**NETWORK AND COMMUNICATIONS**

**LAB ASSIGNMENT-2**

ERROR DETECTION AND FLOW CONTROL MECHANISM

**Name:** SAKSHAM MINOCHA

**Reg No.:** 19BCE0466

**Q1.** Write a C program for detecting error using Hamming code.

**Algorithm:**

1. Accept the length of the data string from the user.
2. Then, the data to be transmitted is entered as an input.
3. The length of the hamming code is calculated and it is displayed along with the number of parity bits and their values.
4. The sender side receiver code is displayed.
5. It is asked to enter the receiver side hamming code in order to perform the detection of the error.
6. Then, after performing the calculation, the values of the parity bits is displayed and the position at which the error has occurred is also shown.
7. After that, the program corrects the error bit and complements it (i.e. 1 becomes 0 and vice-versa).
8. The corrected code is displayed to the user.

**Code:**

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

void main()

{

int maxp=6;

int a[50],temp[70],temp2[70];

int t,i,j,k,nd,n,nh,sum=0,pos=0;

printf("Enter Length of Data String: ");

scanf("%d",&nd);

printf("Enter Data String: ");

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

{

scanf("%d",&a[i]);

}

printf("\n",nd);

for(i=0,j=0;i<nd;i++)

{

for(k=0;k<maxp;k++)

{

t=pow(2,k)-1;

if(j==t)

{

temp[j]=0;

j++;

}

}

temp[j]=a[i];

j++;

}

nh=j;

printf("Length of Hamming code: %d bits\n",nh);

n=nh-nd;

printf("Number of Parity Bits: %d \n",n);

int b[n];

int m=n-1;

for(k=0;k<n;k++)

{

t=pow(2,k)-1;

for(i=t;i<nh;)

{

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

{

sum=sum+temp[i];

i++;

if(i>=nh)

break;

}

if(i>=nh)

break;

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

{

i++;

if(i>=nh)

break;

}

if(i>=nh)

break;

}

temp[t]=sum%2;

sum=0;

printf("P%d: %d\n",t+1,temp[t]);

}

printf("\nHamming code: Sender side: ");

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

{

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

}

printf("\nHamming code: Receiver side: ");

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

{

scanf("%d",&temp2[i]);

}

sum=0;

for(k=0;k<n;k++)

{

t=pow(2,k)-1;

for(i=t;i<nh;)

{

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

{

sum=sum+temp2[i];

i++;

if(i>=nh)

break;

}

if(i>=nh)

break;

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

{

i++;

if(i>=nh)

break;

}

if(i>=nh)

break;

}

b[m]=sum%2;

sum=0;

printf("P%d: %d\n",t+1,b[m]);

m--;

}

for(m=0;m<n;m++)

{

pos=pos+b[n-m-1]\*pow(2,m);

}

printf("Position of Error: %d\n",pos);

if(temp2[pos-1]==0)

temp2[pos-1]=1;

else

temp2[pos-1]=0;

printf("\nHamming code: Receiver side: Error Corrected: ");

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

{

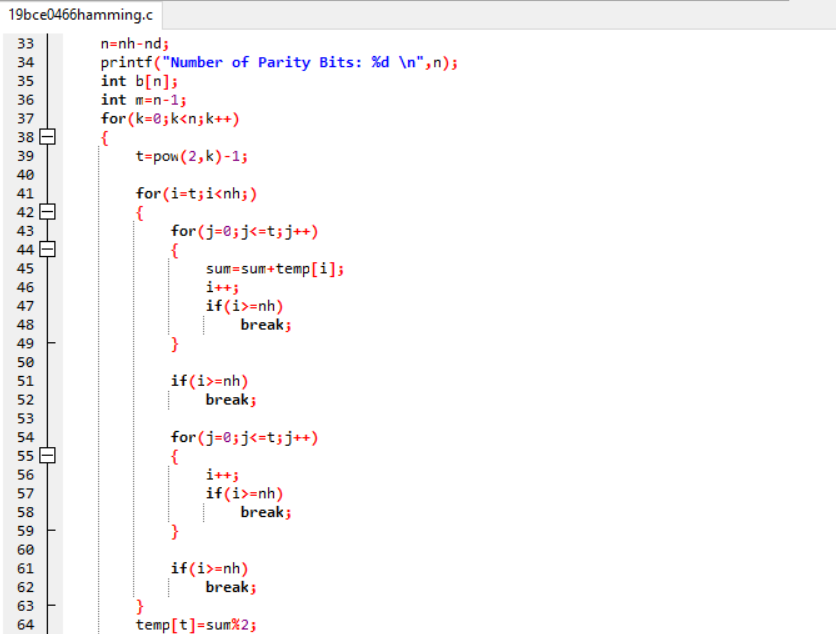
printf("%d ",temp2[i]);

