федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №5

Вариант 7

Студент:	Кулагин Вячеслав Дмитриевич
Преподаватель:	Наумова Надежда Александровна
Группа:	P3209

Санкт-Петербург
2025

Содержание

1 Цели работы

Решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

2 Вычислительная реализация задачи

x	y	Вариант	X_1	X_2
0.50	1.5320	**7**	0.751	0.651
0.55	2.5356			
0.60	3.5406			
0.65	4.5462			
0.70	5.5504			
0.75	6.5559			
0.80	7.5594			

2.1 Таблица разностей

x_i	y_i	Δy_i	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$	$\Delta^5 y_i$	$\Delta^6 y_i$
0.50	1.5320	1.0036	0.0014	-0.0008	-0.0012	0.0059	**-0.0166**
0.55	2.5356	1.0050	0.0006	-0.002	0.0047	**-0.0107**	
0.60	3.5406	1.0056	-0.0014	0.0027	**-0.006**		
0.65	4.5462	1.0042	0.0013	**-0.0033**			
0.70	5.5504	1.0055	**-0.0020**				
0.75	6.5559	**1.0035**					
0.80	**7.5594**						

2.2 Для X_1

Воспользуемся формулой Ньютона для интерполирования назад, потому что $X_1 = 0.751$ лежит во второй (правой) части отрезка. Таким образом ($h = 0.05$),

Для $X_1 = 0.751$, $t = \frac{x - x_n}{h} = \frac{0.751 - 0.8}{0.05} = -0.98$

$$N_6(x) = y_6 + t\Delta y_5 + \frac{t(t+1)}{2!}\Delta^2 y_4 + \frac{t(t+1)(t+2)}{3!}\Delta^3 y_3 + \frac{t(t+1)(t+2)(t+3)}{4!}\Delta^4 y_2 +$$

$$+ \frac{t(t+1)(t+2)(t+3)(t+4)}{5!}\Delta^5 y_1 + \frac{t(t+1)(t+2)(t+3)(t+4)(t+5)}{6!}\Delta^6 y_0$$

Получаем,

$$y(0.751) = 6.576$$

2.3 Для X_2

Воспользуемся формулой Гаусса для точки $X_2 = 0.651$. Центральная точка $a = 0.65$, при этом $X_2 > a$, значит используем первую формулу Гаусса. $t = \frac{x - x_0}{h} = 0.02$, $h = 0.05$:

$$P_6(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_{-1} + \frac{t(t+1)(t-1)}{3!}\Delta^3 y_{-1} + \frac{t(t+1)(t-1)(t-2)}{4!}\Delta^4 y_{-2} +$$

$$+\frac{t(t+1)(t+2)(t-1)(t-2)}{5!}\Delta^5 y_{-2}+\frac{t(t+1)(t+2)(t-1)(t-2)(t-3)}{6!}\Delta^6 y_{-3}$$

