Computer Networks

Implementation of Go Back N protocol

Submitted to: Prof. Hossam Fahmy

2019/2020

Cairo, Egypt

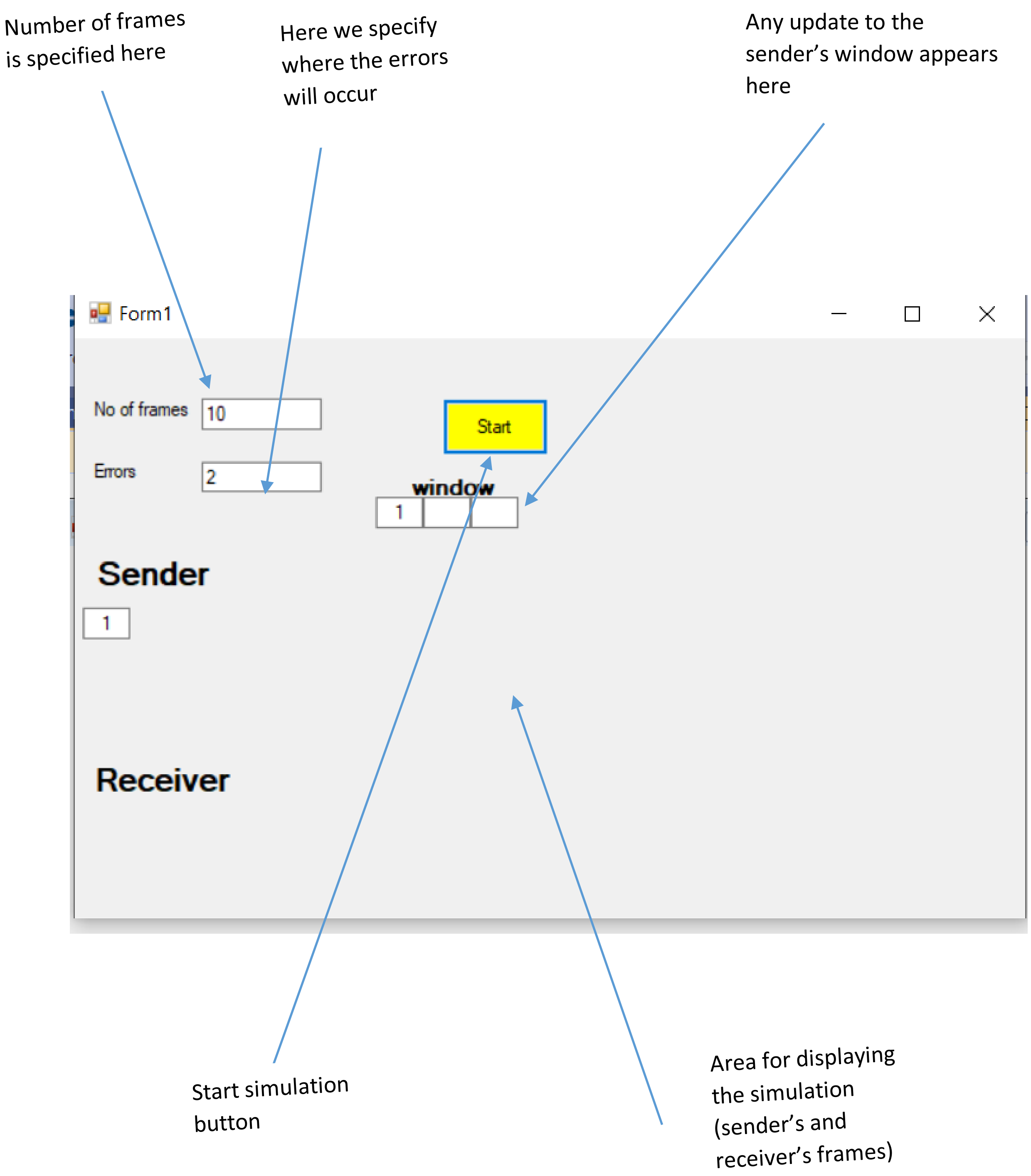
Student code	Team Members’ Names
15T0118	عماد مصرى ابراهيم بباوى
1501093	مارينا جرجس شكرى جرجس
1501580	ندى ايهاب احمد محمد عبدالجواد
1400004	ابانوب ميخائيل عيسى بقطر

Go Back N Protocol

We’ve made two versions for the GUI, as we wanted to make the GUI dynamic (sending and receiving is done one by one in front of the user) but to make this we had to make the sender’s window size fixed, so we thought of making another static version of variable sender’s window size.