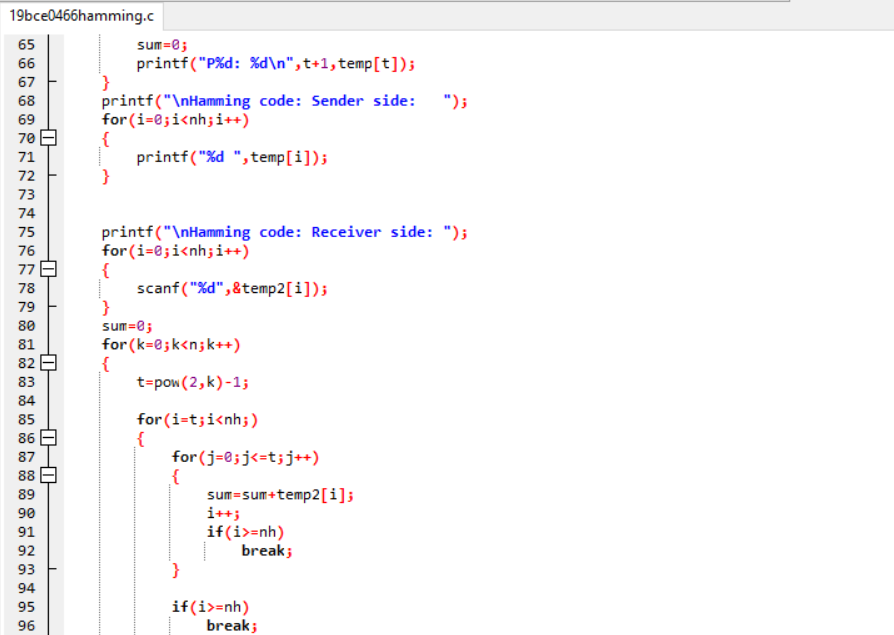
}

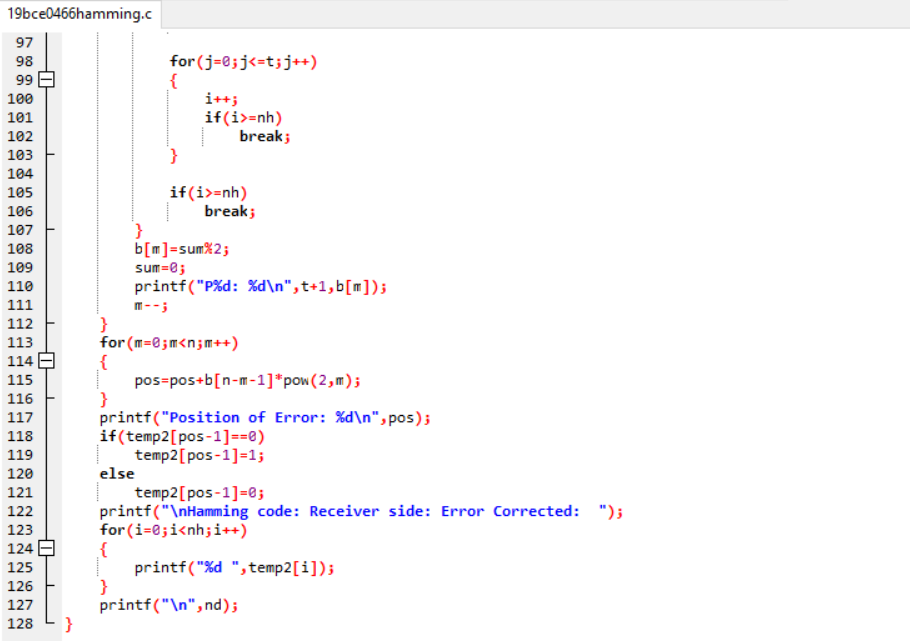
printf("\n",nd);

}

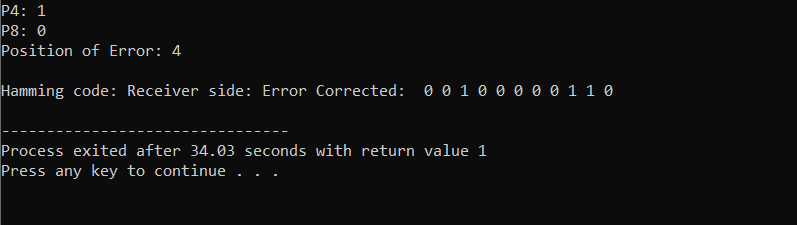
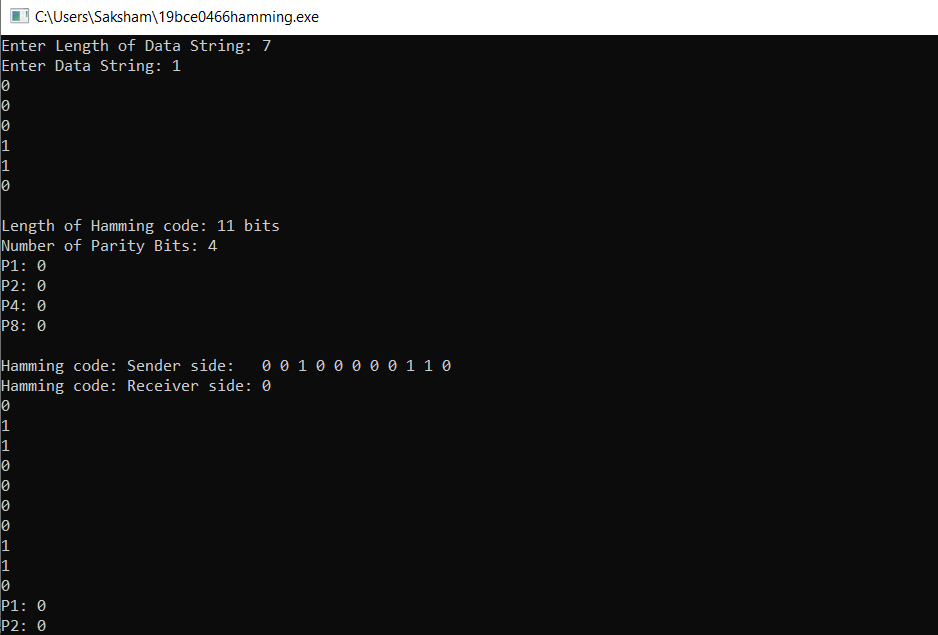








**Output:**



**Q2.** Write a C program to implement CRC error detection method.

**Algorithm:**

1. In the program of cyclic redundancy check, the number of bits in both data and the divisor is asked.
2. The user then enters the data to be sent and the divisor.
3. The operation of division is performed and a remainder is obtained.
4. The first remainder is calculated.
5. The remainder bit is then appended to the original data bits.
6. The final remainder bit is then calculated and if it comes out be zero, it means no error has occurred.
7. Else, the error has occurred.

**Code:**

#include<stdio.h>

#include<conio.h>

void main()

{

int

arr1[20],arr2[20],divarr[10],n,d,i,j,k,l,x,z=0,temp1[]={1,1,1,1,1,1,1},temp0[]={0,0,0,0,0,

0,0},

restarr[10],restarr1[10];

printf("Enter the number bits in data\n");

scanf("%d",&n);

printf("Enter the number of bits in the divisor\n");

scanf("%d",&d);

printf("Enter the data:\n");

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

scanf("%d",&arr1[i]);

printf("Enter the divisor:\n");

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

scanf("%d",&divarr[j]);

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

{

arr2[i]=arr1[i];

}

for(i=0;i<d-1;i++)

{

arr1[n]=0;

n++;

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

{

if(arr1[i]==divarr[0])

{

k=i;

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

{

if(arr1[k]==divarr[j])

{

arr1[k]=0;

k++;

}

else

{

arr1[k]=1;

k++;

}

}

}

else if(arr1[i]==1)

