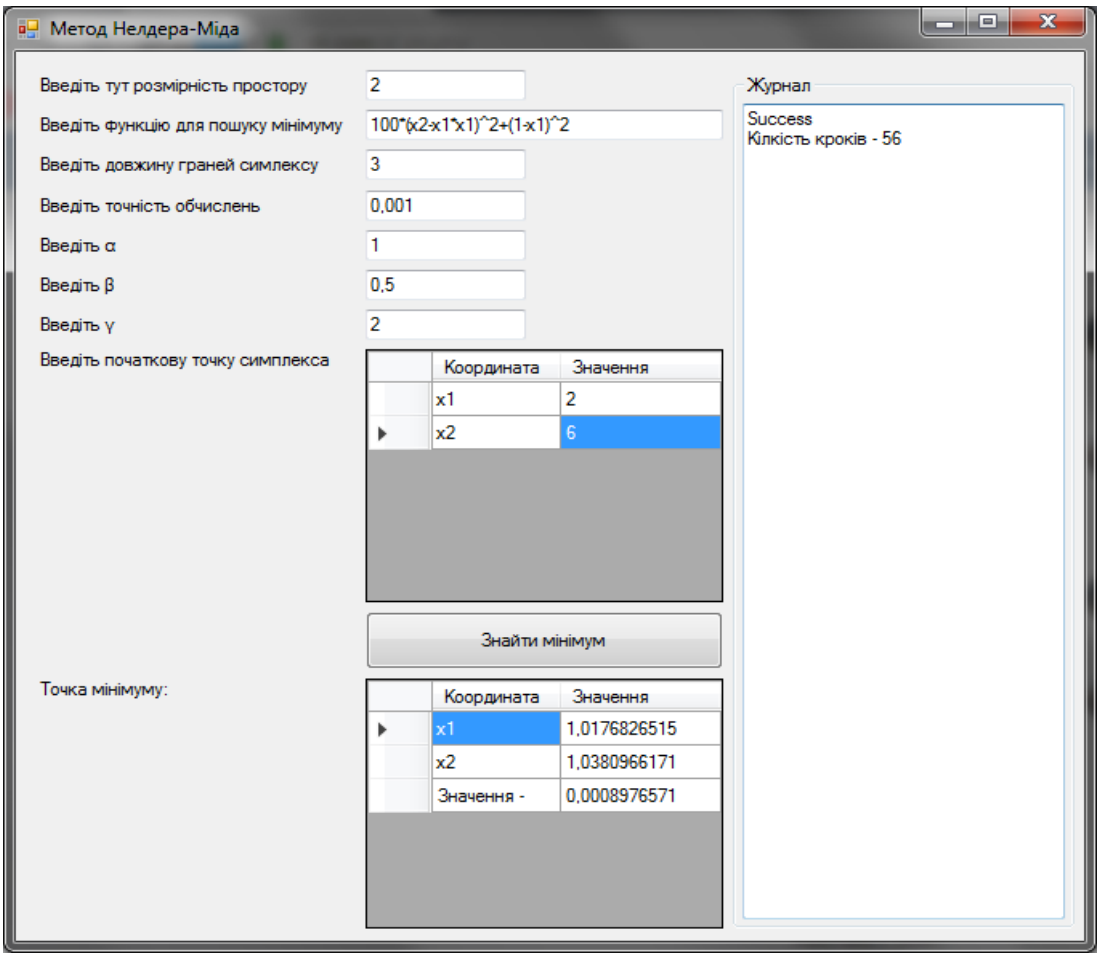
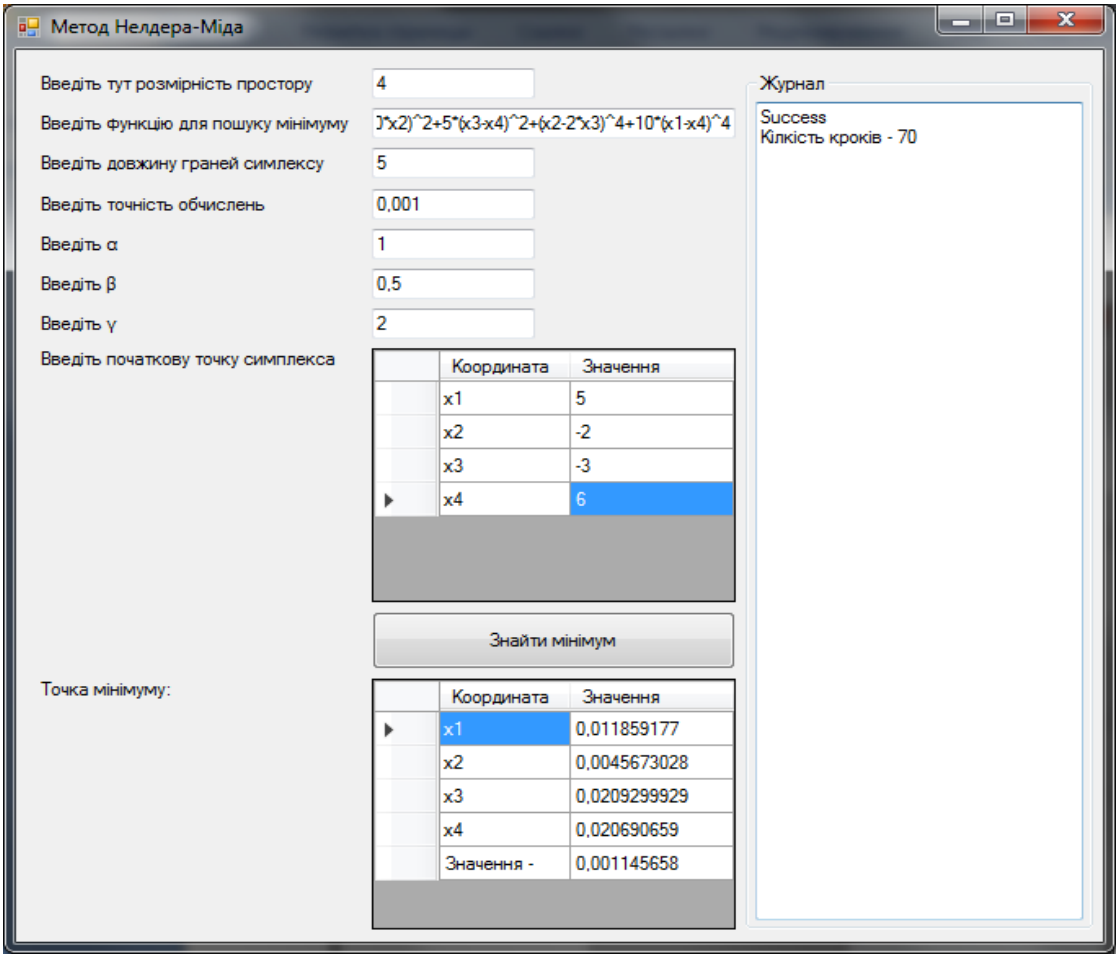

Метод Нелдера-Міда
Введіть тут розмірність простору
2
Введіть функцію для пошуку мінімуму
100*(x2-x1*x1)^2+(1-x1)^2
Введіть довжину граней симлексу
3
Введіть точність обчислень
0,001
Введіть α
1
Введіть β
0,5
Введіть γ
2
Введіть початкову точку симплекса
Координата
Значення
x1
2
x2
6
Знайти мінімум
Точка мінімуму:
Координата
Значення
x1
1,0176826515
x2
1,0380966171
Значення -
0,0008976571
Журнал
Success
Кількість кроків - 56


Метод Нелдера-Міда
Введіть тут розмірність простору
4
Введіть функцію для пошуку мінімуму
J*x2)^2+5*(x3-x4)^2+(x2-2*x3)^4+10*(x1-x4)^4
Введіть довжину граней симлексу
5
Введіть точність обчислень
0,001
Введіть α
1
Введіть β
0,5
Введіть γ
2
Введіть початкову точку симплекса
Координата
Значення
x1
5
x2
-2
x3
-3
x4
6
Знайти мінімум
Точка мінімуму:
Координата
Значення
x1
0,011859177
x2
0,0045673028
x3
0,0209299929
x4
0,020690659
Значення -
0,001145658
Журнал
Success
Кількість кроків - 70

Код програми

```
int i, j, k;
            Parser pars = new Parser();
            if (textBox2.Enabled == false)
            {
                textBox4.Text += "Неможливо почати роботу - для початку введіть усі дані"
+ Environment.NewLine;
                return;
            }
            double step;
            try
            {
                step = Convert.ToDouble(textBox3.Text);
            }
            catch
            {
                textBox4.Text += "Довжина грані введено невірно" + Environment.NewLine;
                return;
            }
            if ((step <= 0) || (step >= 100))
            {
                textBox4.Text += "Довжина граніведено невірно. Памятайте, що вона має
бути додатнім" + Environment.NewLine;
                return;
            }
            string function = textBox2.Text;
            try
            {
                pars.AddVariable("e", Math.E);
                pars.AddVariable("pi", Math.PI);
                for (i = 1; i <= NumberOfDim; i++)
                {
                    pars.AddVariable("x" + i, 0);
                }
                pars.SimplifyDouble(function);
                for (i = 1; i <= NumberOfDim; i++)
                {
                    pars.RemoveVariable("x" + i);
                }
            }
            catch
            {