GUI

Version 1

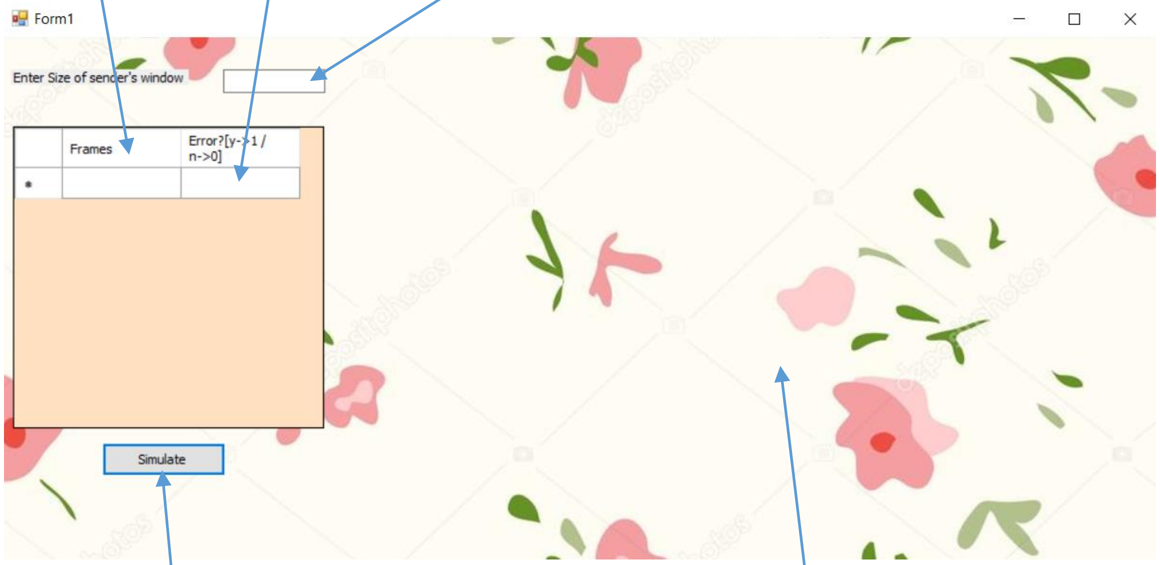

Number of frames is specified here
Here we specify where the errors will occur
Any update to the sender's window appears here
Form1
No of frames
10
Errors
2
Start
window
1
Sender
1
Receiver
Start simulation button
Area for displaying the simulation (sender's and receiver's frames)

Version 2

Frames coming from the Network Layer are entered here

Here we specify where the errors will occur

Here we specify the size of the sender's window



Start simulation button

Area for displaying the simulation (sender's and receiver's frames)

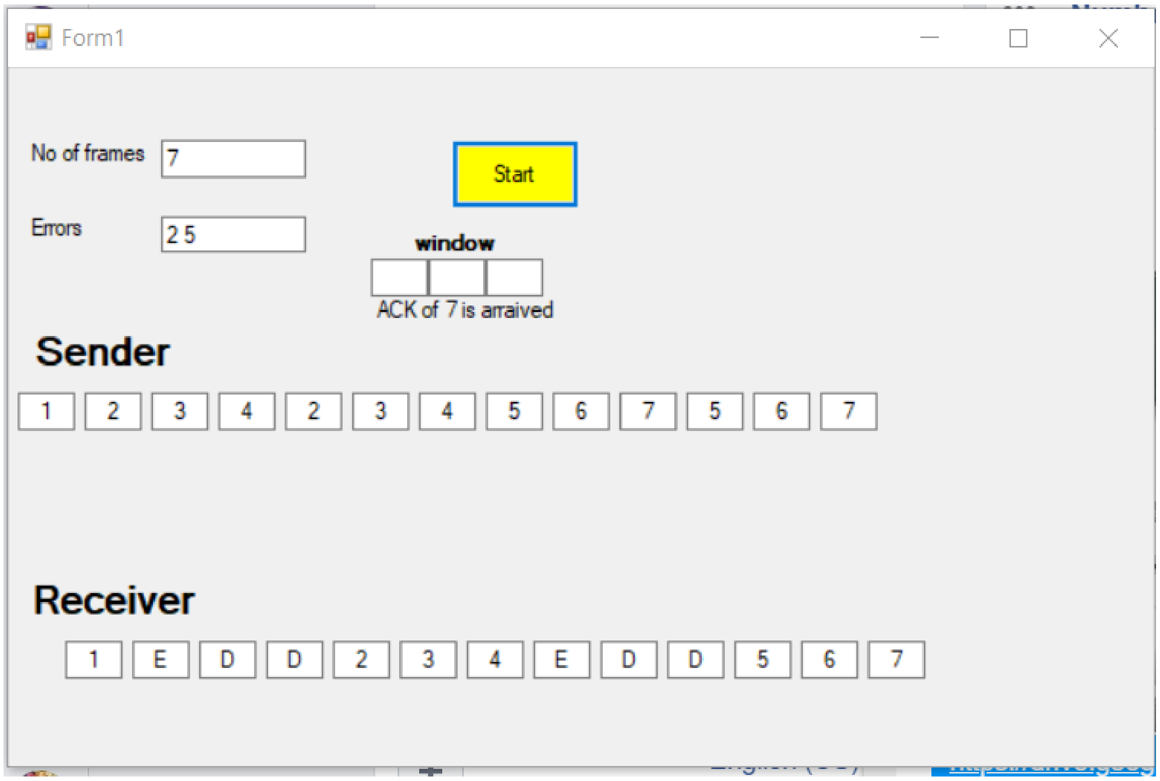
Test cases snapshots

Version 1 (the dynamic version)