{

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

{

k=i;

if(arr1[k]==temp1[j])

{

arr1[k]=0;

k++;

}

else

{

arr1[k]=1;

k++;

}

}

}

else if(arr1[i]==0)

{

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

{

k=i;if(arr1[k]==temp0[j])

{

arr1[k]=0;

k++;

}

else

{

arr1[k]=1;

k++;

}

}

}

}

x=n;

for(i=0;i<d-1;i++)

{

restarr[i]=arr1[x-2];

x++;

}

printf("The first remainder is:\n");

for(i=0;i<d-1;i++)

{

printf("%d\n",restarr[i]);

}

getch();

printf("The original value is:\n");

for(i=0;i<6;i++)

{

printf("%d\n",arr2[i]);

}

x=n;

for(i=0;i<d-1;i++)

{

arr2[x-3]=restarr[i];

x++;

}

printf("The remainder bit is appended with the original data\n");

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

{

printf("%d\n",arr2[i]);

}

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

{

if(arr2[i]==divarr[0])

{

k=i;for(j=0;j<d;j++)

{

if(arr2[k]==divarr[j])

{

arr2[k]=0;

k++;

}

else

{

arr2[k]=1;

k++;

}

}

}

else if(arr2[i]==1)

{

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

{

k=i;

if(arr2[k]==temp1[j])

{

arr2[k]=0;

k++;

}

else

{

arr2[k]=1;

k++;

}

}

}

else if(arr2[i]==0)

{

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

{

k=i;

if(arr2[k]==temp0[j])

{

arr1[k]=0;

k++;

}

else

{

arr2[k]=1;k++;

}

}

}

}

for(i=0;i<d-1;i++)

{

restarr1[i]= arr2[n-3];

n++;

}

printf("The final remainder is:\n");

for(i=0;i<d-1;i++)

{

printf("%d\n",restarr1[i]);

}

for(i=0;i<d-1;i++)

{

if(restarr1[i]==0)

z++;

}

if(z==d-1)

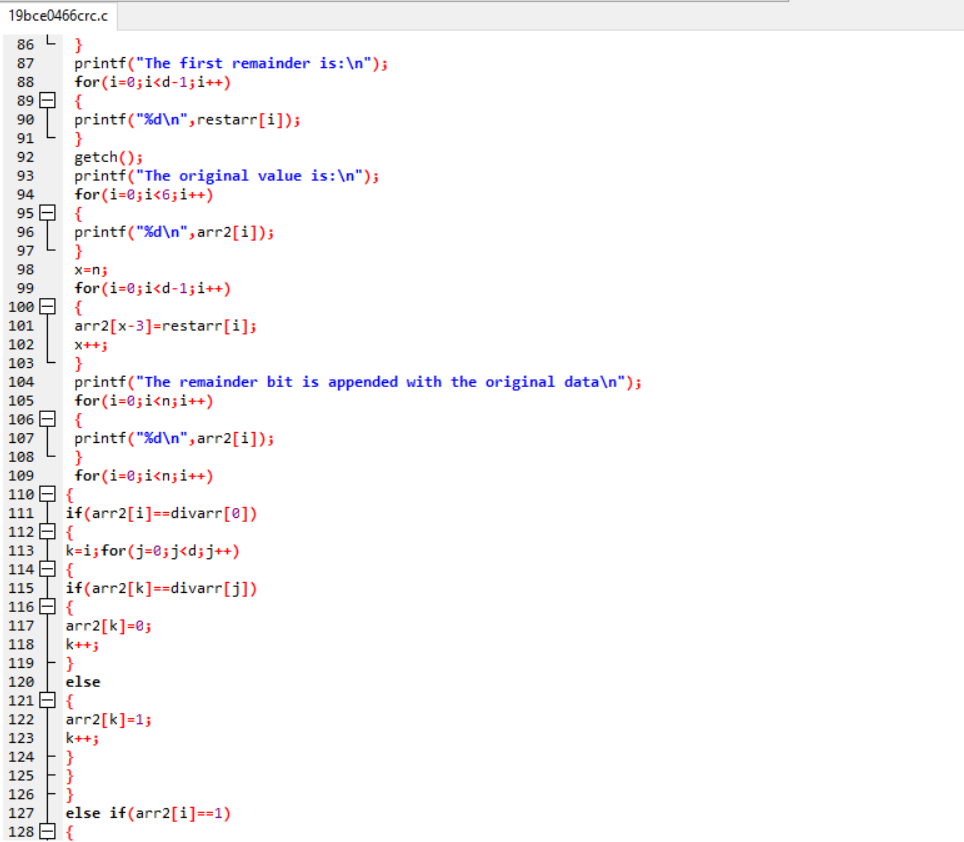
printf("There is no error in the bit\n");

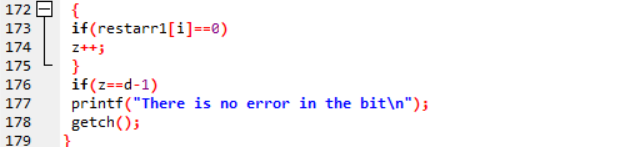
getch();

}

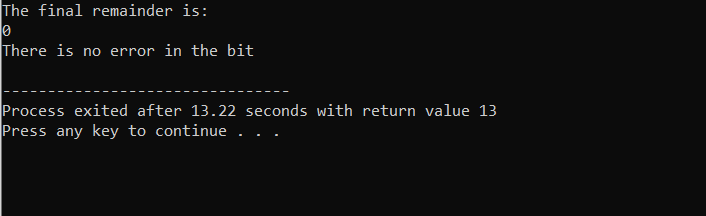
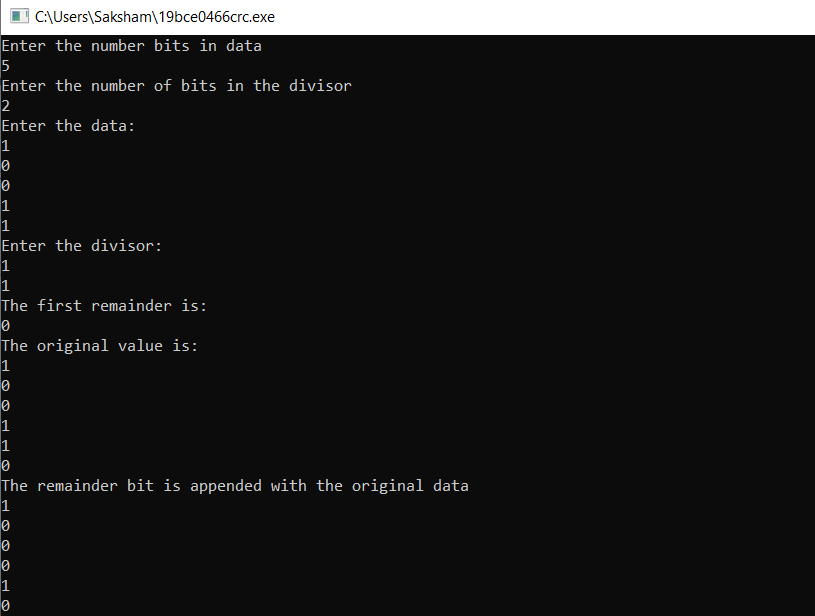








