1. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.

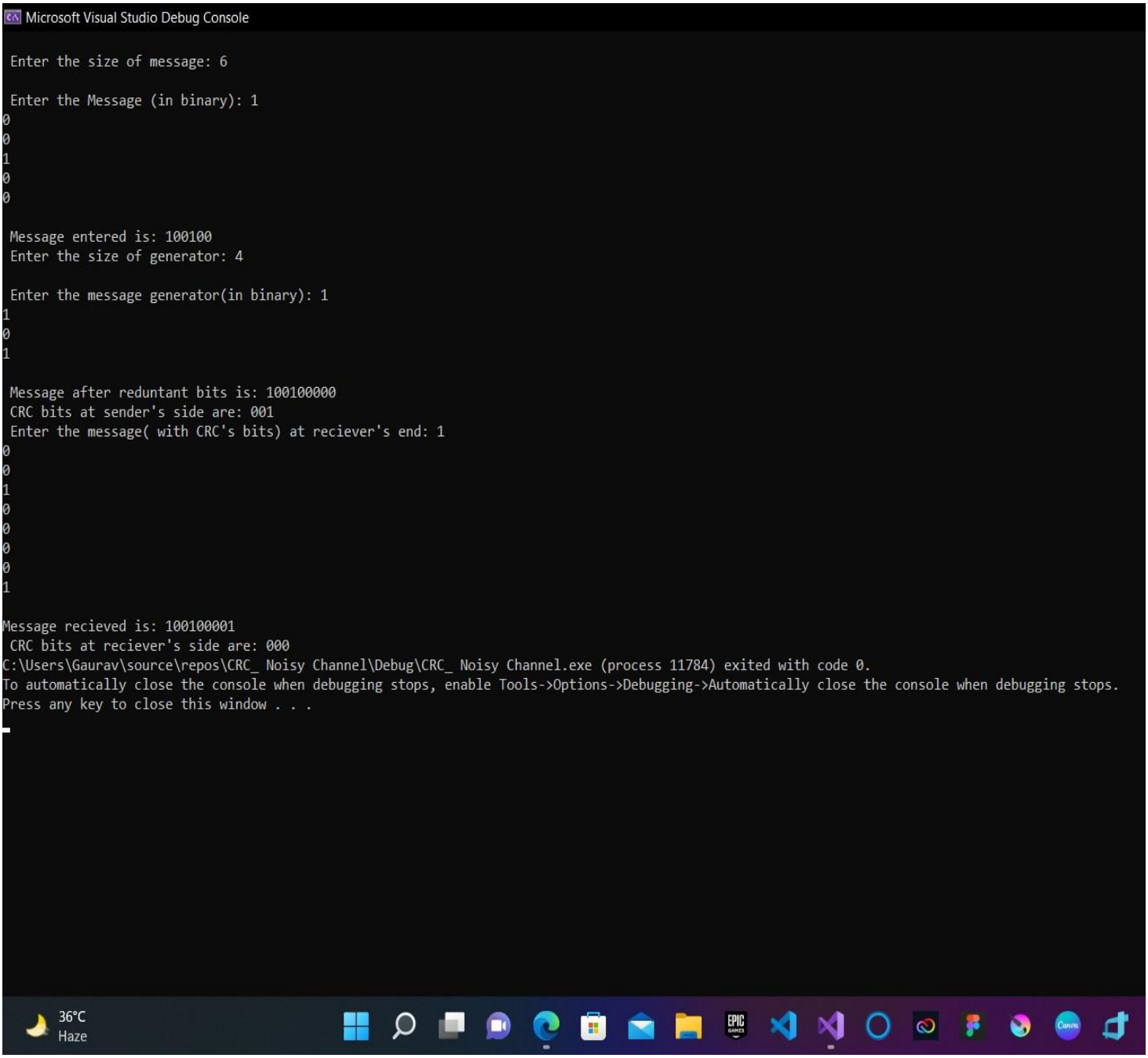
```
#include<stdio.h>
#include<iostream>
#include<conio.h>
using namespace std;



int main()
{
int n, m;
int i;
cout << "\n Enter the size of message: ";
cin >> n;



int arr[100], temp[10];
cout << "\n Enter the Message (in binary): ";

for (i = 0; i < n; i++)
{
cin >> arr[i];
}

for (i = 0; i < n; i++)
{
temp[i] = arr[i];
}

cout<<"\n Message entered is: ";
for (int i = 0; i < n; i++)
{
cout << arr[i];
```

```
}

cout << "\n Enter the size of generator: ";
cin >> m;

int arr1[100];

cout << "\n Enter the message generator(in binary): ";

for (int i = 0; i < m; i++)
{
cin >> arr1[i];
}
cout << "\n Message after reduntant bits is: ";

for (int i = n; i < (n + m - 1); i++)
{
arr[i] = 0;
}

for (int i = 0; i < (n + m - 1); i++)
{
cout << arr[i];
}

for (int i = 0; i < n; i++)
{
if (arr1[0] == arr[i])
{
for (int j = 0, k = i; j < m; j++, k++)
{
if (!(arr[k] ^ arr1[j]))        arr[k] = 0;
```

```
else       arr[k] = 1;
}
}
}



cout<<"\n CRC bits at sender's side are: ";
for (int i = n; i < (n + m - 1); i++)
{
cout << arr[i];
}

int arr2[100];
cout<<"\n Enter the message( with CRC's bits) at reciever's end: " ;

for (int i = 0; i < (n + m - 1); i++)
{ cin >> arr2[i]; }

cout<<"\nMessage recieved is: ";
for (int i = 0; i < (n + m - 1); i++)
{
cout<<arr2[i];
}

for (int i = 0; i < n; i++)
{
if (arr1[0] == arr2[i])
{
for (int j = 0, k = i; j < m; j++, k++)
{
if (!(arr2[k] ^ arr1[j]))
arr2[k] = 0;
```

```
else
arr2[k] = 1;
}
}
}

cout<<"\n CRC bits at reciever's side are: ";
for (int i = n; i < (n + m - 1); i++)
{
cout << arr2[i];
}

return 0;
}
```