                textBox4.Text += "Функція введена невірно. Можливо, ви ввели розривну
функцію" + Environment.NewLine;
                return;
            }
            double epsilon;
            try
            {
                epsilon = Convert.ToDouble(textBox5.Text);
            }
            catch
            {
                textBox4.Text += "Точність введена невірно" + Environment.NewLine;
                return;
            }
            double alfa, beta, gamma;
            try
            {
                alfa = Convert.ToDouble(textBox6.Text);
            }
            catch
            {
                textBox4.Text += "Параметр альфа введено невірно" + Environment.NewLine;
```

```
                return;
            }
            try
            {
                beta = Convert.ToDouble(textBox8.Text);
            }
            catch
            {
                textBox4.Text += "параметр бета введено невірно" + Environment.NewLine;
                return;
            }
            try
            {
                gamma = Convert.ToDouble(textBox7.Text);
            }
            catch
            {
                textBox4.Text += "Параметр гамма введено невірно" + Environment.NewLine;
                return;
            }
            double[,] Cors = new double[NumberOfDim + 1, NumberOfDim + 1];
            for (i = 0; i <= NumberOfDim; i++)
            {
                for (j = 1; j <= NumberOfDim; j++)
                {
                    Cors[i,j] = Convert.ToDouble(dataGridView1.Rows[j-1].Cells[1].Value);
                }
            }
            for (j = 1; j <= NumberOfDim; j++)
            {
                Cors[j,j] += step;
            }
            try
            {
                for (i = 0; i <= NumberOfDim; i++)
                {
                    for (j = 1; j <= NumberOfDim; j++)
                    {
                        pars.AddVariable("x" + j, Cors[i,j]);
                    }
                    Cors[i,0] = pars.SimplifyDouble(function);
                    for (j = 1; j <= NumberOfDim; j++)
                    {
                        pars.RemoveVariable("x" + j);
                    }
                }
            }
            catch
            {
                textBox4.Text += "Початкова точка введена невірно" + Environment.NewLine;