**Output:**



**Q3.** Write a C program to implement error detection using checksum.

**Algorithm:**

1. Accept the choice of the user whether the user is a sender or a receiver.
2. For sender, accept the data to be transmitted.
3. The data in the form of string is divided into 2-byte words and change them into a 2-digit hexadecimal number.
4. The checksum is calculated and shown to the user.
5. Now, at the receiver side, the data received is entered by the receiver along with the checksum received.
6. Now, again the checksum for the data received is generated and it is compared with the checksum received.
7. The message is accepted if both the checksums are same, otherwise an error has occurred.

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char data[100];

int rightSum(int l)

{

int sum=0, i=1;

for(;i<l;i=i+2)

sum=sum + (int) data[i];

return sum;

}

int leftSum(int l)

{

int sum=0, i=0;

for(;i<l;i=i+2)

sum=sum + (int) data[i];

return sum;

}

int main()

{

char buf[100];

int i, n, op=0, irs=0, ils=0, prs=0, cls=0, wc=0, pls=0, s=0, ocs=0, len=0;

while(op==0)

{

printf("Main Menu....SAKSHAM MINOCHA 19BCE0466\n\n1. Sender\n2. Receiver\n3. Exit\nEnter your choice...");

scanf("%d",&n);

switch(n)

{

case 1:

{

printf("\nEnter the data to be transmitted: ");

gets(buf);

gets(data);

len=strlen(data);

if(len%2!=0)

len++;

irs=rightSum(len);

prs=irs%256;

cls=irs/256;

ils=cls+leftSum(len);

pls=ils%256;

wc=ils/256;

s=pls\*256+prs+wc;

ocs = 65535 - s;

printf("The checksum generated is %X\n", ocs);

}

break;

case 2:

{

char cs[100];

int ch[100];

printf("\nEnter the data received: ");

gets(buf);

gets(data);

printf("\nEnter the received checksum: ");

gets(cs);

len=strlen(data);

if(len%2!=0)

len++;

for(i=0;i<strlen(cs);i++)

{

if(cs[i]>='0' && cs[i]<='9')

ch[i]=cs[i]-48;

else if(cs[i]>='A' && cs[i]<='F')

ch[i]=cs[i]-55;

else if(cs[i]>='a' && cs[i]<='f')

ch[i]=cs[i]-87;

}

irs=rightSum(len) + ch[2]\*16 + ch[3];

prs=irs%256;

cls=irs/256;

ils=cls+leftSum(len) + ch[0]\*16 + ch[1];

pls=ils%256;

wc=ils/256;

s=pls\*256+prs+wc;

ocs = 65535 - s;

if(ocs==0)

printf("\nThe message is accepted!\n");

else

printf("\nThe message is rejected as there is an error!\n");

}

break;

case 3: exit(0);

}

printf("\nPress 1 to return to main menu or 0 to exit...");

scanf("%d", &i);

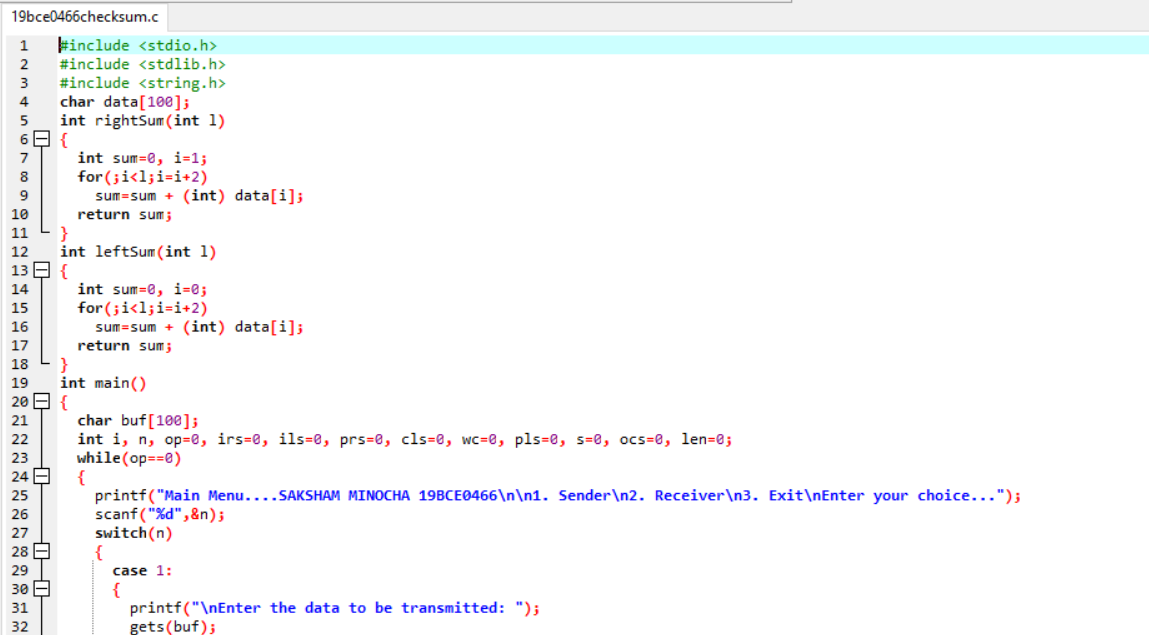
if(i==0)

op=1;

