Autor: Marek Olszewski

Zadanie D3:

Napisać program realizujący metodę różnicową dla równania różniczkowo całkowego

$$\partial_t z(t,x) = x_2 \partial^2_{x_1x_2} z(t,x) + x_1 \partial^2_{x_1x_2} z(t,x) + \frac{1}{4} x_1 x_2 \partial^2_{x_1x_2}(t,x) - z\big(t, 0.5(x_1+x_2), 0.5(x_2-x_1)\big) + \int_{-1}^{1}\int_{-1}^{1} z(t, x_1+s_1, x_2+s_2) ds_1 ds_2 + f(t,x)$$

z warunkiem początkowym

$$z(0,x) = x_1 x_2, \quad x = (x_1, x_2) \in \mathbb{R}^2$$

gdzie

$$f(t,x) = (x_1^2 + x_2^2)\left(1 + \frac{t}{2}\right) - 2t(x_1+x_2) - x_1x_2\left(\frac{t}{4} + 4\right) + \frac{x_2^2 - x_1^2}{4} - \frac{4t}{3}(3x_1^2 + 3x_2^2 + 2).$$

Sporządzić tabelę błędów na zbiorach

$$\{0.5\} \times [100,101] \times [100,101], \ \{0.5\} \times [-101,-100] \times [-101,-100].$$

Rozwiązaniem jest $\bar{z}(t,x) = t(x_1^2 + x_2^2) + x_1x_2$.

Teoria:

Metoda różnicowa (zwana też metodą różnic skończonych) jest metodą numeryczną służącą do aproksymacji rozwiązań równań różniczkowych przy użyciu równań różnic skończonych do przybliżenia pochodnych.

Wyrażona jest ona wzorem:

$$f'(a) \approx \frac{f(a+h) - f(a)}{h}$$

dla pewnego małego h.

Posługując się powyższym wzorem (podstawiając go w miejsce pochodnych) jesteśmy w stanie za pomocą przekształceń obliczyć wartość $f(a+h)$

Opis metody:

Program zgodnie z przedstawioną wcześniej teorią w każdej iteracji wylicza przybliżoną wartość funkcji y. Poszczególne wyniki zostają zapamiętane w globalnej tablicy, tak aby mogły zostać wykorzystane w kolejnym kroku. Dodatkowo (głównie do celów testowych) rysowany jest obraz rozwiązania przybliżonego i dokładnego funkcji y oraz osobno zapisywana do pliku tabela błędów aproksymacji. Program pochłania dużo pamięci ponieważ musi w każdym kroku zapamiętać wartość funkcji dla całej „planszy". Do wyliczenia wartości całki używana jest metoda Simpsona.

Kod programu:

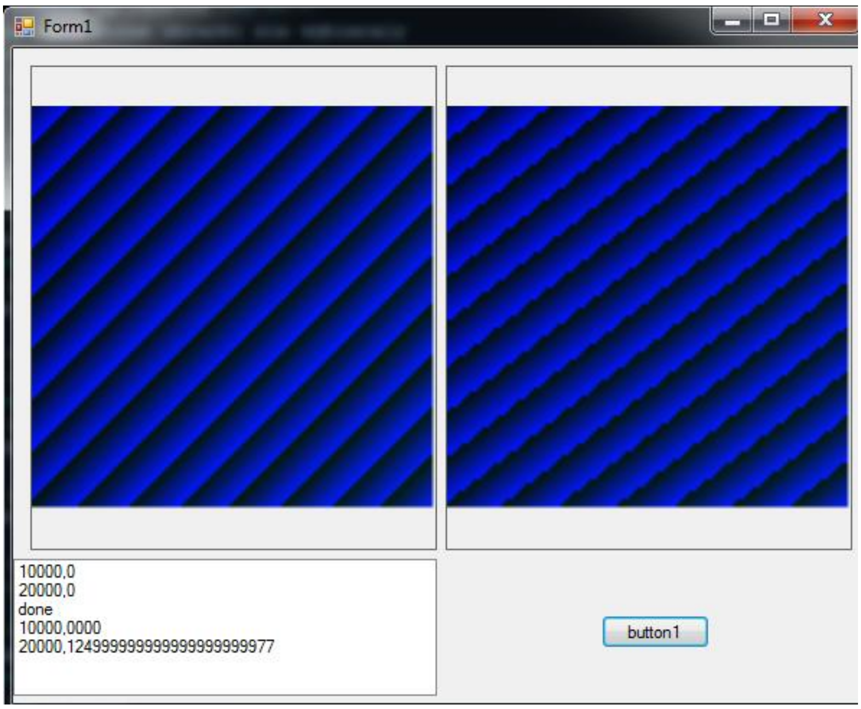
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;