OUTPUT:


Microsoft Visual Studio Debug Console
Enter the size of message: 6
Enter the Message (in binary): 1
0
0
1
0
0
Message entered is: 100100
Enter the size of generator: 4
Enter the message generator(in binary): 1
1
0
1
Message after reduntant bits is: 100100000
CRC bits at sender's side are: 001
Enter the message(with CRC's bits) at reciever's end: 1
0
0
1
0
0
0
0
1
Message recieved is: 100100001
CRC bits at reciever's side are: 000
C:\Users\Gaurav\source\repos\CRC_ Noisy Channel\Debug\CRC_ Noisy Channel.exe (process 11784) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
36°C
Haze

2. Simulate and implement stop and wait protocol for noisy channel.

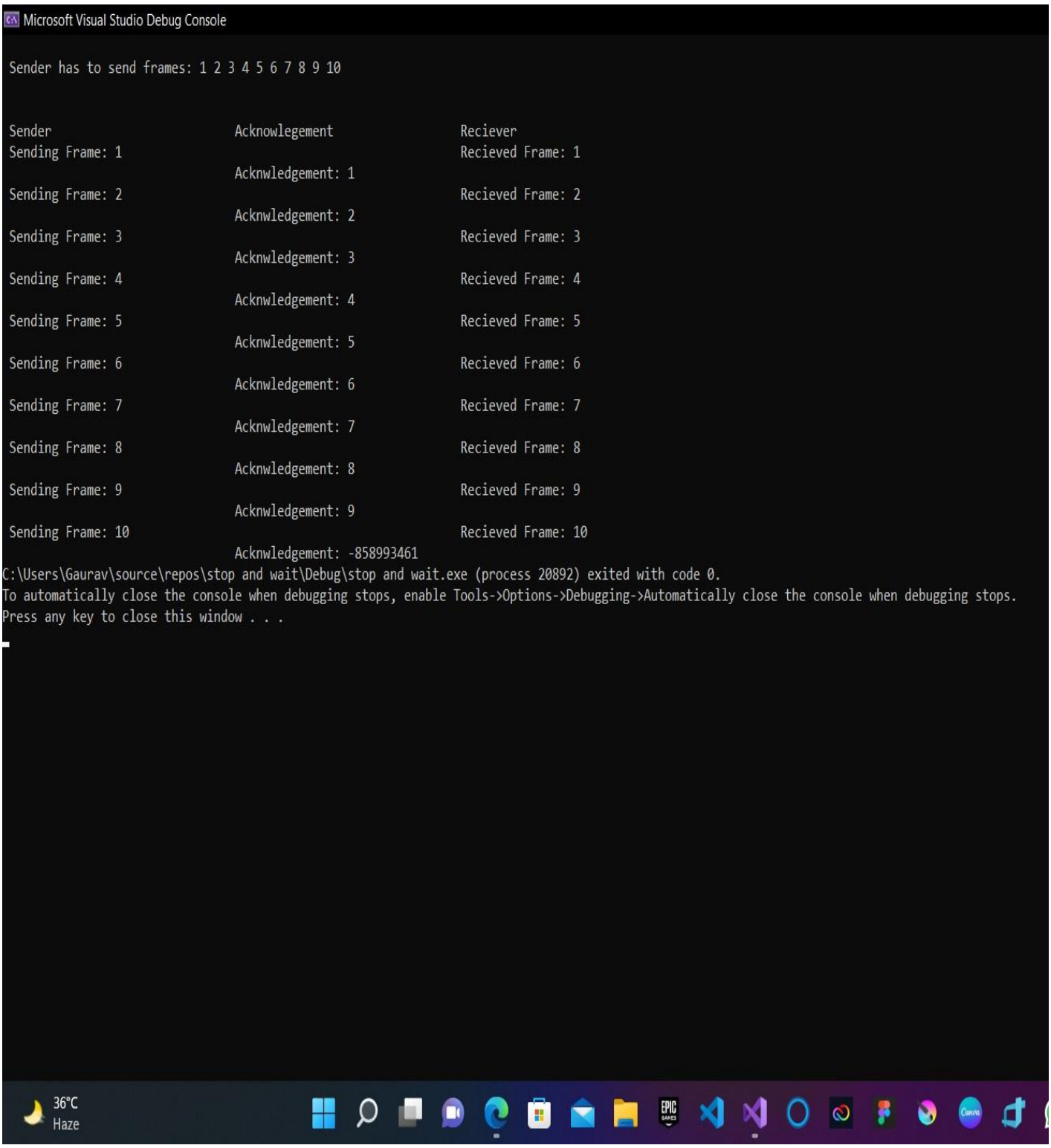
```
#include<iostream>
#include<time.h>
#include<cstdlib>
#include<ctime>
using namespace std;

class timer {
private:
unsigned long begtime;
public:
void start()
{
begtime = clock();
}
unsigned long elapsedtime()
{
return ((unsigned long)clock() - begtime) / CLOCKS_PER_SEC;
}

bool istimeout(unsigned long seconds)
{
return seconds >= elapsedtime();
}

};
```

```
int main()
{
int frames[] = { 1,2,3,4,5,6,7,8,9,10 };
