Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»_____

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 3

Тема Построение и программная реализация алгоритма сплайн-интерполяции табличных функций.

Студент Малышев И.А.

Группа ИУ7-41Б

Оценка (баллы) _____

Преподаватель Градов В.М.

Москва.
2021 г

Цель работы: Получение навыков владения методами интерполяции таблично заданных функций с помощью кубических сплайнов.

1. Исходные данные

1. Таблица функции с количеством узлов N. Задать с помощью формулы $y = x^2$ в диапазоне [0..10] с шагом 1.

2. Значение аргумента x в первом интервале, например, при x=0.5 и в середине таблицы, например, при x= 5.5. Сравнить с точным значением.

2. Код программы

```
using System;

namespace Lab_03
{
    class Spline
    {
        double[,] table_func;
        double[] c;
        int n;
        int prev, next;

        public double Func(double x) => x * x;

        public Spline(double start, double end, double step = 1.0)
        {
            n = (int)((end - start) / step) + 1;
            table_func = new double[n, 2];

            for (int i = 0; i < n; i++)
            {
                table_func[i, 0] = start + step * i;
                table_func[i, 1] = Func(start + step * i);
            }
        }

        public double GetValue(double x)
        {
            for (int i = 0; i < n - 1; i++)
                if (table_func[i, 0] <= x && x <= table_func[i + 1, 0])
                {
                    prev = i;
                    next = i + 1;
                }

            GetC();
            double a = table_func[prev, 1];
            double b = (table_func[next, 1] - table_func[prev, 1]) / h(next) - h(next) / 3 *
(c[next + 1] + 2 * c[next]);
            double d = (c[next + 1] - c[next]) / (3 * h(next));

            return a + b * (x - table_func[prev, 0]) + c[next] * Math.Pow(x -
table_func[prev, 0], 2) + d * Math.Pow(x - table_func[prev, 0], 3);
```

```
        }

        private double h(int i) => table_func[i, 0] - table_func[i - 1, 0];

        private double f(int i) => 3 * ((table_func[i, 1] - table_func[i - 1, 1]) / h(i) -
(table_func[i - 1, 1] - table_func[i - 2, 1]) / h(i - 1));

        private void GetC()
        {
            c = new double[n + 1];
            double[] xi = new double[n + 1];
            double[] eta = new double[n + 1];

            c[n] = 0;
            xi[2] = 0;
            eta[2] = 0;

            for (int i = 2; i < n; i++)
            {
                xi[i + 1] = -h(i) / (h(i - 1) * xi[i] + 2 * (h(i - 1) + h(i)));
                eta[i + 1] = (f(i) - h(i - 1) * eta[i]) / (h(i - 1) * xi[i] + 2 * (h(i - 1)
+ h(i)));
            }

            for (int i = n; i > next; i--)
                c[i - 1] = xi[i] * c[i] + eta[i];
        }
    }

    static class NewtonePolynome
    {
        static double[,] data;
        static int len;

        static NewtonePolynome()
        {
            double start = 0.0, end = 10.0, step = 1.0;
            len = (int)((end - start) / step) + 1;
            data = new double[len, 2];

            for (int i = 0; i < len; i++)
            {
                data[i, 0] = start + step * i;
                data[i, 1] = (start + step * i) * (start + step * i);
            }
        }

        static double GetDividedDiff(double x0, double xn, double y0, double yn) => (y0 -
yn) / (x0 - xn);

        static (int start, int end) GetDataInterval(double x, int n)
        {
            int index = 0;
            double min_diff = Math.Abs(x - data[0, 0]);

            for (int i = 1; i < len; i++)
                if (min_diff > Math.Abs(x - data[i, 0]))
                {
                    index = i;
                    min_diff = Math.Abs(x - data[i, 0]);
                }

            (int start, int end) interval = (index, index);

            if (n % 2 != 0 && interval.start - 1 >= 0 && interval.end + 1 < len)
```

```
            {
                if (Math.Abs(x - data[interval.start - 1, 0]) < Math.Abs(x -
data[interval.end + 1, 0]))
                    interval.start -= 1;
                else
                    interval.end += 1;
            }

            interval.start = interval.start - n / 2 < 0 ? interval.start : interval.start -
n / 2;
            interval.end = interval.end + n / 2 >= len ? interval.end : interval.end + n /
2;

            return interval;
        }

        public static double GetValue(int n, double x)
        {
            var (start, end) = GetDataInterval(x, n);
            int nodes_count = end - start + 1;

            double[,] div_diff_table = new double[nodes_count, n + 2];

            for (int i = 0; i < nodes_count; i++)
            {
                div_diff_table[i, 0] = data[start + i, 0];
                div_diff_table[i, 1] = data[start + i, 1];
            }

            for (int j = 2, step = 1; j < n + 2; j++, step++)
                for (int i = 0; i < nodes_count + 1 - j; i++)
                    div_diff_table[i, j] = GetDividedDiff(div_diff_table[i, 0],
div_diff_table[i + step, 0],
                                                          div_diff_table[i, j - 1],
div_diff_table[i + 1, j - 1]);

            double res = div_diff_table[0, 1];
            double temp = 1;
            for (int i = 0; i < nodes_count - 1; i++)
            {
                temp *= x - div_diff_table[i, 0];

                res += temp * div_diff_table[0, i + 2];
            }

            return res;
        }
    }


    class Program
    {
        static void Main(string[] args)
        {
            Spline myspline = new Spline(0, 10, 1);
            double x;

            x = 0.5;
            Console.WriteLine("Значение кубического сплайна при x = {0:g2}: {1:g6}", x,
myspline.GetValue(x));
            x = 5.5;
            Console.WriteLine("Значение кубического сплайна при x = {0:g2}: {1:g6}", x,
myspline.GetValue(x));
            Console.WriteLine();
```

```
            x = 0.5;
            Console.WriteLine("Значение Полинома Ньютона 3-ей степени при x = {0:g2}:
{1:g6}", x, NewtonePolynome.GetValue(3, x));
            x = 5.5;
            Console.WriteLine("Значение Полинома Ньютона 3-ей степени при x = {0:g2}:
{1:g6}", x, NewtonePolynome.GetValue(3, x));

            Console.Read();
        }
    }
}
```

