

**Федеральное государственное автономное образовательное
учреждение
высшего образования
«Национальный исследовательский университет ИТМО»
Факультет Программной Инженерии и Компьютерной Техники**



**Вариант №9
Лабораторная работа №5
по теме
Интерполяция функции
по дисциплине
Вычислительная математика**

Выполнил Студент группы Р3212
Кобелев Р.П.
к. т. н. Преподаватель:
Наумова Н.А.

г. Санкт-Петербург
2024г.

Содержание

1	Цель работы	2
2	Порядок выполнения работ	2
2.1	Обязательное задание (до 80 баллов)	2
2.1.1	Вычислительная реализация задачи:	2
2.1.2	Программная реализация задачи:	2
2.2	Необязательное задание (до 20 баллов)	2
3	Вычислительная реализация задачи	3
3.1	Основные данные	3
3.2	Вычисление таблицы разностей	3
3.3	Вычисление методом Ньютона для $X_1 = 1.562$	3
3.4	Вычисление методом Гаусса для $X_2 = 1.362$	4
4	Программная реализация задачи	5
4.1	Листинг программы	9
4.2	Пример работы программы	12
5	Github	13
6	Вывод	13

1 Цель работы

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

2 Порядок выполнения работ

2.1 Обязательное задание (до 80 баллов)

2.1.1 Вычислительная реализация задачи:

1. Выбрать из табл. 1 заданную по варианту таблицу $y = f(x)$ (таблица 1.1);
2. Построить таблицу конечных разностей для заданной таблицы. Таблицу отразить в отчете;
3. Вычислить значения функции для аргумента X_1 (см. табл. 1.1), используя первую или вторую интерполяционную формулу Ньютона. Обратит внимание какой конкретно формулой необходимо воспользоваться;
4. Вычислить значения функции для аргумента X_2 (см. табл. 1.1), используя первую или вторую интерполяционную формулу Гаусса. Обратит внимание какой конкретно формулой необходимо воспользоваться;
5. Подробные вычисления привести в отчете.

2.1.2 Программная реализация задачи:

1. Исходные данные задаются тремя способами:
2. (а) в виде набора данных (таблицы x, y), пользователь вводит значения с клавиатуры;
(б) в виде сформированных в файле данных (подготовить не менее трех тестовых вариантов);
(с) на основе выбранной функции, из тех, которые предлагает программа, например, $\sin x$. Пользователь выбирает уравнение, исследуемый интервал и количество точек на интервале (не менее двух функций).
3. Сформировать и вывести таблицу конечных разностей;
4. Вычислить приближенное значение функции для заданного значения аргумента, введенного с клавиатуры, указанными методами (см. табл. 2). Сравнить полученные значения;
5. Построить графики заданной функции с отмеченными узлами интерполяции и интерполяционного многочлена Ньютона/Гаусса (разными цветами);
6. Программа должна быть протестирована на различных наборах данных, в том числе и некорректных.
7. Проанализировать результаты работы программы.

2.2 Необязательное задание (до 20 баллов)

1. Реализовать в программе вычисление значения функции для заданного значения аргумента, введенного с клавиатуры, используя схемы Стирлинга;
2. Реализовать в программе вычисление значения функции для заданного значения аргумента, введенного с клавиатуры, используя схемы Бесселя.

3 Вычислительная реализация задачи

3.1 Основные данные

x	y
1.05	0.1213
1.15	1.1316
1.25	2.1459
1.35	3.1565
1.45	4.1571
1.55	5.1819
1.65	6.1969

X_1	X_2
1.562	1.362

3.2 Вычисление таблицы разностей

Для первого ряда мы имеем формулу:

$$\Delta y_i = y_{i+1} - y_i$$

Вычислим значения для 1 ряда:

$$\Delta y_1 = 1.1316 - 0.1213 = 1.0103$$

$$\Delta y_2 = 2.1459 - 1.1316 = 1.0143$$

$$\Delta y_3 = 3.1565 - 2.1459 = 1.0106$$

$$\Delta y_4 = 4.1571 - 3.1565 = 1.0006$$

$$\Delta y_5 = 5.1819 - 4.1571 = 1.0248$$

$$\Delta y_6 = 6.1969 - 5.1819 = 1.015$$

В следующем у нас зануляется нижний элемент и дальше используем формулу:

$$\Delta^2 y_i = \Delta y_{i+1} - \Delta y_i$$

Вычислим значения для 2 ряда:

