```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Threading;
using System.Net;
namespace SrvUDPMSG
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        delegate void ShowMessage(String Message);

        public void Show(string Message)
        {
            if (lstReceive.InvokeRequired)
            {
                ShowMessage message = new ShowMessage(Show);
                lstReceive.Invoke(message, new object[]
{Message});
                return;
            }
            lstReceive.Items.Add(Message);
        }

        public void ShowIP(string Message)
        {
            if (txtIP.InvokeRequired)
            {
                ShowMessage message = new ShowMessage(ShowIP);
                txtIP.Invoke(message, new object[] { Message });
                return;
            }
```

```
            txtIP.Text = Message;
        }

        public string Process(string Message)
        {
            return Message.Trim();
        }
        public void ListenMessage()
        {
            try
            {
                // Khởi tạo UdpClient lắng nghe tại cổng chỉ định
                UdpClient udpClient = new
UdpClient(int.Parse(txtPort.Text));
                while (true)
                {
                    IPEndPoint remote = new
IPEndPoint(IPAddress.Any, 0);
                    ShowIP(remote.Address.ToString());
                    Byte[] receiveBytes = udpClient.Receive(ref
remote);
                    // Chuyển đổi mảng Byte thành chuỗi Unicode
để xử lý
                    string data =
Encoding.Unicode.GetString(receiveBytes);
                    string msg = " Thông điệp nhận từ -> " +
remote.Address.ToString() + " : " + data.ToString();
                    Show(msg);
                    data = Process(data);
                    Byte[] sendBytes =
Encoding.Unicode.GetBytes(data);
                    udpClient.Send(sendBytes, sendBytes.Length,
remote);
                    msg = " Gởi tới -> " +
remote.Address.ToString() + " : " + data.ToString();
                    Show(msg);

                }
            }
            catch (Exception exc)
            {
```

```
                MessageBox.Show(" Không thể khởi động máy chủ !
\n" + exc.ToString());
            }

        }

        private void btnStart_Click(object sender, EventArgs e)
        {
            lstReceive.Items.Clear();
            new Thread(new ThreadStart(ListenMessage)).Start();
            lstReceive.Items.Add("Đang lắng nghe ... ");
        }

        private void lstReceive_SelectedIndexChanged(object
sender, EventArgs e)
        {
            if (lstReceive.SelectedItem != null)
                txtResult.Text =
lstReceive.SelectedItem.ToString();
        }

    }
}
```