                return;
            }
            Cors = MySort(Cors, NumberOfDim+1);
            textBox4.Text += "Success" + Environment.NewLine;
            //////
            int Num = 0;
            double[] SoG, Xr = new double[NumberOfDim + 1], Xe = new double[NumberOfDim +
1], Xc = new double[NumberOfDim + 1];
            while (!CheckAnswer(Cors, NumberOfDim+1, epsilon))
            {
                Num++;
                SoG = FindSog(Cors,NumberOfDim+1);
                for (j = 1; j <= NumberOfDim; j++)
                {
                    pars.AddVariable("x" + j, SoG[j]);
```

```
}
SoG[0] = pars.SimplifyDouble(function);
for (j = 1; j <= NumberOfDim; j++)
{
    pars.RemoveVariable("x" + j);
}
//////
for (i = 1; i < NumberOfDim + 1; i++)
{
    Xr[i] = (1 + alfa) * SoG[i] - alfa * Cors[0, i];
}
for (j = 1; j <= NumberOfDim; j++)
{
    pars.AddVariable("x" + j, Xr[j]);
}
Xr[0] = pars.SimplifyDouble(function);
for (j = 1; j <= NumberOfDim; j++)
{
    pars.RemoveVariable("x" + j);
}
/////
if(Xr[0]<Cors[NumberOfDim,0])
{
    //////
    for (i = 1; i < NumberOfDim + 1; i++)
    {
        Xe[i] = gamma * Xr[i] + (1 - gamma) * SoG[i];
    }
    for (j = 1; j <= NumberOfDim; j++)
    {
        pars.AddVariable("x" + j, Xe[j]);
    }
    Xe[0] = pars.SimplifyDouble(function);
    for (j = 1; j <= NumberOfDim; j++)
    {
        pars.RemoveVariable("x" + j);
    }
    //////
    if (Xe[0] < Cors[NumberOfDim, 0])
    {
        for (i = 0; i < NumberOfDim + 1; i++)
        {
            Cors[0, i] = Xe[i];
        }
        Cors = MySort(Cors, NumberOfDim + 1);
    }
    else
    {
        for (i = 0; i < NumberOfDim + 1; i++)
        {
            Cors[0, i] = Xr[i];
        }
        Cors = MySort(Cors, NumberOfDim + 1);
    }
}
else
{
    if (Xr[0] > Cors[1, 0])
    {
        if (!(Xr[0] > Cors[0, 0]))
        {
            for (i = 0; i < NumberOfDim + 1; i++)
            {
                Cors[0, i] = Xr[i];
            }
```

```
                        Cors = MySort(Cors, NumberOfDim + 1);
                    }
                    //////
                    for (i = 1; i < NumberOfDim + 1; i++)
                    {
                        Xc[i] = beta * Cors[0, i] + (1 - beta) * SoG[i];
                    }
                    for (j = 1; j <= NumberOfDim; j++)