3. Результат работы программы

```
Значение кубического сплайна при x = 0,5: 0,341506
Значение кубического сплайна при x = 5,5: 30,2503

Значение Полинома Ньютона 3-ей степени при x = 0,5: 0,5
Значение Полинома Ньютона 3-ей степени при x = 5,5: 30,25
```

4. Ответы на вопросы защиты лабораторной работы

1. Получить выражения для коэффициентов кубического сплайна, построенного на двух точках.

$a = y_0$

$b = \frac{y_1 - y_0}{h_1} + 2\frac{c_1 \cdot h_1}{3}$

$d = \frac{-c_1}{3h_1}$

$c_0 = 0,\ c_1 = \eta_1,\ c_2 = 0$

2. Выписать все условия для определения коэффициентов сплайна, построенного на 3-х точках.

$\Psi(xi\text{-}1) = y_{i-1} = a_i$

$\Psi(xi) = y_i = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3$

$hi = x_i - x_{i-1},\ 1 <= i < 3$

$\Psi'(x) = b_{i+1} = b_i + 2c_i h_i + 3d_i h_i^2$

$\Psi''(x) = 2c_i + 6d_i h_i = \frac{c_{i+1}}{2}$

$\Psi''(x_0) = 0,\ c_1 = 0,\ \Psi''(x_3) = 0,\ c_3 + 3d_3 h_3 = 0$

3. Определить начальные значения прогоночных коэффициентов, если принять, что для коэффициентов сплайна справедливо $C_1=C_2$.

Т. к. $C_1 = 0$, $\xi_2 = 0$ и $\eta_2 = 0$, $C_1 = C_2$, следовательно $\xi_3 = 0$ и $\eta_3 = 0$.

4. Написать формулу для определения последнего коэффициента сплайна C_N, чтобы можно было выполнить обратный ход метода прогонки, если в качестве граничного условия задано $kC_{N-1}+mC_N=p$, где k, m и p – заданные числа.

$C_{N+1} = 0$

$$C_N = \frac{p - mC_{N+1}}{k} = \frac{p}{k}$$