unsigned long seconds = 5;
srand(time(NULL));
timer t;
cout << "\n Sender has to send frames: ";
for (int i = 0; i < 10; i++)
{
cout << frames[i] << " ";
}
cout << endl;
int count = 0;
bool delay = false;
cout << endl << "\n Sender \t\t\t Acknowlegement \t\t Reciever ";
while (count != 10)
{
bool timeout = false;
cout << "\n Sending Frame: " << frames[count];
cout.flush();
cout << "\t\t";
t.start();
if (rand() % 2)
{
int to = 24600 + rand() % (64000 - 24600) + 1;
for (int i = 0; i < 64000; i++)
{
for (int j = 0; j < to; j++)
```

```
{


}
}
}

if (t.elapsedtime() <= seconds)
{
cout << "\t\t\t\t Recieved Frame: " << frames[count] << " ";
if (delay)
{
cout << "\n Duplicate";
delay = false;

}
cout << "\n";
count++;
}

else
{
cout << "------- " << endl;
cout << "\n Timeout";
timeout = true;
}

t.start();
if (rand() % 2 || !timeout)
```

```
{
int to = 24600 + rand() % (64000 - 24600) + 1;
for (int i = 0; i < 64000; i++)
{
for (int j = 0; j < to; j++)
{


}
}
if (t.elapsedtime() > seconds)
{
cout << "\n Delayed Ack";
count--;
delay = true;
}

else if(!timeout)
{
cout << "\t\t\t\t Acknwledgement: " << frames[count] - 1;
}


}
}
return 0;
}
```