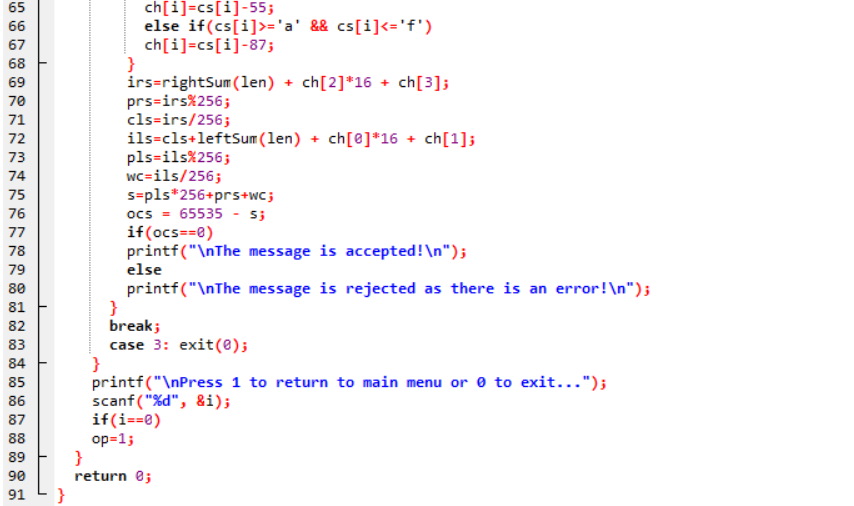
}

return 0;

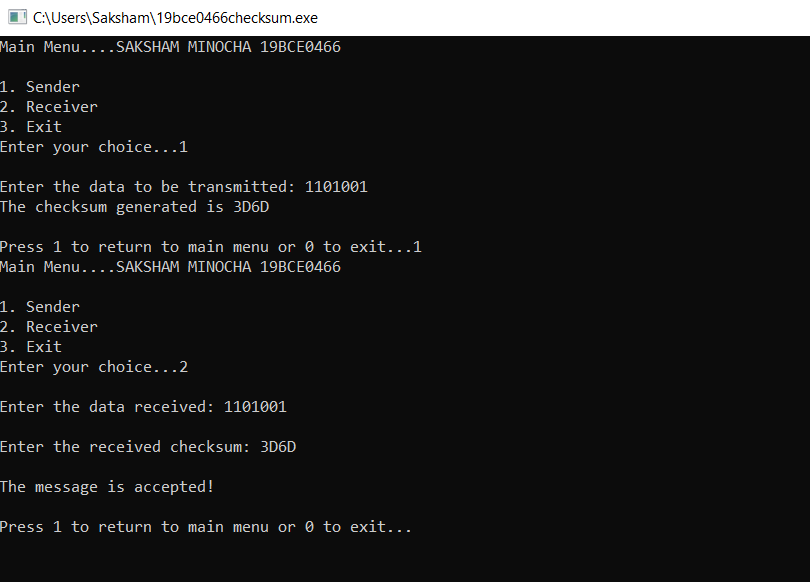
}



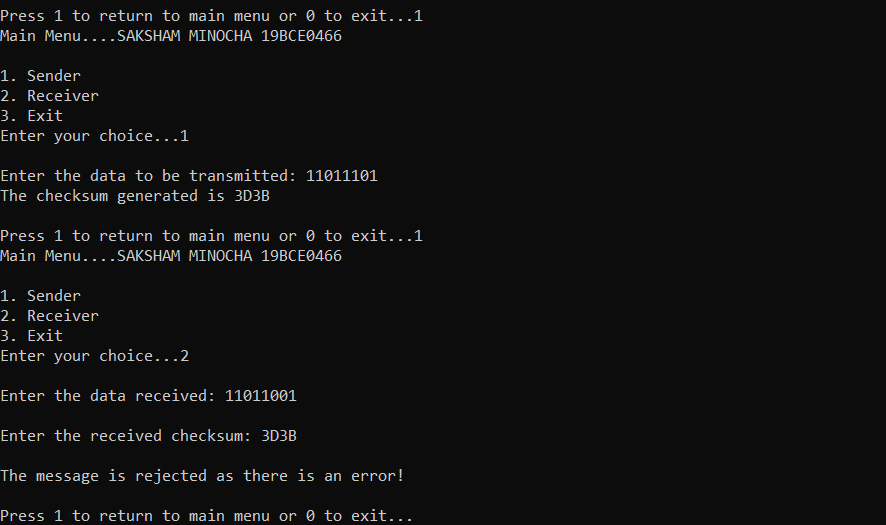




**Output (Data transmitted is in binary form):**

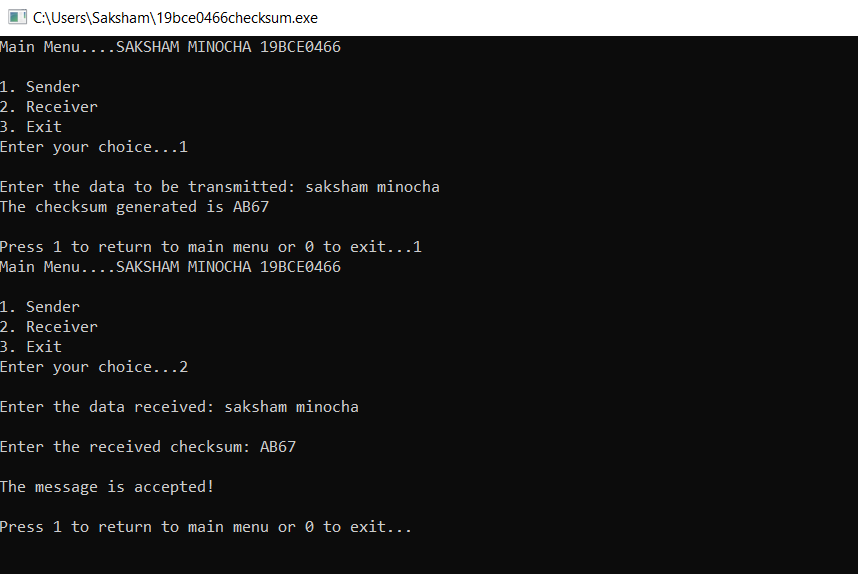


The correct message is received here at the receiver side and hence no error.