Video demonstrating the GUI (and a test case)

CLICK HERE

Test case 2



No of frames 7

Errors 2 5

ACK of 7 is arraived

Sender: 1 2 3 4 2 3 4 5 6 7 5 6 7

Receiver: 1 E D D 2 3 4 E D D 5 6 7

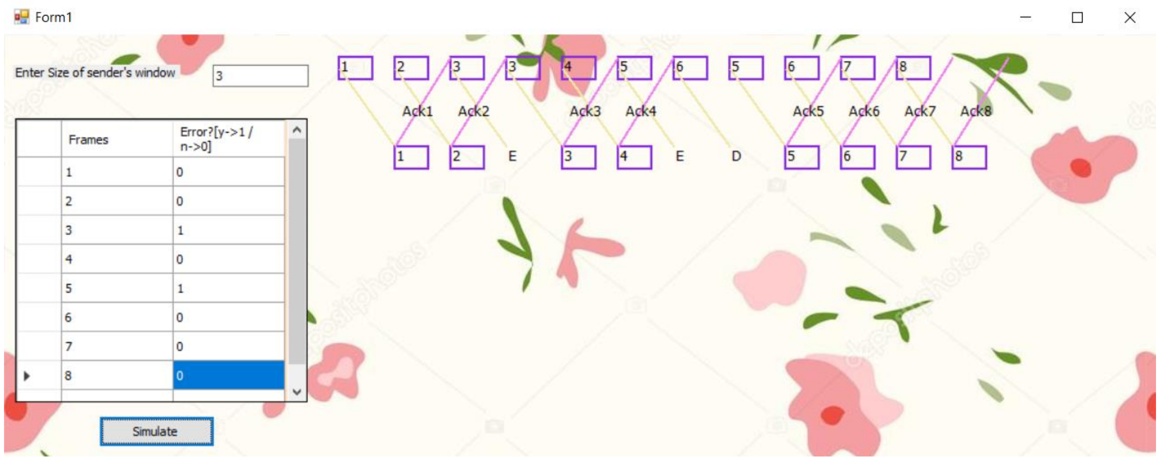
Version 2

Video demonstrating the GUI (and a test case)

CLICK HERE

Test case 2

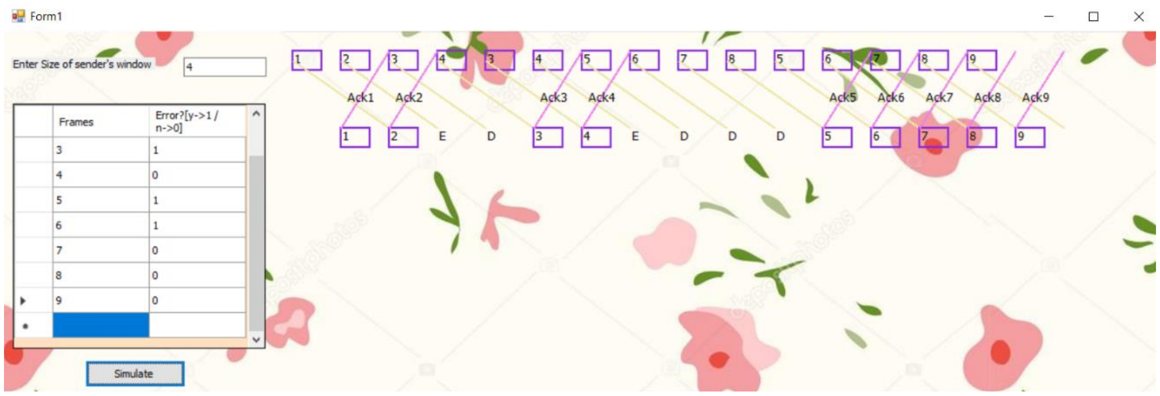
In the errors section, ‘1’ means an error occurred, and ‘0’ means no error occurred.