OUTPUT:


Microsoft Visual Studio Debug Console
Sender has to send frames: 1 2 3 4 5 6 7 8 9 10
Sender Acknowlegement Reciever
Sending Frame: 1 Recieved Frame: 1
Acknwledgement: 1
Sending Frame: 2 Recieved Frame: 2
Acknwledgement: 2
Sending Frame: 3 Recieved Frame: 3
Acknwledgement: 3
Sending Frame: 4 Recieved Frame: 4
Acknwledgement: 4
Sending Frame: 5 Recieved Frame: 5
Acknwledgement: 5
Sending Frame: 6 Recieved Frame: 6
Acknwledgement: 6
Sending Frame: 7 Recieved Frame: 7
Acknwledgement: 7
Sending Frame: 8 Recieved Frame: 8
Acknwledgement: 8
Sending Frame: 9 Recieved Frame: 9
Acknwledgement: 9
Sending Frame: 10 Recieved Frame: 10
Acknwledgement: -858993461
C:\Users\Gaurav\source\repos\stop and wait\Debug\stop and wait.exe (process 20892) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
36°C
Haze

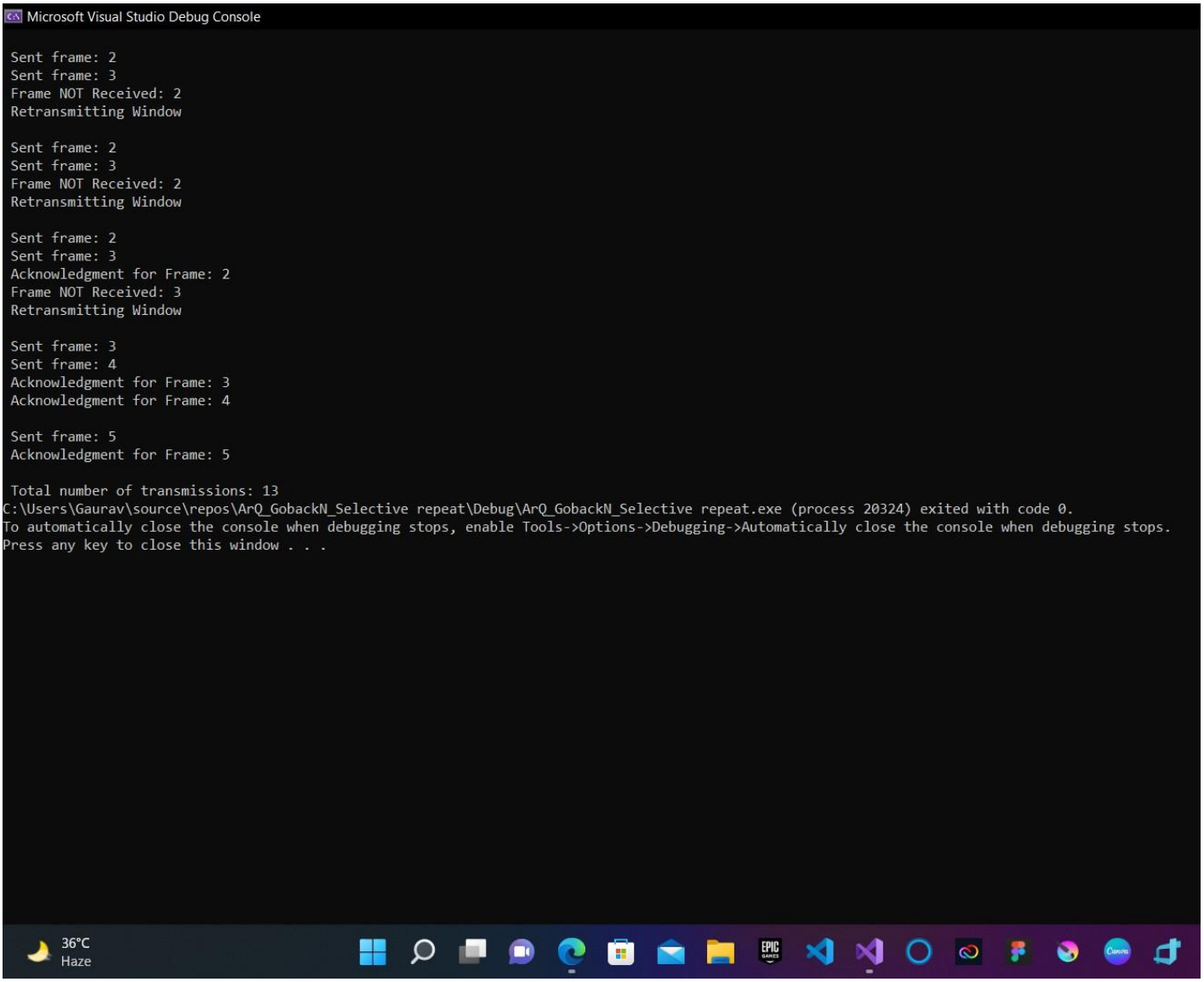
3. Simulate and implement go back N sliding window protocol.

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<iostream>
#include<conio.h>
using namespace std;

int main()
{
   int nf, N;
   int no_tr = 0;

   srand(time(NULL));

   cout<<"\n Enter no. of frames: ";
   cin>>nf;

   cout<<"\n Enter the Window size: ";
   cin>>N;

   cout << endl;

   int i = 1, j;
   while (i <= nf)
   {
      int x = 0;
```

```
    for (j = i; j < i + N && j <= nf; j++, no_tr++)
    {
        cout << "\n Sent frame: " << j;
    }

    for (j = i; j < i + N && j <= nf; j++)
    {
        int flag = rand() % 2;

        if (!flag)
        {
            cout<<"\n Acknowledgment for Frame: "<< j;
            x++;
        }
        else
        {
            cout<<"\n Frame NOT Received: "<< j;
            cout<<"\n Retransmitting Window";
            break;
        }
    }
    cout << endl;
    i += x;
}

cout<<"\n Total number of transmissions: "<< no_tr;