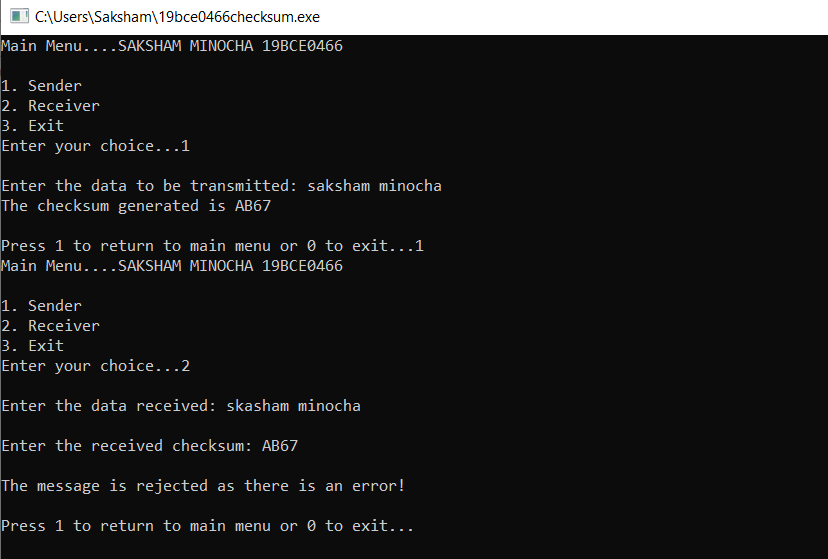


Here, the wrong message is received by the receiver, thus, it rejects the message.

**Output (Data transmitted is in String form):**



Correct data is transmitted; hence the message is accepted.



Purposely, here, **skasham** is written instead of **saksham** to show the error.

**Q4.** Write a C program to implement stop and wait protocol

**Algorithm:**

1. The user is asked to enter the number of frames.
2. First Frame is transmitted.
3. Acknowledgement of the first frame is received.
4. Now transmit the next frame.
5. Find the remaining frames to be transmitted.
6. If for any frame an acknowledgement is not received then re-transmit that specific frame again.
7. Repeat the steps 4 to 6 until the number of frames become zero.

**Code:**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

void main()

{

int i,j,noframes,x,x1=10,x2;

i=1;

j=1;

printf("SAKSHAM MINOCHA 19BCE0466\n");

printf("Enter the number of frames\n");

scanf("%d",&noframes);

getch();

while(noframes>0)

{

printf("\nsending frame %d",i);

srand(x1++);

x = rand()%10;

if(x%2 == 0)

{

for (x2=1; x2<2; x2++)

{

printf("waiting for %d seconds\n", x2);

sleep(x2);

}

printf("\nsending frame %d",i);

srand(x1++);

x = rand()%10;}

printf("\nack for frame %d",j);

noframes-=1;

i++;

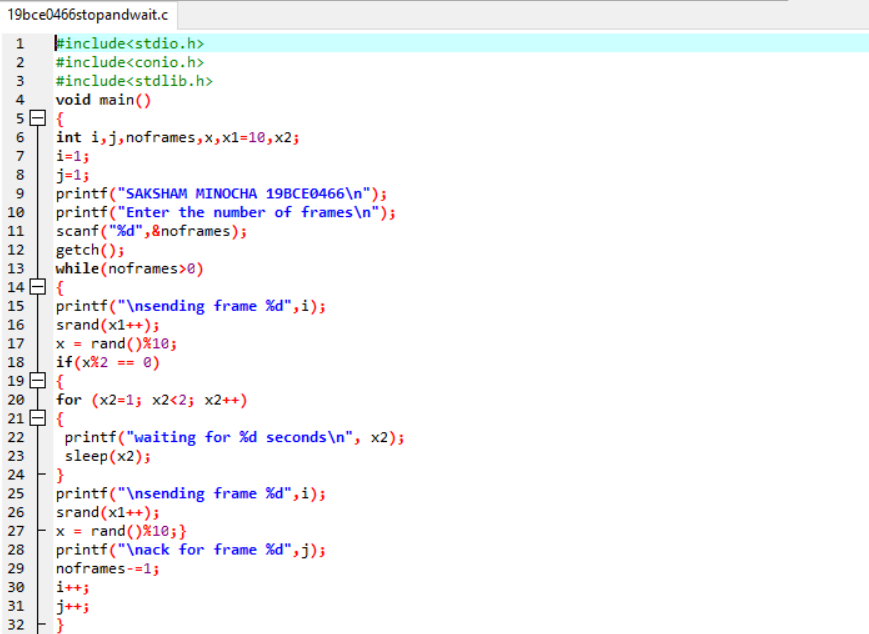
j++;

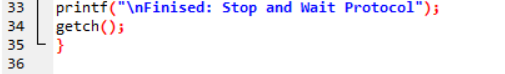
}

printf("\nFinised: Stop and Wait Protocol");

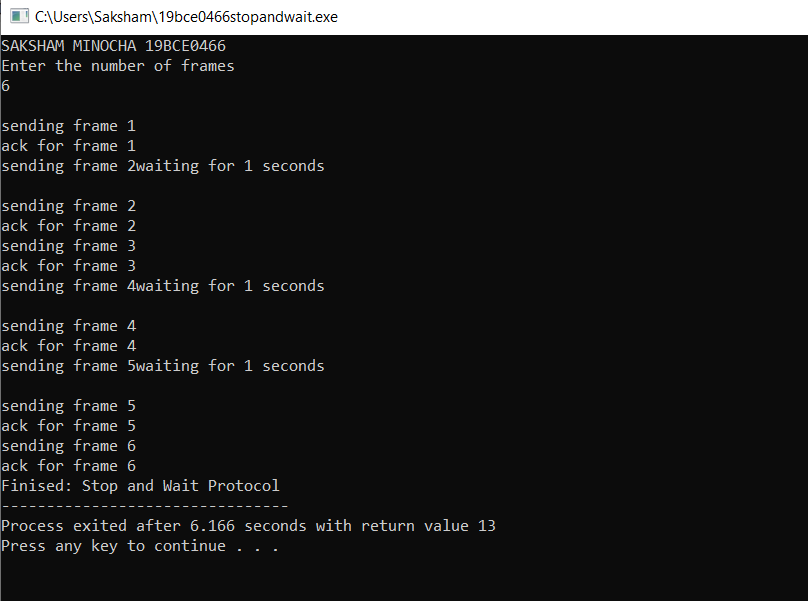
getch();

}





**Output:**



**Q5.** Write a C program to implement Go Back N (Sliding Window Protocol)

**Algorithm:**

1. The user enters the number of frames to be transmitted.
2. The size of the window is set.
3. Identify the number of frames to be transmitted at a given time.
4. Transmit the frames and receive the acknowledgment for the frame sent.
5. Find the remaining frames to be sent.
6. If for any frame, acknowledgement is not received, transmit that frame once again.
7. Repeat the steps from 4 to 6 until the number of frames to be sent becomes zero.