Получаем,

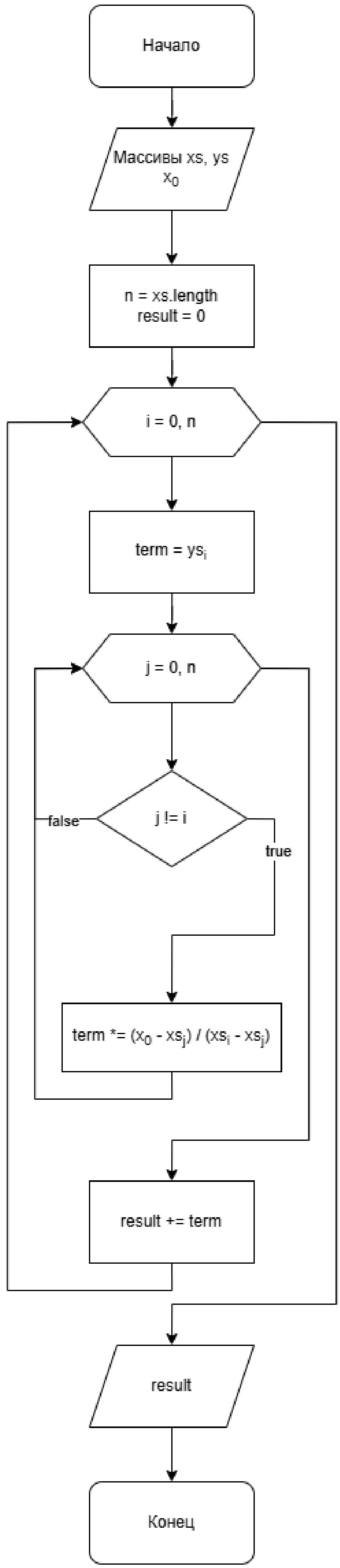
$$y(0.651) = 4.566$$

3 Программная реализация

3.1 Лагранж

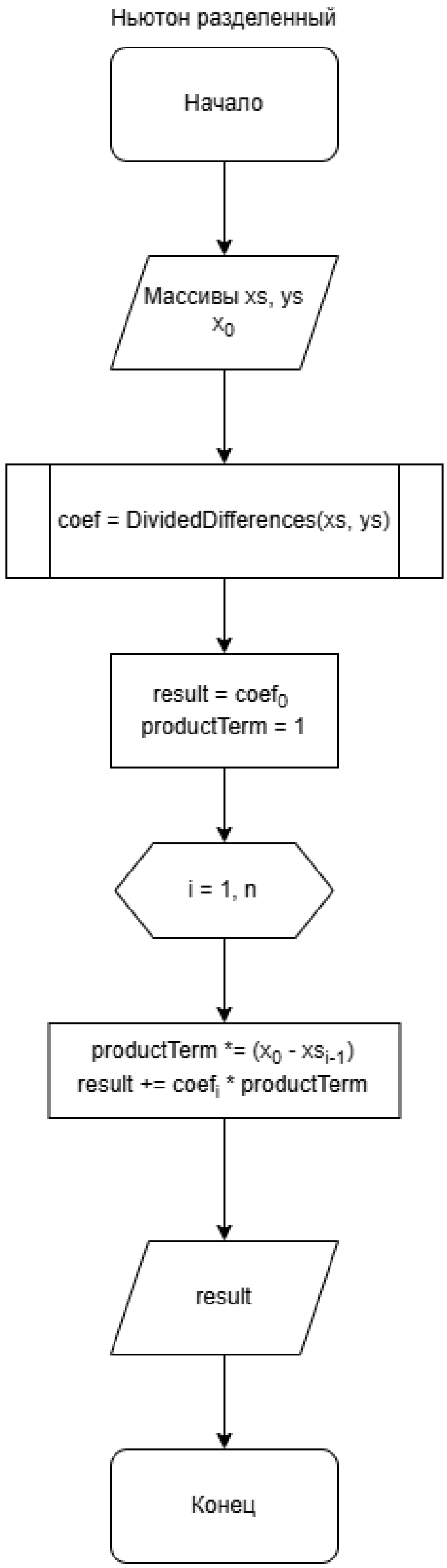
```
public static double interpolate(double[] xs, double[] ys, double x0) {
    int n = xs.length;
    double result = 0.0;
    for (int i = 0; i < n; i++) {
        double term = ys[i];
        for (int j = 0; j < n; j++) {
            if (j != i) {
                term *= (x0 - xs[j]) / (xs[i] - xs[j]);
            }
        }
        result += term;
    }
    return result;
}
```