return 0;
```

}

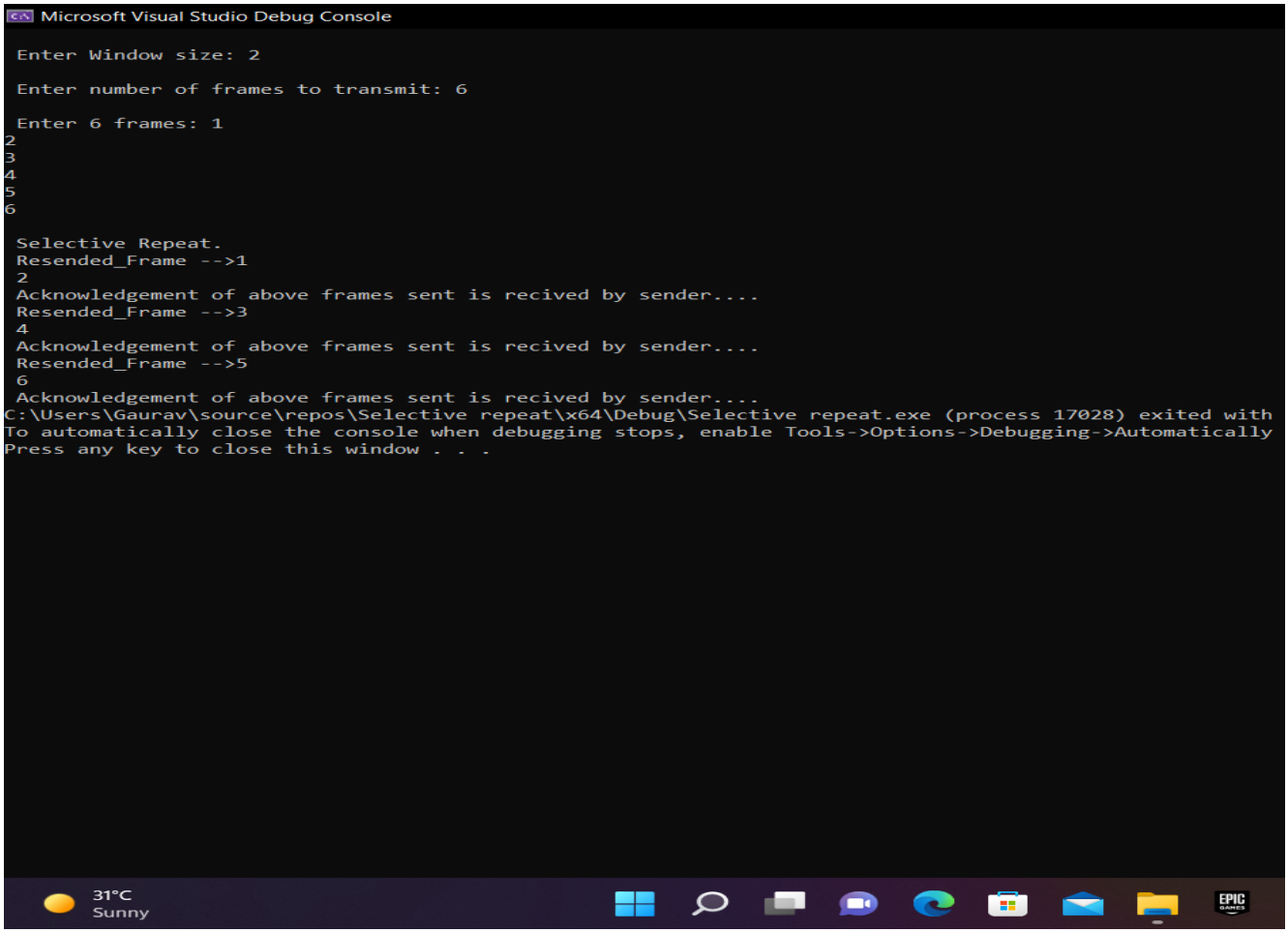
OUTPUT:


Microsoft Visual Studio Debug Console
Sent frame: 2
Sent frame: 3
Frame NOT Received: 2
Retransmitting Window
Sent frame: 2
Sent frame: 3
Frame NOT Received: 2
Retransmitting Window
Sent frame: 2
Sent frame: 3
Acknowledgment for Frame: 2
Frame NOT Received: 3
Retransmitting Window
Sent frame: 3
Sent frame: 4
Acknowledgment for Frame: 3
Acknowledgment for Frame: 4
Sent frame: 5
Acknowledgment for Frame: 5
Total number of transmissions: 13
C:\Users\Gaurav\source\repos\ArQ_GobackN_Selective repeat\Debug\ArQ_GobackN_Selective repeat.exe (process 20324) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
36°C
Haze

4. Simulate and implement selective repeat sliding window protocol.

