

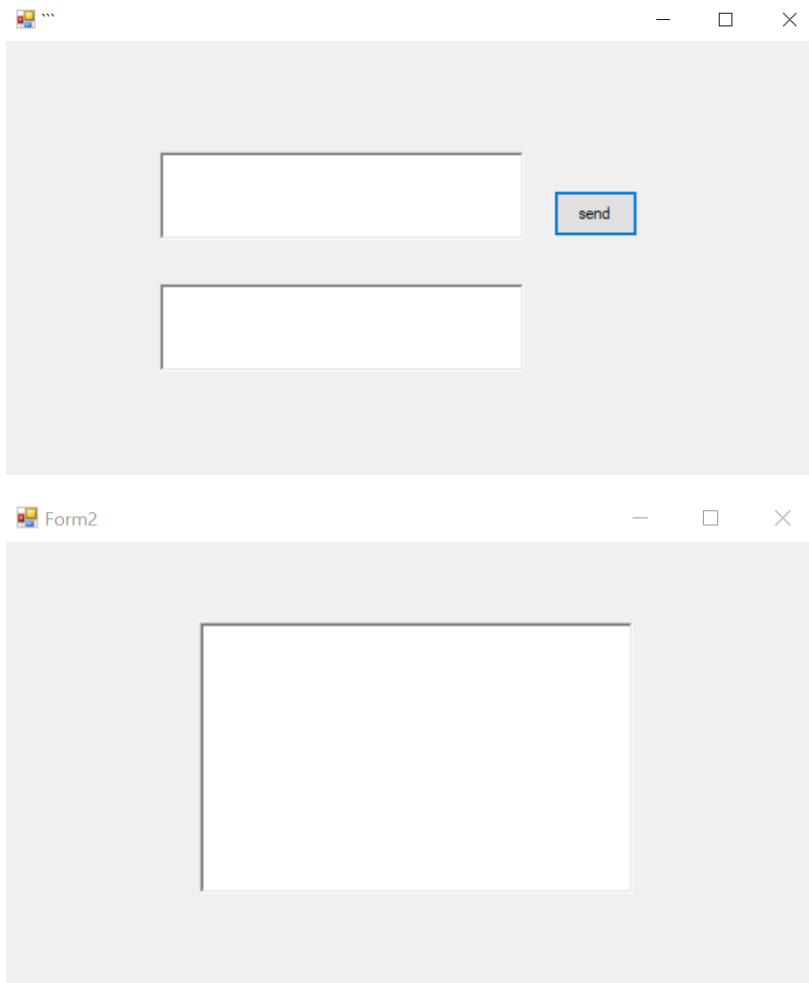
Zadanie 4

Jakub Giembicki 184755

Łukasz Czarzasty 184596

Temat: symulacja transmisji szeregowej zgodnej ze standardem RS232.

Program został wykonany w języku C#.



Opis:

Program pozwala na przestanie wiadomości między dwoma oknami. Wiadomość wpisywana jest w górnym Polu tekstowym pierwszego formularza. Po naciśnięciu przycisku „send” wiadomość zostaje zamieniona na ciąg bitów, który zostaje wyświetlony w drugim polu tekstowym tego okna. Następnie ciąg bitów zostaje przestany do drugiego okna i rozszyfrowany na znaki ASCII.

Form1

wiadomosc do wyslania

send

```
0011101111100110100111001100001110011001
0011001101111110011011011100110111111001
1100111100110001111000100000110011001001
100110111110001000001100111011111001111
0011100111001111001101100110011000011100
```

Form2

wiadomosc do wyslania

```

private void send_button_Click(object sender, EventArgs e)
{
    // Read text from send_textbox
    string textToSend = send_textbox.Text;
    textToSend = CensorText(textToSend);

    // Convert text to byte array
    byte[] byteArray = Encoding.UTF8.GetBytes(textToSend);

    // Convert each byte to bits and concatenate them
    string bitString = "";
    foreach (byte b in byteArray)
    {
        bitString += "0" + Convert.ToString(b, 2).PadLeft(8, '0') + "11";
    }

    // Display bits in receive_textbox
    receive_textbox.Text = bitString;
    form2.SetReceiveMessage(bitString);
}

```


Po wciśnięciu przycisku do wysyłania wiadomości, tekst zostaje zamieniony w ciąg bitów, z których każdy zaczyna się od bitu startu „0” i kończy na dwóch bitach stopu”11”.

```

public void SetReceiveMessage(string message)
{
    int charcode;
    string bitstring;
    char[] message_letter = "00000000".ToCharArray();
    char letter;
    for (int i = 0; i < message.Length/11; i++)
    {
        if (message[i * 11] == '0' && message[i * 11 + 9] == '1' && message[i * 11 + 10] == '1')
        {
            for (int j = 0; j < 8; j++)
            {
                message_letter[j] = message[i * 11 + 1 + j];
            }
            bitstring = new string(message_letter);
            charcode = Convert.ToInt32(bitstring, 2);
            letter = (char)charcode;
            textbox_receive_message.Text += letter;
        }
    }
}

```

Zakodowana wiadomość jest dekodowana bit po bicie i wyświetlana w drugim oknie.



przekleństwo

send

```

0001010101100010101011000101010110001010101
1000101010110001010101100010101011000101010
1100010101011000101010110001010101100010101
011
  
```

```

private void LoadCensoredWords()
{
    censoredWords = new List<string>();

    try
    {
        using (StreamReader sr = new StreamReader("cenzura.txt"))
        {
            string line;
            while ((line = sr.ReadLine()) != null)
            {
                censoredWords.Add(line.Trim());
            }
            foreach (string word in censoredWords)
            {
                MessageBox.Show(word);
            }
        }
    }
    catch (Exception e)
    {
        MessageBox.Show("Could not read the file 'cenzura.txt': " + e.Message);
    }
}

```

Wczytywanie słów ze słownika wulgaryzmów.

```

private string CensorText(string text)
{
    StringBuilder censoredText = new StringBuilder(text);

    foreach (string word in censoredWords)
    {
        int index = 0;
        while ((index = text.IndexOf(word, index, StringComparison.OrdinalIgnoreCase)) != -1)
        {
            for (int i = 0; i < word.Length; i++)
            {
                censoredText[index + i] = '*';
            }
            index += word.Length;
        }
    }

    return censoredText.ToString();
}

```

Cenzura zakazanych słów.

Wady:

Brak obsługi polskich znaków.