                    {
                        pars.AddVariable("x" + j, Xc[j]);
                    }
                    Xc[0] = pars.SimplifyDouble(function);
                    for (j = 1; j <= NumberOfDim; j++)
                    {
                        pars.RemoveVariable("x" + j);
                    }
                    //////
                    if (Xc[0] > Cors[0, 0])
                    {
                        for (i = 0; i < NumberOfDim; i++)
                        {
                            for (j = 1; j < NumberOfDim + 1; j++)
                            {
                                Cors[i, j] += 0.5 * (Cors[i, j] - Cors[NumberOfDim +
1, j]);

                            }
                        }
                        for (i = 0; i <= NumberOfDim; i++)
                        {
                            for (j = 1; j <= NumberOfDim; j++)
                            {
                                pars.AddVariable("x" + j, Cors[i, j]);
                            }
                            Cors[i, 0] = pars.SimplifyDouble(function);
                            for (j = 1; j <= NumberOfDim; j++)
                            {
                                pars.RemoveVariable("x" + j);
                            }
                        }
                        Cors = MySort(Cors, NumberOfDim+1);
                    }
                    else
                    {
                        for (i = 0; i < NumberOfDim + 1; i++)
                        {
                            Cors[0, i] = Xc[i];
                        }
                        Cors = MySort(Cors, NumberOfDim + 1);
                    }
                }
                else
                {
                    for (i = 0; i < NumberOfDim + 1; i++)
                    {
                        Cors[0, i] = Xr[i];
                    }
                    Cors = MySort(Cors, NumberOfDim + 1);
                }
            }
        }
        for (i = 1; i <= NumberOfDim; i++)
        {
            dataGridView2.Rows[i-1].Cells[1].Value = "" +
Math.Round(Cors[NumberOfDim, i],10);
        }
```

```
            dataGridView2.Rows[NumberOfDim].Cells[1].Value = "" +
Math.Round(Cors[NumberOfDim, 0],10);
           textBox4.Text += "Кількість кроків - " +Num + Environment.NewLine;
```