```
#include<iostream>
#include<stdlib.h>
#include<conio.h>
using namespace std;

void Selective_repeat()
{
int w, i, frames[50],f;

cout << "\n Enter Window size: ";
cin >> w;

cout << "\n Enter number of frames to transmit: ";
cin >> f;

cout << "\n Enter " << f << " frames: ";

for (i = 1; i <= f; i++)
{
cin >> frames[i];
}

cout << "\n Selective Repeat.";

for (i = 1; i <= f; i++)
{
if (i % w == 0)
{
cout << "\n " << frames[i];
cout << "\n Acknowledgement of above frames sent is recived by sender....";
}
else if(i%w!=0)
{
cout << "\n Resended_Frame -->" << frames[i];
}

}

if (f % w != 0)
{
cout << "\n Acknowledgement of above frames is recieved by sender...";
}

}

int main()
{
Selective_repeat();
return 0;

}
```

OUTPUT:


Microsoft Visual Studio Debug Console
Enter Window size: 2
Enter number of frames to transmit: 6
Enter 6 frames: 1
2
3
4
5
6
Selective Repeat.
Resended_Frame -->1
2
Acknowledgement of above frames sent is recived by sender....
Resended_Frame -->3
4
Acknowledgement of above frames sent is recived by sender....
Resended_Frame -->5
6
Acknowledgement of above frames sent is recived by sender....
C:\Users\Gaurav\source\repos\Selective repeat\x64\Debug\Selective repeat.exe (process 17028) exited with
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically
Press any key to close this window . . .
31°C
Sunny

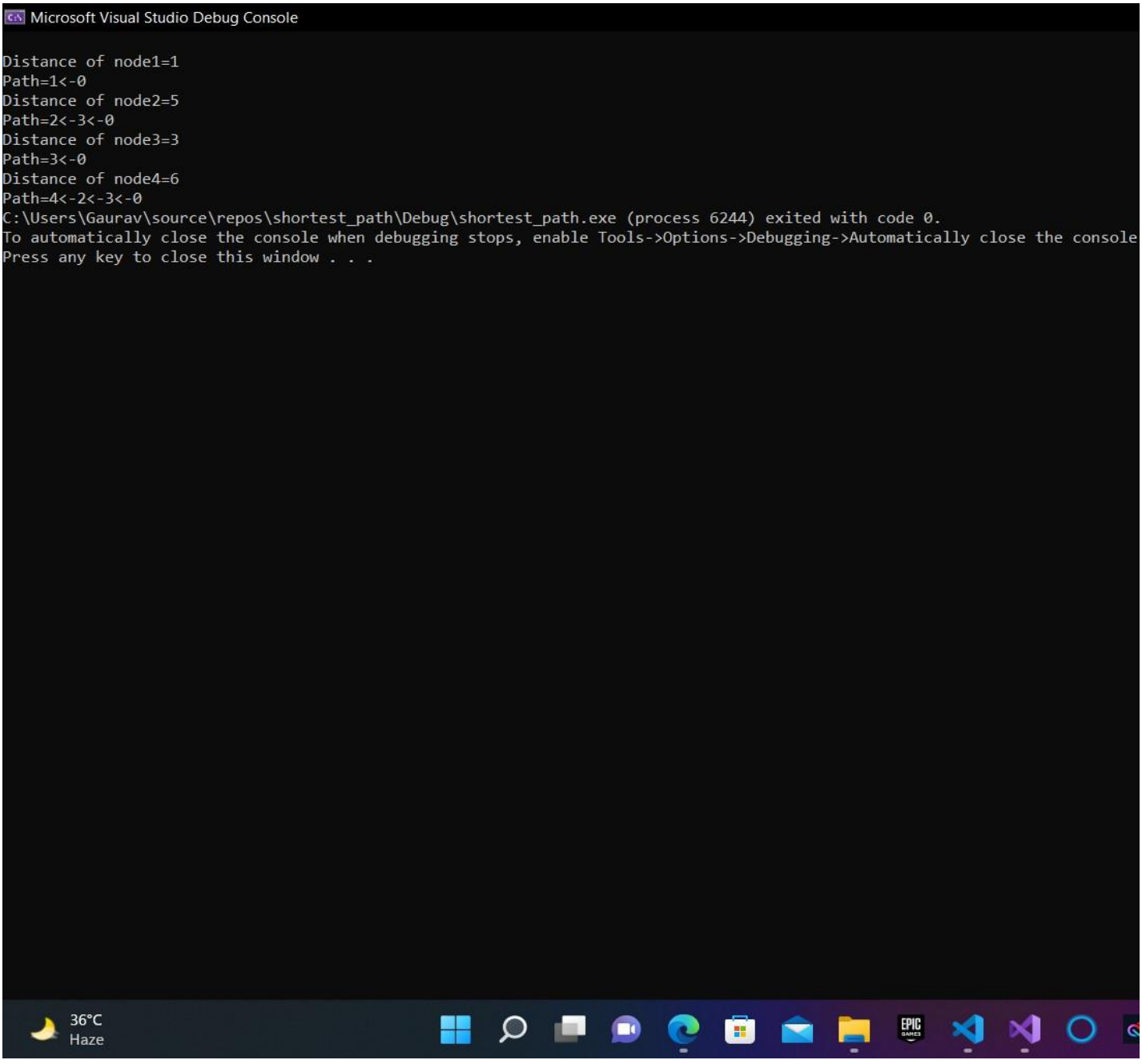
5. Shortest Path algorithm.