Enter Size of sender's window: 3

Frames	Error?[y->1 / n->0]
1	0
2	0
3	1
4	0
5	1
6	0
7	0
8	0

Test case 2

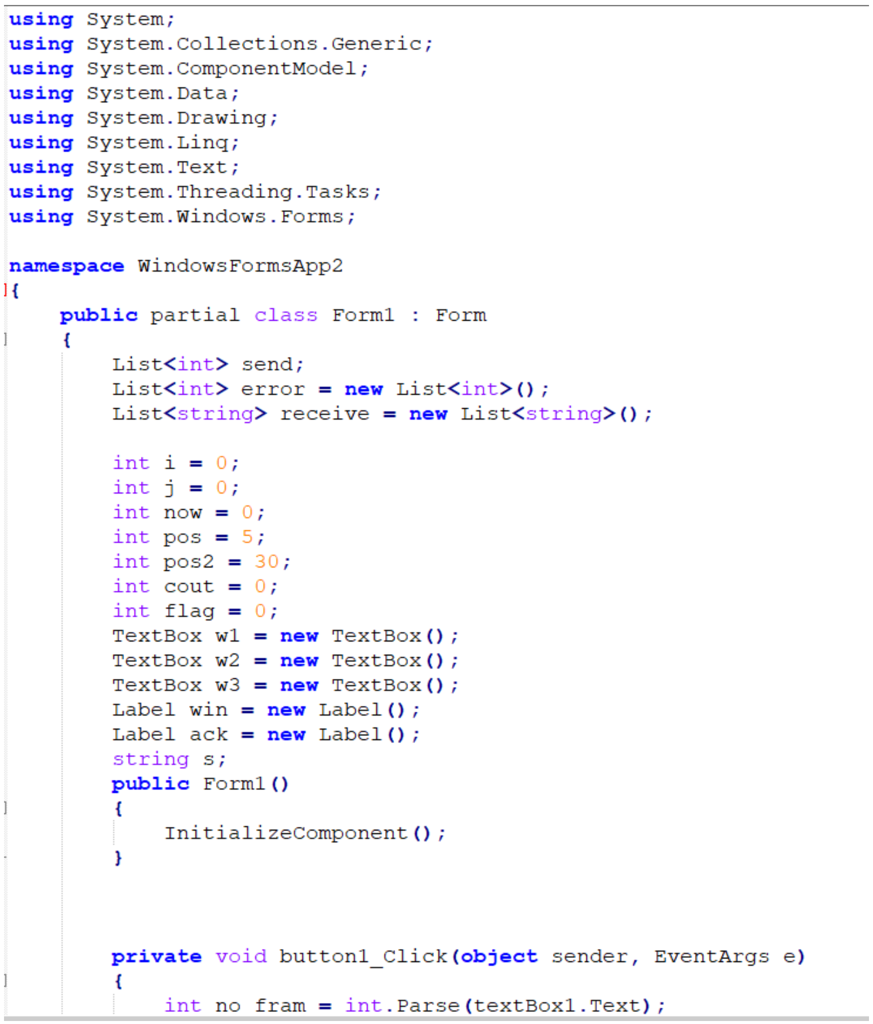

Form1
Simulate
1 2 3 4 3 4 5 6 7 8 5 6 7 8 9
Ack1 Ack2 Ack3 Ack4 Ack5 Ack6 Ack7 Ack8 Ack9
1 2 E D 3 4 E D D D 5 6 7 8 9

Enter Size of sender's window: 4

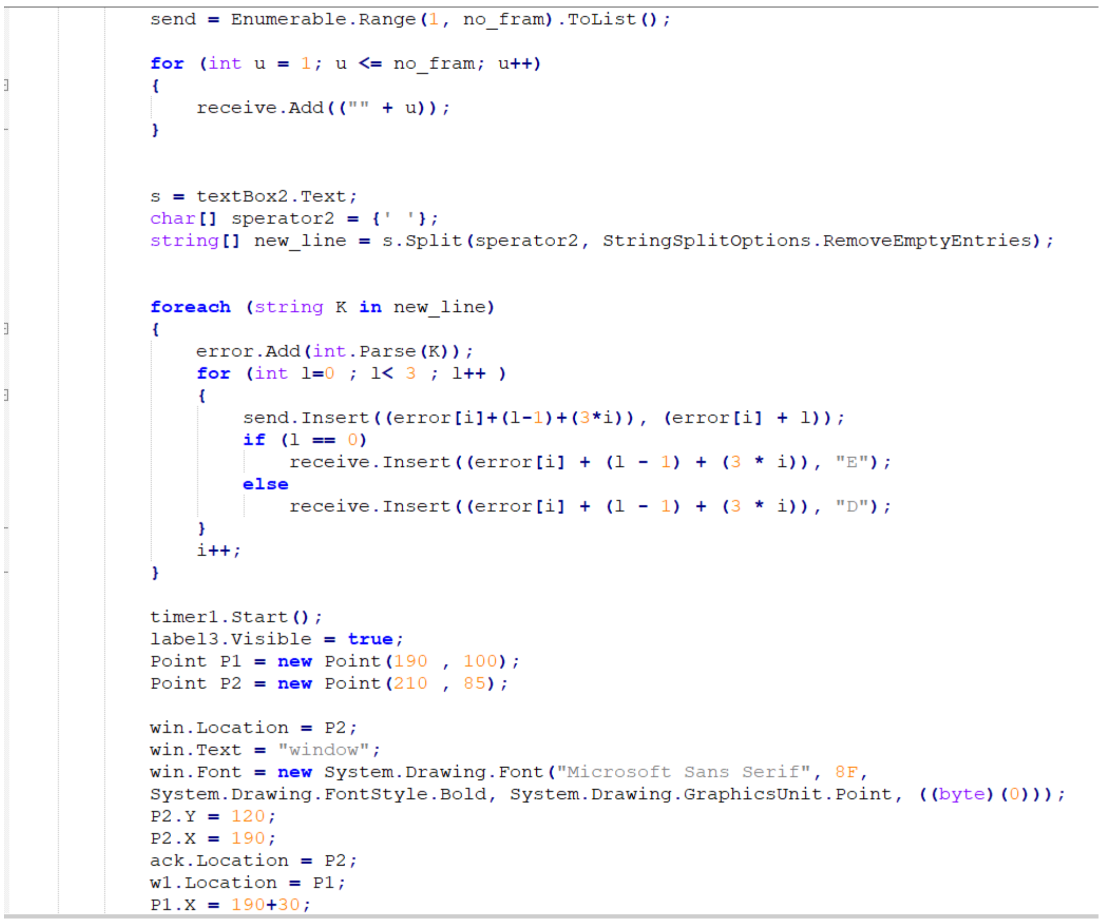
Frames	Error?[y->1 / n->0]
3	1
4	0
5	1
6	1
7	0
8	0
9	0