namespace D3
{
    public partial class Form1 : Form
    {
        int x1_start = 100;
        int x1_stop = 101;
        int x2_start = 100;
        int x2_stop = 101;
        decimal t_stop = 0.5m;
        decimal h = 0.5m; //h dla t
        decimal h_ = 0.01m; //h dla x1,x2
        decimal h__ = 1m; //h dla calki
        decimal multi;
        decimal multi_;

        decimal[, ,] Z;

        public Form1()
        {
            InitializeComponent();
            multi_ = 1.0m / h_;
            multi = 1.0m / h;
            Z = new decimal[(int)(multi * t_stop+1), (int)(5 * multi_)+1, (int)(5 * multi_)+1];
        }

        public int mt(decimal t)
        {
            return (int)(t*multi);
        }

        public int mx(decimal x)
        {
            if (x1_start < 0)
            {
                x += 201;
            }
            int n = (int)((x + 1m) * multi_);
            if (n >= (int)(2 * multi_))
                n -= (int)(98 * multi_);
            return n;
        }

        public decimal z_optimal(decimal t, decimal x1, decimal x2)
        {
            return t * (x1 * x1 + x2 * x2) + x1 * x2;
        }

        public decimal f(decimal t, decimal x1, decimal x2)
        {
            return (x1 * x1 + x2 * x2) * (1 + 0.5m * t) - 2 * t * (x1 + x2) - x1 * x2 * (0.25m * t +
4) + 0.25m * (x2 * x2 - x1 * x1) - (4 * t / 3) * (3 * x1 * x1 + 3 * x2 * x2 + 2);
        }

        public decimal całka(decimal t, decimal x1, decimal x2)
        {
```

```
    //całkowanie funkcji 2 zmiennych metodą Simpsona

    decimal a = x1 - 1.0m;
    decimal b = x1 + 1.0m;
    decimal c = x2 - 1.0m;
    decimal d = x2 + 1.0m;

    decimal n = 9;


    decimal hx=(b-a)/n;
    decimal hy=(d-c)/n;

    decimal[] A = {hx/6,4*hx/6,hx/6};
    decimal[] B = {hy/6,4*hy/6,hy/6};

    int i,j;
    decimal x,y=c;

    decimal sum=0;
    decimal px, py;
    int np=0;
    for (j=0; j<n; j++) //100x
    {
        y=c+hy*j;
        for (i=0; i<n; i++) //100x
        {
            x=a+hx*i;
            decimal psum=0;
            py=y;

            decimal[,] ztab = new decimal[3,3];
            for (int k=0; k<3; k++) //3x
            {
                decimal ppsum=0;
                px=x;
                for (int l=0; l<3; l++) //3x
                {
                    decimal fv = z_optimal(t, px, py);
                    ppsum+=A[l]*fv;

                    px+=hx/2;
                }
                psum+=B[k]*ppsum;
                py+=hy/2;
            }

            np++;

            sum+=psum;
        }
    }
    return sum;
}

public decimal pochodna_po_x1(decimal t, decimal x1, decimal x2)
{
    //var ro = (z_optimal(t, x1 + h_, x2) - z_optimal(t, x1, x2)) / h_;
    var rp = (x1 != x1_stop ?
        (Z[mt(t), mx(x1 + h_), mx(x2)] - Z[mt(t), mx(x1), mx(x2)]) / h_ :
        (Z[mt(t), mx(x1), mx(x2)] - Z[mt(t), mx(x1 - h_), mx(x2)]) / h_);
    //if (ro != rp) throw new Exception("Blad pochodnej: " + Math.Abs(ro-rp));
    return rp;
}

public decimal pochodna_po_x1x1(decimal t, decimal x1, decimal x2)
{
    return (pochodna_po_x1(t, x1 + h_, x2) - pochodna_po_x1(t, x1, x2)) / h_;
}
```

```
        public decimal pochodna_po_x2(decimal t, decimal x1, decimal x2)
        {
            //var ro = (z_optimal(t, x1, x2 + h_) - z_optimal(t, x1, x2)) / h_;
            var rp = (x2 != x2_stop ?
                (Z[mt(t), mx(x1), mx(x2 + h_)] - Z[mt(t), mx(x1), mx(x2)]) / h_ :
                (Z[mt(t), mx(x1), mx(x2)] - Z[mt(t), mx(x1), mx(x2 - h_)]) / h_);
            //if (ro != rp) throw new Exception("Blad pochodnej: " + Math.Abs(ro - rp));
            return rp;
        }

        public decimal pochodna_po_x2x2(decimal t, decimal x1, decimal x2)
        {
            return (pochodna_po_x2(t, x1, x2 + h_) - pochodna_po_x2(t, x1, x2)) / h_;
        }