```
#include<iostream>
#include<stdio.h>
using namespace std;
#define INFINITY 9999
#define max 5
void shortest_path(int G[max][max], int n, int startnode);
int main() {
   int G[max][max] = { {0,1,0,3,10},{1,0,5,0,0},{0,5,0,2,1},{3,0,2,0,6},{10,0,1,6,0} };
   int n = 5;
   int u = 0;
   shortest_path(G, n, u);
   return 0;
}
void shortest_path(int G[max][max], int n, int startnode) {
   int cost[max][max], distance[max], pred[max];
   int visited[max], count, mindistance, nextnode, i, j;
   for (i = 0; i < n; i++)
      for (j = 0; j < n; j++)
```

```
        if (G[i][j] == 0)
            cost[i][j] = INFINITY;
        else
            cost[i][j] = G[i][j];
for (i = 0; i < n; i++) {
    distance[i] = cost[startnode][i];
    pred[i] = startnode;
    visited[i] = 0;
}
distance[startnode] = 0;
visited[startnode] = 1;
count = 1;
while (count < n - 1) {
    mindistance = INFINITY;
    for (i = 0; i < n; i++)
        if (distance[i] < mindistance && !visited[i]) {
            mindistance = distance[i];
            nextnode = i;
        }
    visited[nextnode] = 1;
    for (i = 0; i < n; i++)
        if (!visited[i])
            if (mindistance + cost[nextnode][i] < distance[i]) {
                distance[i] = mindistance + cost[nextnode][i];
                pred[i] = nextnode;
```

```
            }
        count++;
    }
    for (i = 0; i < n; i++)
        if (i != startnode) {
            cout << "\nDistance of node" << i << "=" << distance[i];
            cout << "\nPath=" << i;
            j = i;
            do {
                j = pred[j];
                cout << "<-" << j;
            } while (j != startnode);
        }
}
```

OUTPUT:


Microsoft Visual Studio Debug Console
Distance of node1=1
Path=1<-0
Distance of node2=5
Path=2<-3<-0
Distance of node3=3
Path=3<-0
Distance of node4=6
Path=4<-2<-3<-0
C:\Users\Gaurav\source\repos\shortest_path\Debug\shortest_path.exe (process 6244) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console
Press any key to close this window . . .
36°C
Haze