Code

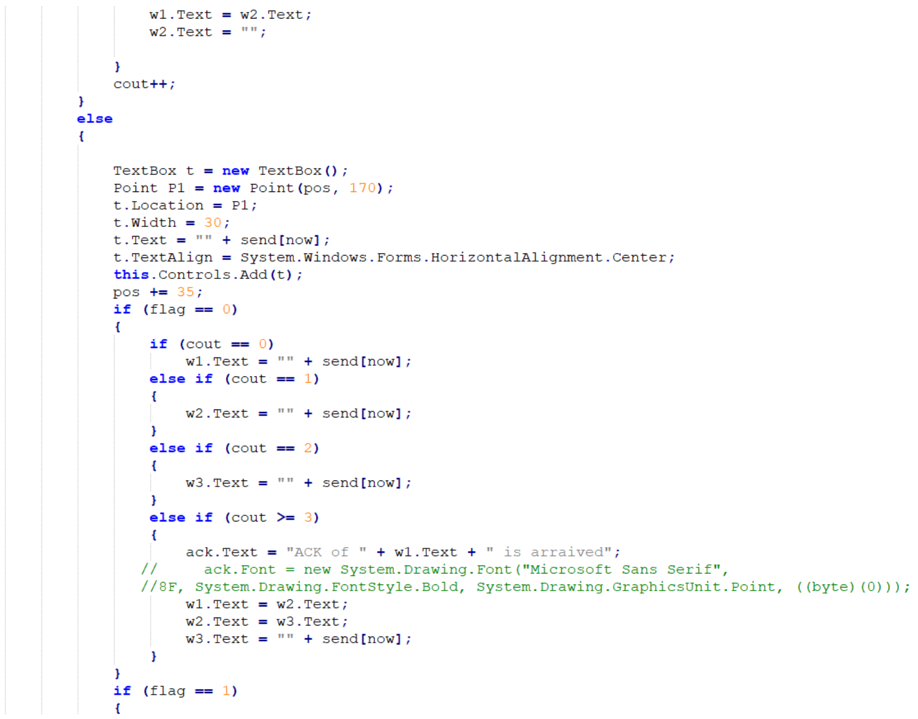
Version 1



```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp2
{
    public partial class Form1 : Form
    {
        List<int> send;
        List<int> error = new List<int>();
        List<string> receive = new List<string>();

        int i = 0;
        int j = 0;
        int now = 0;
        int pos = 5;
        int pos2 = 30;
        int cout = 0;
        int flag = 0;
        TextBox w1 = new TextBox();
        TextBox w2 = new TextBox();
        TextBox w3 = new TextBox();
        Label win = new Label();
        Label ack = new Label();
        string s;
        public Form1()
        {
            InitializeComponent();
        }



        private void button1_Click(object sender, EventArgs e)
        {
            int no fram = int.Parse(textBox1.Text);
```



```
send = Enumerable.Range(1, no_fram).ToList();

for (int u = 1; u <= no_fram; u++)
{
    receive.Add(("" + u));
}


s = textBox2.Text;
char[] sperator2 = {' '};
string[] new_line = s.Split(sperator2, StringSplitOptions.RemoveEmptyEntries);


foreach (string K in new_line)
{
    error.Add(int.Parse(K));
    for (int l=0 ; l< 3 ; l++ )
    {
        send.Insert((error[i]+(l-1)+(3*i)), (error[i] + l));
        if (l == 0)
            receive.Insert((error[i] + (l - 1) + (3 * i)), "E");
        else
            receive.Insert((error[i] + (l - 1) + (3 * i)), "D");
    }
    i++;
}

timer1.Start();
label3.Visible = true;
Point P1 = new Point(190 , 100);
Point P2 = new Point(210 , 85);

win.Location = P2;
win.Text = "window";
win.Font = new System.Drawing.Font("Microsoft Sans Serif", 8F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
P2.Y = 120;
P2.X = 190;
ack.Location = P2;
w1.Location = P1;
P1.X = 190+30;
```



```
            w1.Text = w2.Text;
            w2.Text = "";

        }
        cout++;
    }
    else
    {

        TextBox t = new TextBox();
        Point P1 = new Point(pos, 170);
        t.Location = P1;
        t.Width = 30;
        t.Text = "" + send[now];
        t.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
        this.Controls.Add(t);
        pos += 35;
        if (flag == 0)
        {
            if (cout == 0)
                w1.Text = "" + send[now];
            else if (cout == 1)
            {
                w2.Text = "" + send[now];
            }
            else if (cout == 2)
            {
                w3.Text = "" + send[now];
            }
            else if (cout >= 3)
            {
                ack.Text = "ACK of " + w1.Text + " is arraived";
          //      ack.Font = new System.Drawing.Font("Microsoft Sans Serif",
          //8F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
                w1.Text = w2.Text;
                w2.Text = w3.Text;
                w3.Text = "" + send[now];
            }
        }
        if (flag == 1)
        {
```