**Code:**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

void main()

{

int temp1,temp2,temp3,temp4,i,winsize=8,noframes,moreframes;

char c;

int reciever(int);

int simulate(int);

temp4=0,temp1=0,temp2=0,temp3=0;

printf("Enter the number of frames:");

scanf("%d",&noframes);

moreframes=noframes;

while(moreframes>=0)

{

temp1=simulate(winsize);

winsize-=temp1;

temp4+=temp1;

if(temp4 >noframes)

temp4 = noframes;

for(i=temp3+1;i<=temp4;i++)

printf("\nsending frame %d",i);

temp2=reciever(temp1);

temp3+=temp2;

if(temp3 > noframes)

temp3 = noframes;

printf("\n acknowledgement for the frames up to %d",temp3);

moreframes-=temp2;

temp4=temp3;

if(winsize<=0)

winsize=8;

}

printf("\nGo Back N Protocol: Finished");

}

int reciever(int temp1)

{

int i;

for(i=1;i<100;i++)

rand();

i=rand()%temp1;

return i;

}

int simulate(int winsize)

{

int temp1,i;

for(i=1;i<50;i++)

temp1=rand();

if(temp1==0)

temp1=simulate(winsize);

i = temp1%winsize;

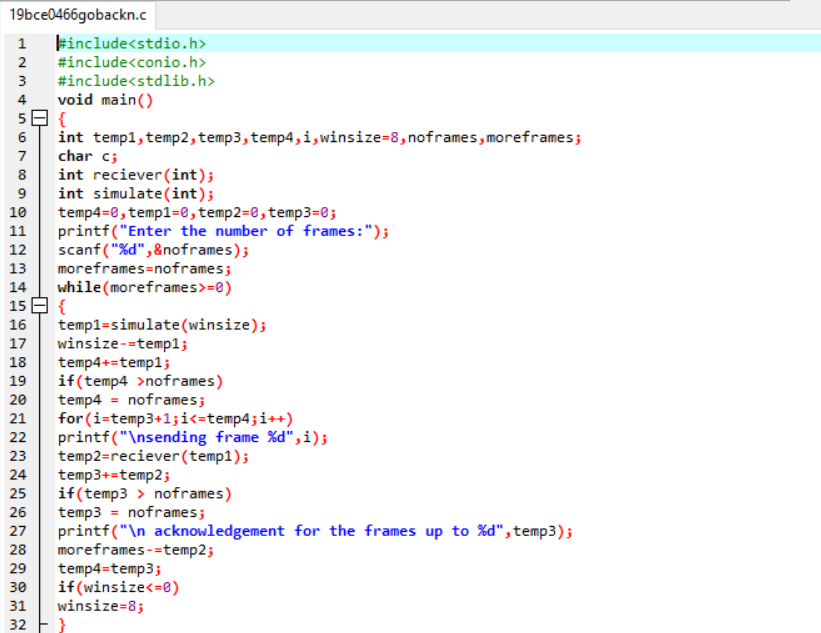
if(i==0)

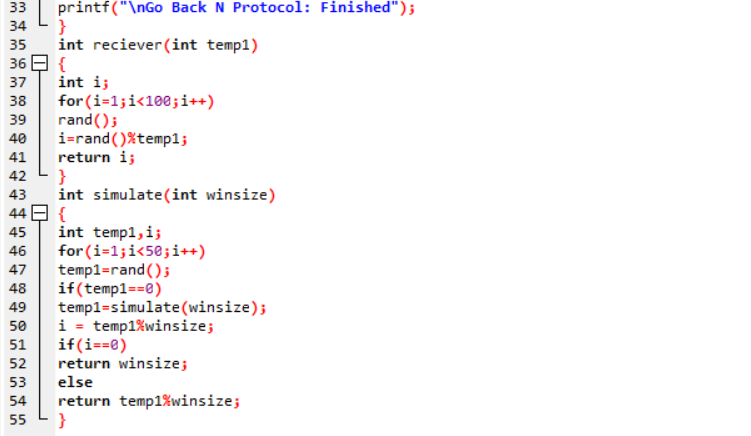
return winsize;

else

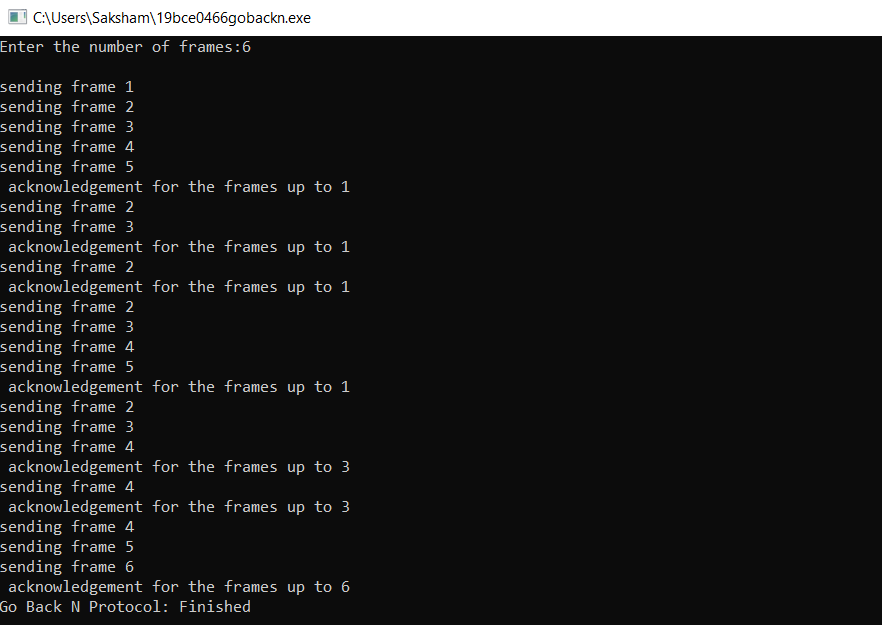
return temp1%winsize;

}





**Output:**



**Q6.** Write a C program to implement Selective Repeat.

**Algorithm:**

1. The user enters the number of frames to be sent.
2. The size of the widow is set.
3. Identify the number of frames to be sent at a given time.
4. Transmit the frame.
5. Wait and receive the acknowledgement frame.
6. Check for the acknowledgement of each frame and repeat the process for that frame for which an acknowledgement is not received, else continue the process.
7. Repeat the steps 4 to 6 until the number of frames to be transmitted becomes zero.

