = 0 0 1 0 0 1 0 0
Frame4 = 1 0 0 0 0 1 0 0
SUM       : 0 0 1 0 0 1 0 1
CHECKSUM : 1 1 0 1 1 0 1 0
RUN SUCCESSFUL (total time: 48s)
```



```
        <---------- RECEIVER ----------->
Enter the 32 bit Message with 8 bit checksum data : 1001100111100010001001001000010011011011
Dividing Message into 5 Equal Parts :
Frame1 = 1 0 0 1 1 0 0 1
Frame2 = 1 1 1 0 0 0 1 0
Frame3 = 0 0 1 0 0 1 0 0
Frame4 = 1 0 0 0 0 1 0 0
Frame5 = 1 1 0 1 1 0 1 1

SUM      : 0 0 0 0 0 0 0 1
CHECKSUM : 1 1 1 1 1 1 1 0
ERROR DETECTED......!!!
YOUR DATA HAS BEEN REJECTED...

RUN SUCCESSFUL (total time: 59s)
```

At sender side,

The given data unit is divided into segments of 8 bits as-

10011001	11100010	00100100	10000100

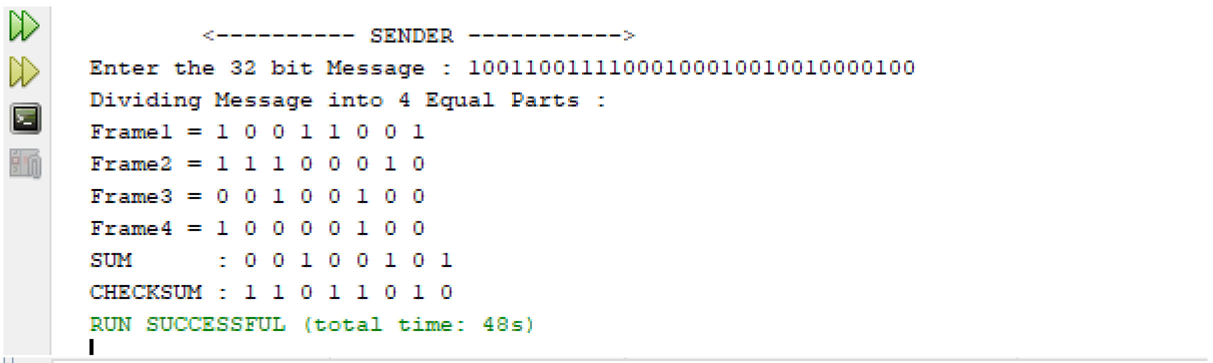
Now, all the segments are added and the result is obtained as-

- **10011001 + 11100010 + 00100100 + 10000100 = 1000100011**
- **Since the result consists of 10 bits, so extra 2 bits are wrapped around.**
- **00100011 + 10 = 00100101 (8 bits)**
- **Now, 1's complement is taken which is 11011010.**
- **Thus, checksum value = 11011010**

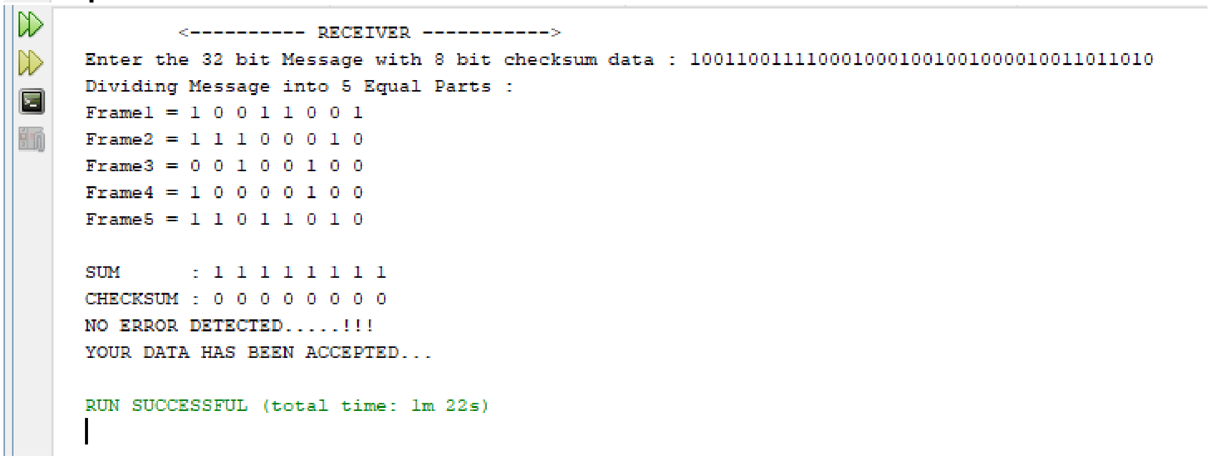
At receiver side,

- **The received data unit is divided into segments of 8 bits.**
- **All the segments along with the checksum value are added.**
- **Complemented value = 00000001**
- **Since the result is not a 0, receiver Says error occurred in the data and therefore Rejected it.**

Test case 2



```
        <---------- SENDER ----------->
Enter the 32 bit Message : 10011001111000100010010010000100
Dividing Message into 4 Equal Parts :
Frame1 = 1 0 0 1 1 0 0 1
Frame2 = 1 1 1 0 0 0 1 0
Frame3 = 0 0 1 0 0 1 0 0
Frame4 = 1 0 0 0 0 1 0 0
SUM      : 0 0 1 0 0 1 0 1
CHECKSUM : 1 1 0 1 1 0 1 0
RUN SUCCESSFUL (total time: 48s)
```



```
        <---------- RECEIVER ----------->
Enter the 32 bit Message with 8 bit checksum data : 1001100111100010001001001000010011011010
Dividing Message into 5 Equal Parts :
Frame1 = 1 0 0 1 1 0 0 1
Frame2 = 1 1 1 0 0 0 1 0
Frame3 = 0 0 1 0 0 1 0 0
Frame4 = 1 0 0 0 0 1 0 0
Frame5 = 1 1 0 1 1 0 1 0

SUM      : 1 1 1 1 1 1 1 1
CHECKSUM : 0 0 0 0 0 0 0 0
NO ERROR DETECTED.....!!!
YOUR DATA HAS BEEN ACCEPTED...

RUN SUCCESSFUL (total time: 1m 22s)
```

At sender side,

The given data unit is divided into segments of 8 bits as-

10011001	11100010	00100100	10000100

Now, all the segments are added and the result is obtained as-

- 10011001 + 11100010 + 00100100 + 10000100 = 1000100011
- Since the result consists of 10 bits, so extra 2 bits are wrapped around.
- 00100011 + 10 = 00100101 (8 bits)
- Now, 1's complement is taken which is 11011010.
- Thus, checksum value = 11011010

At receiver side,

- The received data unit is divided into segments of 8 bits.
- All the segments along with the checksum value are added.
- Sum of all segments + Checksum value = 00100101 + 11011010 = 11111111
- Complemented value = 00000000
- Since the result is 0, receiver assumes no error occurred in the data and therefore accepts it.

References

Computer Networks 5th Edition is a book authored by Andrew S. Tanenbaum and David J. Wetherall.