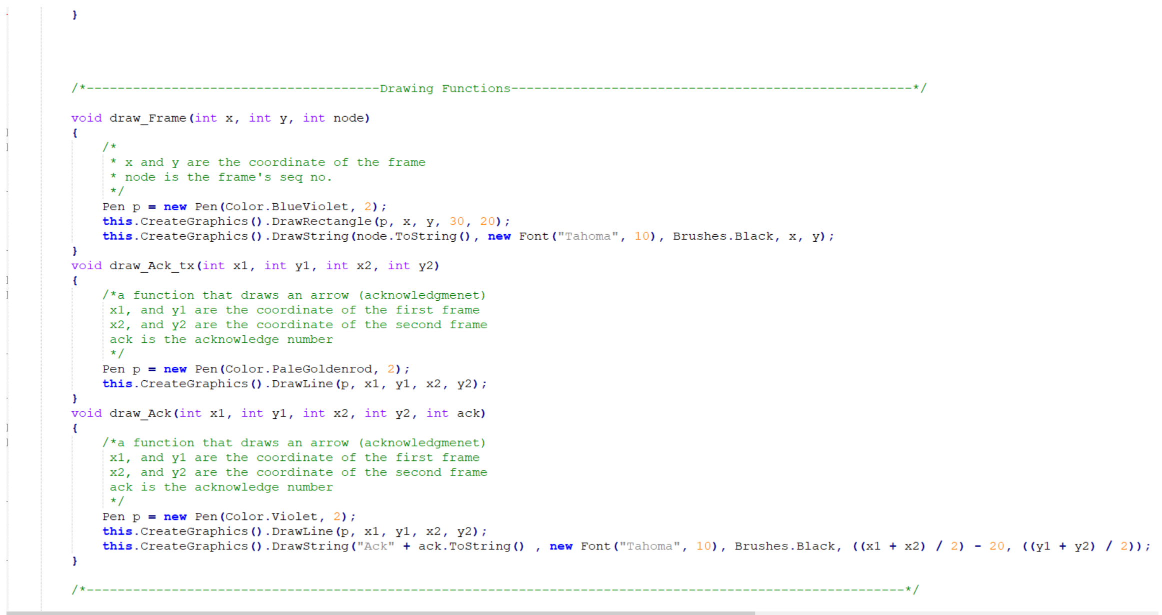
```
        w1.Text = w2.Text;
        w2.Text = "";

    }
    cout++;
}
else
{

    TextBox t = new TextBox();
    Point P1 = new Point(pos, 170);
    t.Location = P1;
    t.Width = 30;
    t.Text = "" + send[now];
    t.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
    this.Controls.Add(t);
    pos += 35;
    if (flag == 0)
    {
        if (cout == 0)
            w1.Text = "" + send[now];
        else if (cout == 1)
        {
            w2.Text = "" + send[now];
        }
        else if (cout == 2)
        {
            w3.Text = "" + send[now];
        }
        else if (cout >= 3)
        {
            ack.Text = "ACK of " + w1.Text + " is arraived";
       //      ack.Font = new System.Drawing.Font("Microsoft Sans Serif",
       //8F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            w1.Text = w2.Text;
            w2.Text = w3.Text;
            w3.Text = "" + send[now];
        }
    }
    if (flag == 1)
    {
```

```
                    if (cout == 1)
                    {
                        ack.Text = "ACK of " + w1.Text + " is arraived";
                   //     ack.Font = new System.Drawing.Font("Microsoft Sans Serif",
                   //8F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
                        w1.Text = w2.Text;
                        w2.Text = w3.Text;
                        w3.Text = "";

                        flag = 0;
                    }
                }
                if (w1.Text.Contains(s) && flag == 0)
                {
                    cout = -2;
                    flag = 1;
                }
                cout++;
                now++;

                if (now == 1)
                {
                    timer2.Start();
                    label4.Visible = true;

                }
                if (now == send.Count())
                {
                    ack.Text = "ACK of " + w1.Text + " is arraived";
                //    ack.Font = new System.Drawing.Font("Microsoft Sans Serif",
                //8F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
                    w1.Text = w2.Text;
                    w2.Text = w3.Text;
                    w3.Text = "";
                    cout = 0;
                }
            }
        }

        private void timer2_Tick(object sender, EventArgs e)
        {
            TextBox t = new TextBox();
            Point P1 = new Point(pos2, 300);
            t.Location = P1;
            t.Width = 30;
            t.Text = "" + receive[j];
            t.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
            this.Controls.Add(t);
            pos2 += 35;

            j++;
            if (j == receive.Count())
            {
                timer2.Stop();
            }
        }
    }
}
```