$$\Delta^2 y_1 = 1.0143 - 1.0103 = 0.004$$

$$\Delta^2 y_2 = 1.0106 - 1.0143 = -0.0037$$

$$\Delta^2 y_3 = 1.0006 - 1.0106 = -0.01$$

$$\Delta^2 y_4 = 1.0248 - 1.0006 = 0.0242$$

$$\Delta^2 y_5 = 1.015 - 1.0248 = -0.0098$$

И так далее для следующих рядов - мы получили таблицу разностей:

y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	$\Delta^5 y$	$\Delta^6 y$
0.1213	1.01	0.004	-0.0077	0.0014	0.0391	-0.1478
1.132	1.014	-0.0037	-0.0063	0.0405	-0.1087	0.0
2.146	1.011	-0.01	0.0342	-0.0682	0.0	0.0
3.156	1.001	0.0242	-0.034	0.0	0.0	0.0
4.157	1.025	-0.0098	0.0	0.0	0.0	0.0
5.182	1.015	0.0	0.0	0.0	0.0	0.0
6.197	0.0	0.0	0.0	0.0	0.0	0.0

3.3 Вычисление методом Ньютона для $X_1 = 1.562$

Смотря на значение x можно прийти к выводу, что для вычисления мы используем формулу Ньютона для интерполирования назад, так как X_1 лежит во второй половине.

$$t = \frac{x - x_n}{h} = \frac{1.562 - 1.65}{0.1} = -0.88$$

$$N_5(x) = y_5 + t\Delta y_4 + \frac{t(t+1)}{2!}\Delta^2 y_3 + \frac{t(t+1)(t+2)}{3!}\Delta^3 y_2 +$$

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

$$N_5(1.562) = 5.182 + -0.88 \cdot 1.025 + \frac{-0.88(-0.88+1)}{2!}0.0242 + \frac{-0.88(-0.88+1)(-0.88+2)}{3!}0.0342 + \\ \frac{-0.88(-0.88+1)(-0.88+2)(-0.88+3)}{4!}0.0405 + \frac{-0.88(-0.88+1)(-0.88+2)(-0.88+3)(-0.88+4)}{5!}0.0391 = 4.2773698$$

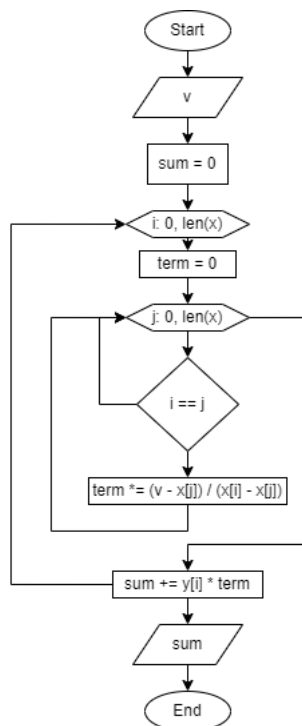
3.4 Вычисление методом Гаусса для $X_2 = 1.362$

$$t = \frac{x - x_0}{h} = \frac{1.362 - 1.35}{0.1} = 0.12$$

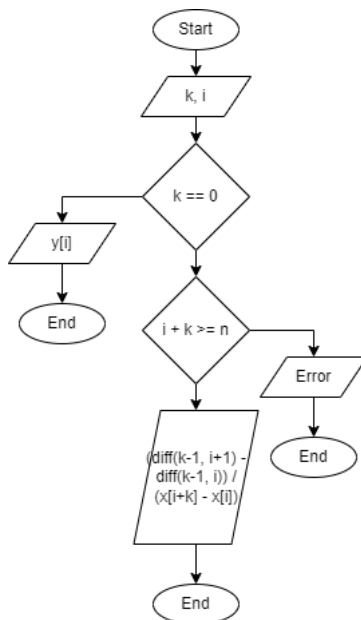
$$P_3(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-1} + \frac{(t+1)t(t-1)(t-2)}{4!}\Delta^4 y_{-2} + \\ \frac{(t+2)(t+1)t(t-1)(t-2)}{5!}\Delta^5 y_{-2} + \frac{(t+2)(t+1)t(t-1)(t-2)(t-3)}{6!}\Delta^6 y_{-3}$$

$$P_3(1.362) = 3.156 + (0.12) \cdot 1.001 + \frac{(0.12)(0.12-1)}{2} \cdot -0.01 + \frac{(0.12+1)(0.12)(0.12-1)}{6} \cdot 0.0342 + \\ \frac{(0.12+1)0.12(0.12-1)(0.12-2)}{24} \cdot 0.0405 + \frac{(0.12+2)(0.12+1)0.12(0.12-1)(0.12-2)}{120} \cdot -0.1087 + \\ \frac{(0.12+2)(0.12+1)0.12(0.12-1)(0.12-2)(0.12-3)}{720} \cdot -0.1478 = 3.2762$$