**Code:**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

void main()

{

int temp1,temp2,temp3,temp4,temp5,i,winsize=8,noframes,moreframes;

char c;

int reciever(int);

int simulate(int);

int nack(int);

temp4=0,temp1=0,temp2=0,temp3=0,temp5 = 0;

printf("Enter the number of frames:");

scanf("%d",&noframes);

moreframes=noframes;

while(moreframes>=0)

{

temp1=simulate(winsize);

winsize-=temp1;

temp4+=temp1;

if(temp4 >noframes)

temp4 = noframes;

for(i=noframes - moreframes;i<=temp4;i++)

printf("\nsending frame %d",i);

temp2=reciever(temp1);

temp3+=temp2;

if(temp3 > noframes)

temp3 = noframes;

temp2 = nack(temp1);

temp5+=temp2;

if (temp5 !=0)

{

printf("\n No acknowledgement for the frame %d",temp5);

for(i=1;i<temp5;i++)

;

printf("\n Retransmitting frame %d",temp5);

}

moreframes-=temp1;

if(winsize<=0)

winsize=8;

}

printf("\n end of sliding window protocol Selective Reject");

}

int reciever(int temp1)

{

int i;

for(i=1;i<100;i++)

rand();

i=rand()%temp1;

return i;

}

int nack(int temp1)

{

int i;

for(i=1;i<100;i++)

rand();

i=rand()%temp1;

return i;

}

int simulate(int winsize)

{

int temp1,i;

for(i=1;i<50;i++)

temp1=rand();

if(temp1==0)

temp1=simulate(winsize);

i = temp1%winsize;

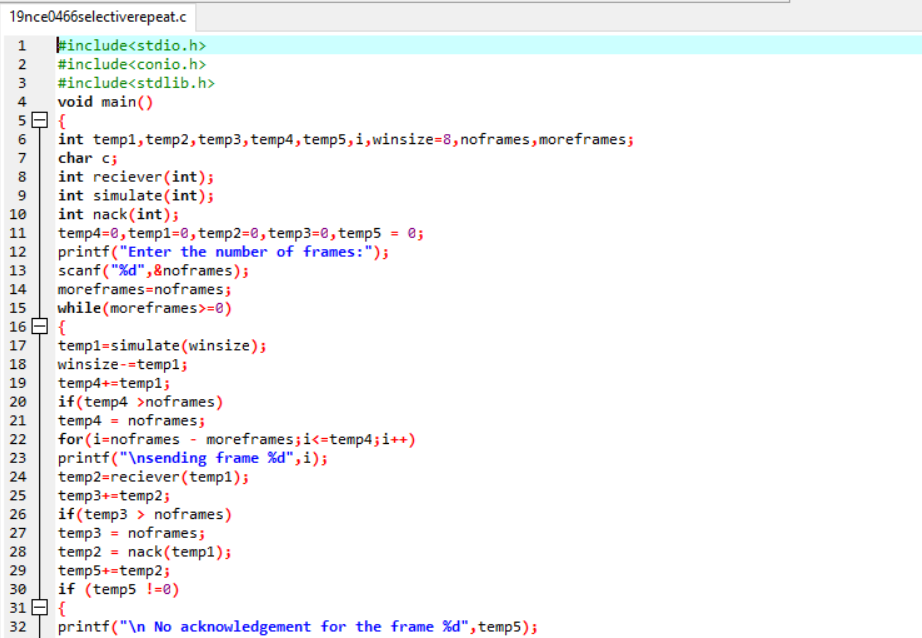
if(i==0)

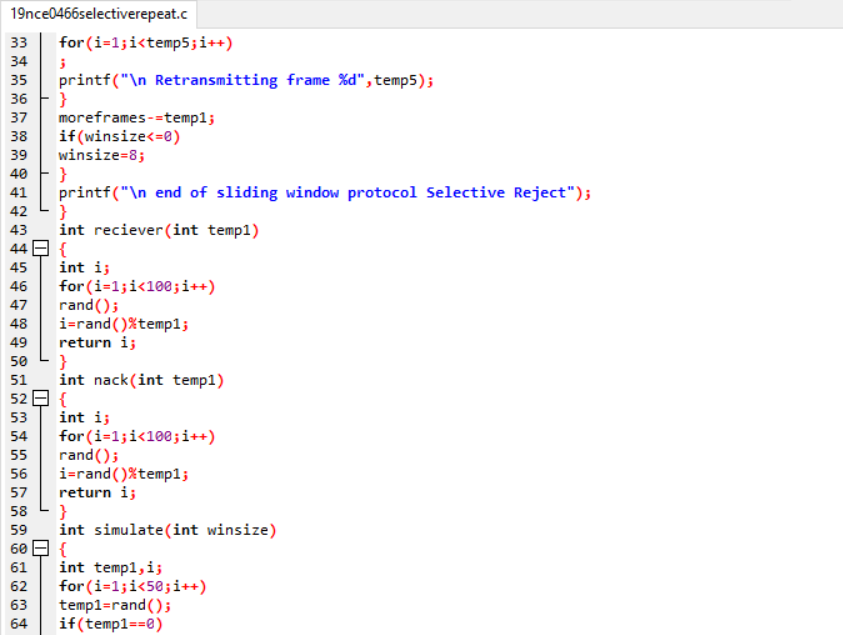
return winsize;

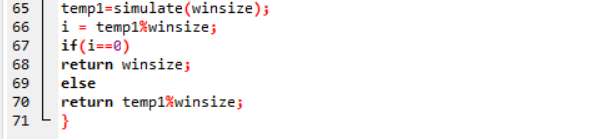
else

return temp1%winsize;

}







**Output:**