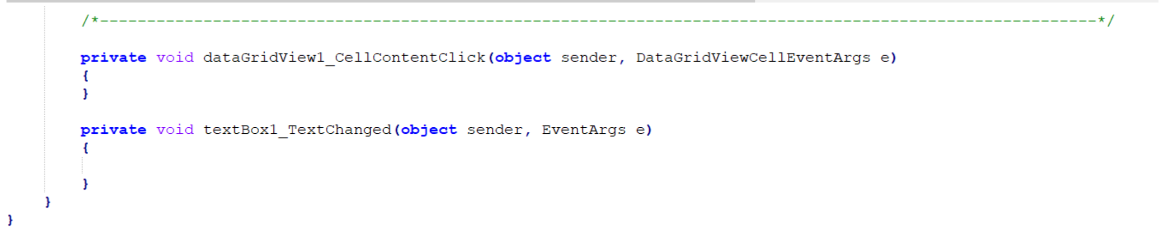
Version 2

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Networks
{
    public partial class Form1 : Form
    {
        /*Frame's location (coordinates)*/
        // 16 is an arbitary no.
        int[,] TX_frame_location = new int[16, 2] { { 300, 20 }, { 350, 20 }, { 400, 20 },{ 450, 20 },
                                                { 500, 20 }, { 550, 20 }, { 600, 20 }, { 650, 20 }, { 700, 20 },
                                                { 750, 20 }, { 800, 20 }, { 850, 20 }, { 900, 20 }, {950, 20 },
                                                {1000, 20 }, { 1050, 20} };
        int[,] TX_ack_location = new int[16, 2] { { 300, 30 }, { 350, 30 }, { 400, 30 }, { 450, 30 },
                                                { 500, 30 }, { 550, 30 }, { 600, 30 }, { 650, 30 }, { 700, 30 },
                                                { 750, 30 }, { 800, 30 }, { 850, 30 }, { 900, 30 }, { 950, 30 },
                                                { 1000, 30 }, { 1050, 30 } };
        int[,] RX_frame_location = new int[16, 2] { { 350, 100 }, { 400, 100 }, { 450, 100 }, { 500, 100 },
                                                { 550, 100 }, { 600, 100 }, { 650, 100 }, { 700, 100 }, { 750, 100 },
                                                { 800, 100 }, { 850, 100 }, { 900, 100 },{ 950, 100}, { 1000, 100},
                                                { 1050, 100}, { 1100, 100} };

        /*Data Grid View*/
        DataTable frames_table = new DataTable();
        int count_ack_tx = 0;
        int count_ack_rx = 0;
        int count_tx = 0;
        int count_rx = 0;

        public Form1()
        {
            InitializeComponent();
        }
```

```
private void Form1_Load(object sender, EventArgs e)
{
    /*Data Grid View,  acts as network layer,  hya manba3 el frames :"D */
    frames_table.Columns.Add("Frames", typeof(int));
    frames_table.Columns.Add("Error?[y->1 / n->0]", typeof(int));
    dataGridView1.DataSource = frames_table;

}

private void button1_Click(object sender, EventArgs e)
{

    List<int> frame_array = new List<int>();
    List<int> error_array = new List<int>();
    List<int> receiver_frames = new List<int>();

    for (int i = 0; i < frames_table.Rows.Count; i++)
    {
        //frame number
        var tx = dataGridView1.Rows[i].Cells[0].Value; // take value of cell
        string tx_a;
        tx_a = tx.ToString(); // change it to string
        int fr = int.Parse(tx_a);
        frame_array.Add(fr);

        //error condition (y -> 1, n -> 0)
        var cond = dataGridView1.Rows[i].Cells[1].Value; // take value of cell
        string condition;
        condition = cond.ToString(); // change it to string
        int state = int.Parse(condition);
        error_array.Add(state);
    }

    int window_size = int.Parse(textBox1.Text); // take sender's window size from the textbox

    List<List<int>> sender_window = new List<List<int>>(); // seq number + timer of each frame

    int counter = 0;
    int fr_count = frames_table.Rows.Count;
    int count_wind = 0;
    int error = 0;
    int one = 0;
    int d = 0;
    int receiver_lenght = 0;
    while (fr_count > 0)
    {

        if (fr_count < window_size)
        {
            window_size = fr_count;
        }
        for (int i = 0; i < window_size; i++)
        {
            // take frames from the network layer 3ala ad el sender's window then send them
            // 0 -> timer
            sender_window.Add(new List<int> { frame_array[count_wind], 0, error_array[count_wind] });

            if(error_array[count_wind] == 1 && one == 0)
            {
                // if an error occured in a frame, save frame's seq no. to retransmit it again
                // the "one" flag is used so that if another frame with an error exists, ignore it (take only first unacked frame)
                error = count_wind;
                one = 1;
            }
            if (error_array[count_wind] == 0 && d == 0)
            {
                receiver_frames.Add(frame_array[count_wind]);
                receiver_lenght++;
            }
            else if (error_array[count_wind] == 1 && d == 0)
            {
                // 100 stands for E -> error
                receiver_frames.Add(100);
                d = 1;
                receiver_lenght++;
            }
            else if (d == 1)
            {
                // 200 stands for discard D -> discard
                receiver_frames.Add(200);
                receiver_lenght++;
            }
        count_wind++;
        fr_count--;
    }
    int count = 0;