        public decimal pochodna_po_x1x2(decimal t, decimal x1, decimal x2)
        {
            return (pochodna_po_x2(t, x1 + h_, x2) - pochodna_po_x2(t, x1, x2)) / h_;
        }

        public decimal z_start(decimal t, decimal x1, decimal x2)
        {
            return x1 * x2;
        }

        public decimal pochodna_z(decimal t, decimal x1, decimal x2)
        {
            return x2 * pochodna_po_x1x1(t, x1, x2) + x1 * pochodna_po_x2x2(t, x1, x2) + 0.25m *
pochodna_po_x1x2(t, x1, x2) - Z[mt(t), mx(0.5m * (x1 + x2)), mx(0.5m * (x2 - x1))] + całka(t, x1,
x2) + f(t, x1, x2);
        }

        public void z(decimal t, decimal x1, decimal x2)
        {
            Z[mt(t + h), mx(x1), mx(x2)] = h * pochodna_z(t, x1, x2) + Z[mt(t), mx(x1), mx(x2)];
        }

        private void button1_Click(object sender, EventArgs e)
        {
            FileStream fs = new FileStream("tabela.txt", FileMode.Create, FileAccess.Write);
            StreamWriter sw = new StreamWriter(fs);


            var bmp = new Bitmap((int)(Math.Floor((x1_stop - x1_start) / h_)) + 1,
(int)(Math.Floor((x2_stop - x2_start) / h_)) + 1);
            var bmp2 = new Bitmap((int)(Math.Floor((x1_stop - x1_start) / h_)) + 1,
(int)(Math.Floor((x2_stop - x2_start) / h_)) + 1);

            Color c = Color.White;

            for(int i = 0; i < bmp.Width; i++)
                for (int j = 0; j < bmp.Height; j++)
                {
                    var r = (UInt32)(z_optimal(t_stop, x1_start + i * h_, x2_start + j * h_) * 10);
                    c = Color.FromArgb((int)(0xFF000000 | r));
                    bmp.SetPixel(i, j, c);
                }
            pictureBox1.Image = bmp;
            pictureBox2.Image = bmp2;
            listBox1.Items.Add(z_optimal(0.0m, x1_start, x1_start));
            listBox1.Items.Add(z_optimal(t_stop, x1_start, x1_start));
            listBox1.Items.Add("done");
            //krok początkowy, plansza dla t=0.0
            for (decimal x1 = x1_start - 1.0m; x1 <= x1_stop + 1.0m; x1 += h_)
            {
                for (decimal x2 = x2_start - 1.0m; x2 <= x2_stop + 1.0m; x2 += h_)
                {
                    Z[mt(0.0m), mx(x1), mx(x2)] = z_start(0.0m, x1, x2);
```

```
            }
        }

        listBox1.Items.Add(Z[mt(0.0m),mx(x1_start),mx(x1_start)]);

        for (decimal t = 0.0m; t + h <= t_stop; t += h) // raz
        {
            for (decimal x1 = x1_start; x1  <= x1_stop; x1 += h_) //100x
            {
                for (decimal x2 = x2_start; x2  <= x2_stop; x2 += h_) //100x
                {
                    z(t, x1, x2); //[40000x]
                }
            }
        }
        listBox1.Items.Add(Z[mt(t_stop), mx(x1_start), mx(x1_start)]);

        sw.WriteLine("Tabela bledow (t(h) = " + h + ", z(h) = " + h_ + ", I(h) = " + h__ + ":");
        sw.WriteLine(" x1, x2    | rozwiazanie optymalne | rozwiazanie przyblizone | blad");


        for(int i = 0; i < bmp.Width; i++)
            for (int j = 0; j < bmp.Height; j++)
            {
                var ro = z_optimal(t_stop, x1_start + i * h_, x2_start + j * h_);
                var rp = Z[mt(t_stop), mx(x1_start + i * h_), mx(x2_start + j * h_)];
                bmp2.SetPixel(i, j, Color.FromArgb((int)(0xFF000000 | (UInt64)(10 *
Z[mt(t_stop), mx(x1_start + i * h_), mx(x2_start + j * h_)]))));
                sw.WriteLine(" "+(x1_start + i * h_) + "x" + (x2_start + j * h_) + " | " + ro +
" | "+  rp + " | " + Math.Abs(ro-rp));
            }

        sw.Close();
        fs.Close();
    }
  }
}
```

Przykładowe wywołanie programu:



Efekt działania programu dla $h = 0.01$ (widać różnicę między obrazkiem przedstawiającym rozwiązanie optymalne, a obrazkiem przedstawiającym rozwiązanie przybliżone).