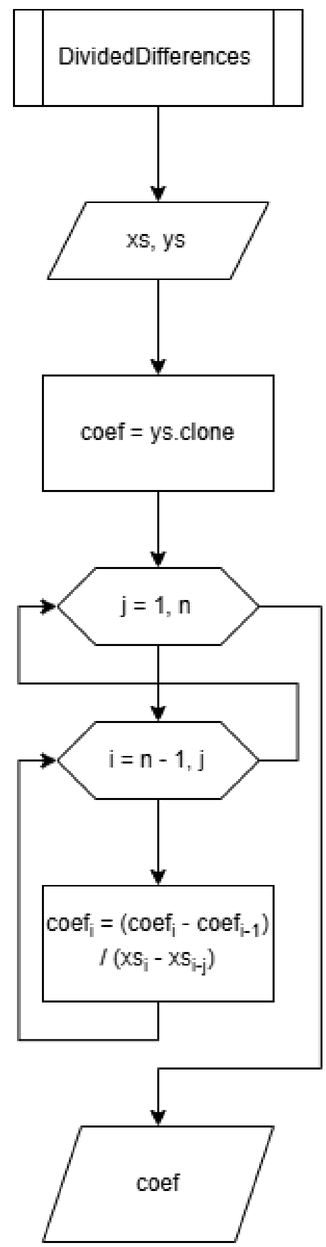
Лагранж


Начало
Массивы xs, ys
x_0
n = xs.length
result = 0
i = 0, n
term = ys_i
j = 0, n
j != i
false
true
term *= $(x_0 - xs_j) / (xs_i - xs_j)$
result += term
result
Конец

3.2 Ньютон с разделенными разностями

```
public static double interpolate(double[] xs, double[] ys, double x0) {
    int n = xs.length;
    double[] coef = computeDividedDifferences(xs, ys);
    double result = coef[0];
    double productTerm = 1.0;

    for (int i = 1; i < n; i++) {
        productTerm *= (x0 - xs[i - 1]);
        result += coef[i] * productTerm;
    }

    return result;
}

private static double[] computeDividedDifferences(double[] xs, double[] ys) {
    int n = ys.length;
    double[] coef = ys.clone();

    for (int j = 1; j < n; j++) {
        for (int i = n - 1; i >= j; i--) {
            coef[i] = (coef[i] - coef[i - 1]) / (xs[i] - xs[i - j]);
        }
    }

    return coef;
}
```