    //timer -> unused part
    for (int i = window_size; i > 0; i--)
    {
        sender_window[count][1] = i;
        count++;
    }

    // draw all frames in the sender's window even if they have errors
    for (int i = 0; i < window_size; i++)
    {
        draw_Frame(TX_frame_location[counter, 0], TX_frame_location[counter, 1], sender_window[i][0]);
        draw_Ack_tx(TX_ack_location[counter, 0], TX_ack_location[counter, 1], RX_frame_location[count_ack_tx, 0],
                    RX_frame_location[count_ack_tx , 1]);
        counter++;
        count_ack_tx ++;
    }

    //clear sender's window to fill it again with the first unacked frame again
    //plus all the frames that comes after it, to retransmit them agian
    sender_window.Clear();
    int size = 0;

    if(error != 0) // enter this part only if there exists at least one frame with an error
    {
        // note that "error" contains the seq no. of the first unacked frame (the frame that has )

        for(int j = error; j < error + window_size - (error % window_size); j++)
        {
            sender_window.Add(new List<int> { frame_array[j], 0, error_array[j] });
            receiver_frames.Add(frame_array[j]);
            receiver_lenght++;
            size++;
        }
    }
    // redraw them
    if(error != 0)
    {
        for(int i = 0; i < size; i++)
        {
            draw_Frame(TX_frame_location[counter, 0], TX_frame_location[counter, 1], sender_window[i][0]);
            draw_Ack_tx(TX_ack_location[counter, 0], TX_ack_location[counter, 1], RX_frame_location[count_ack_tx, 0],
            RX_frame_location[count_ack_tx, 1]);
            counter++;
            count_ack_tx++;


        }
    }

    d = 0;
    sender_window.Clear();
    error = 0;
    one = 0;
}

/*----------------------------------Receiver part--------------------------------------*/
for(int i = 0; i < receiver_lenght; i++)
{
    if(receiver_frames[i] == 100)
    {
        this.CreateGraphics().DrawString("E", new Font("Tahoma", 10), Brushes.Black, RX_frame_location[i, 0],
                                        RX_frame_location[i, 1]);
    }
    else if(receiver_frames[i] == 200)
    {
        this.CreateGraphics().DrawString("D", new Font("Tahoma", 10), Brushes.Black, RX_frame_location[i, 0],
                                        RX_frame_location[i, 1]);
    }
    else
    {
        draw_Frame(RX_frame_location[i, 0], RX_frame_location[i, 1], receiver_frames[i]);
        draw_Ack(RX_frame_location[i, 0], RX_frame_location[i, 1], RX_frame_location[i, 0] + 50, 20, receiver_frames[i]);
        count_ack_rx++;
    }
}
```



```
        }



        /*-------------------------------------Drawing Functions---------------------------------------------------*/

        void draw_Frame(int x, int y, int node)
        {
            /*
             * x and y are the coordinate of the frame
             * node is the frame's seq no.
             */
            Pen p = new Pen(Color.BlueViolet, 2);
            this.CreateGraphics().DrawRectangle(p, x, y, 30, 20);
            this.CreateGraphics().DrawString(node.ToString(), new Font("Tahoma", 10), Brushes.Black, x, y);
        }
        void draw_Ack_tx(int x1, int y1, int x2, int y2)
        {
            /*a function that draws an arrow (acknowledgmenet)
             x1, and y1 are the coordinate of the first frame
             x2, and y2 are the coordinate of the second frame
             ack is the acknowledge number
             */
            Pen p = new Pen(Color.PaleGoldenrod, 2);
            this.CreateGraphics().DrawLine(p, x1, y1, x2, y2);
        }
        void draw_Ack(int x1, int y1, int x2, int y2, int ack)
        {
            /*a function that draws an arrow (acknowledgmenet)
             x1, and y1 are the coordinate of the first frame
             x2, and y2 are the coordinate of the second frame
             ack is the acknowledge number
             */
            Pen p = new Pen(Color.Violet, 2);
            this.CreateGraphics().DrawLine(p, x1, y1, x2, y2);
            this.CreateGraphics().DrawString("Ack" + ack.ToString() , new Font("Tahoma", 10), Brushes.Black, ((x1 + x2) / 2) - 20, ((y1 + y2) / 2));
        }

        /*-----------------------------------------------------------------------------------------------------------*/
```



```
        /*-----------------------------------------------------------------------------------------------------*/

        private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {
        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
```