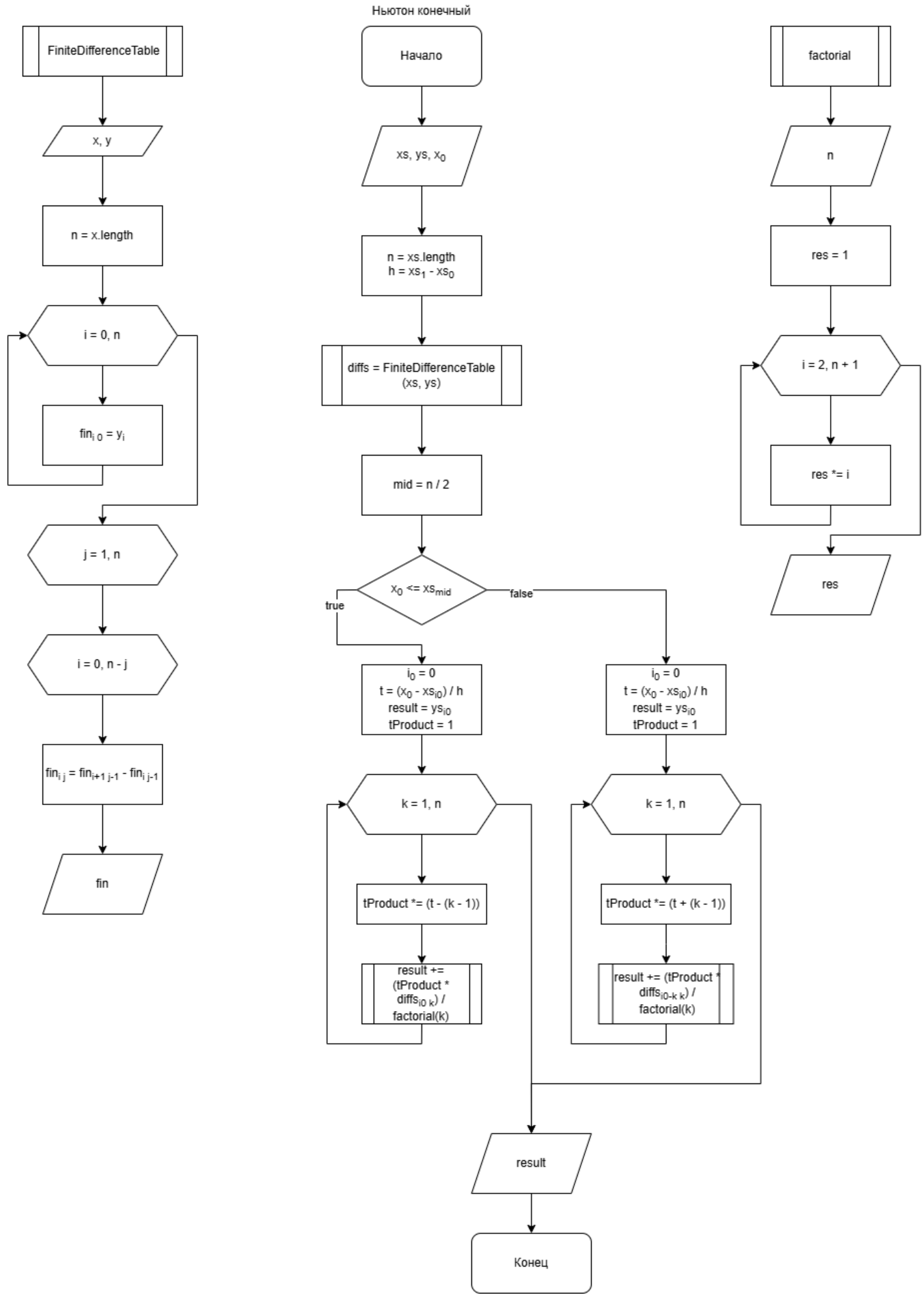


3.3 Ньютон с конечными разностями

```
public static double interpolate(double[] xs, double[] ys, double x0) {
    int n = xs.length;
    double h = xs[1] - xs[0];
```

```java
    FiniteDifferenceTable table = new FiniteDifferenceTable(xs, ys);
    double[][] diffs = table.getFiniteDiff();

    int mid = n / 2;
    if (x0 <= xs[mid]) {
        int i0 = 0;
        double t = (x0 - xs[i0]) / h;
        double result = ys[i0];
        double tProduct = 1.0;

        for (int k = 1; k < n; k++) {
            tProduct *= (t - (k - 1));
            result += (tProduct * diffs[i0][k]) / factorial(k);
        }

        return result;

    } else {
        int i0 = n - 1;
        double t = (x0 - xs[i0]) / h;
        double result = ys[i0];
        double tProduct = 1.0;

        for (int k = 1; k < n; k++) {
            tProduct *= (t + (k - 1));
            result += (tProduct * diffs[i0 - k][k]) / factorial(k);
        }

        return result;
    }
}

private static long factorial(int n) {
    long res = 1;
    for (int i = 2; i <= n; i++) res *= i;
    return res;
}

public FiniteDifferenceTable(double[] x, double[] y) {
    int n = x.length;
    this.x = Arrays.copyOf(x, n);
    this.y = Arrays.copyOf(y, n);
    fin = new double[n][n];

    for (int i = 0; i < n; i++) fin[i][0] = y[i];
    for (int j = 1; j < n; j++) {
        for (int i = 0; i + j < n; i++) {
            fin[i][j] = fin[i + 1][j - 1] - fin[i][j - 1];
        }
```

```
        }
}
```



FiniteDifferenceTable
x, y
n = x.length
i = 0, n
fin_i 0 = y_i
j = 1, n
i = 0, n - j
fin_i j = fin_i+1 j-1 - fin_i j-1
fin
Ньютон конечный
Начало
xs, ys, x0
n = xs.length
h = xs1 - xs0
diffs = FiniteDifferenceTable (xs, ys)
mid = n / 2
x0 <= xs_mid
true
false
i0 = 0
t = (x0 - xs_i0) / h
result = ys_i0
tProduct = 1
k = 1, n
tProduct *= (t - (k - 1))
result += (tProduct * diffs_i0 k) / factorial(k)
tProduct *= (t + (k - 1))
result += (tProduct * diffs_i0-k k) / factorial(k)
result
Конец
factorial
n
res = 1
i = 2, n + 1
res *= i
res

4 Вывод

Проведя эту работу, я научился работать с интерполяцией, была создана программа для расчета значения в случайной точке по имеющимся изначальным точкам. Также были построены графики