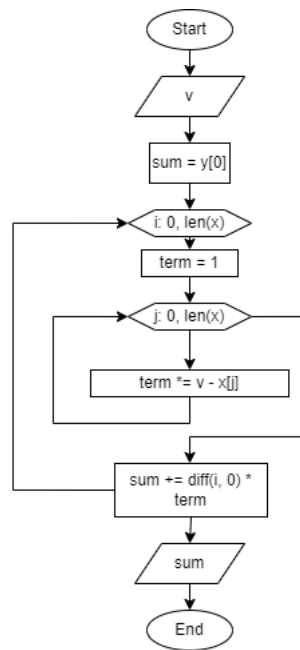
4 Программная реализация задачи



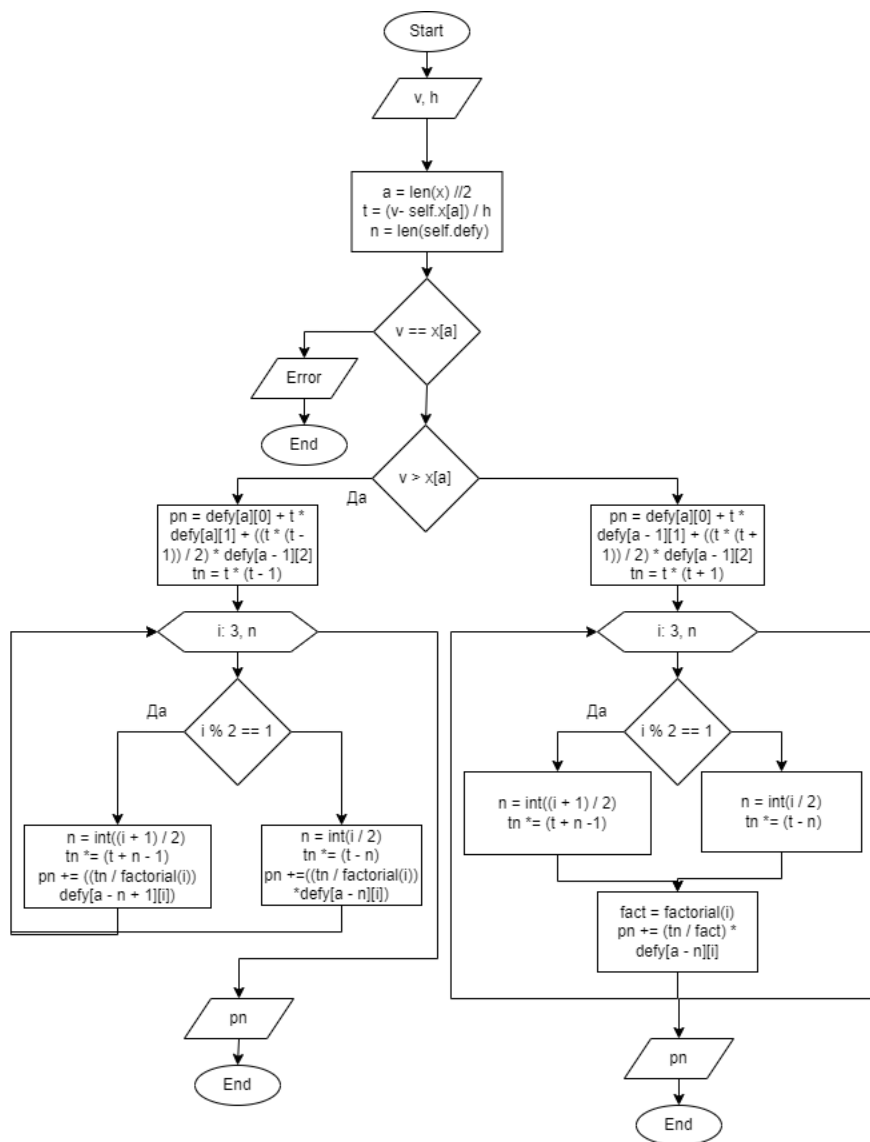
Лагранж



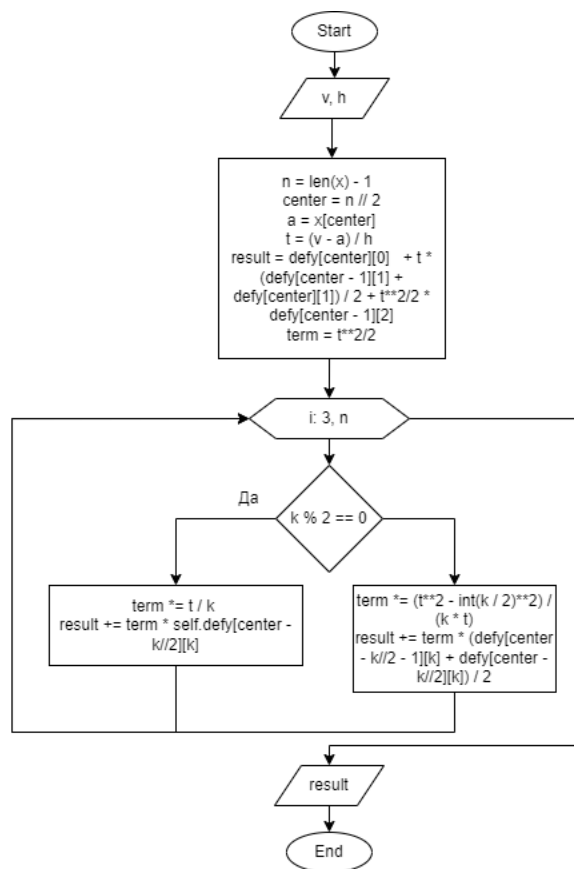
Вспомогательная функция для Ньютона



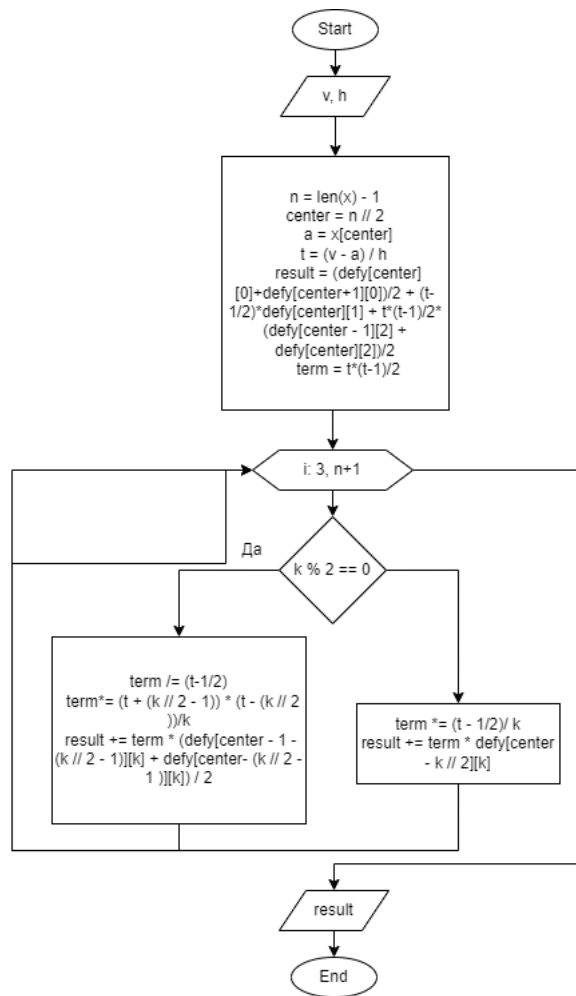
НЬЮТОН



Гайс



Стирлинг



Бессель

4.1 Листинг программы

Interpolation.py

```

1 from dataclasses import dataclass
2 from math import factorial
3
4 @dataclass
5 class Interpolation:
6     n: int
7     x: []
8     y: []
9
10
11 def difference_table(self):
12     n = len(self.y)
13     defy = [[0] * n for _ in range(n)]
14
15     for i in range(n):
16         defy[i][0] = self.y[i]
17
18     for i in range(1, n):
19         for j in range(n - i):
20             defy[j][i] = defy[j + 1][i - 1] - defy[j][i - 1]
  
```

```

21     self.defy = defy
22     return
23
24 def lagrange(self, v):
25     sum = 0.0
26     for i in range(self.n):
27         term = 1.0
28         for j in range(self.n):
29             if j == i:
30                 continue
31             else:
32                 term *= (v - self.x[j]) / (self.x[i] - self.x[j])
33         sum += self.y[i] * term
34     return sum
35
36
37 def diff(self, k, i):
38     if k == 0:
39         return self.y[i]
40     elif i + k ≥ self.n:
41         raise ValueError("Index out of bounds")
42     else:
43         return (self.diff(k - 1, i + 1) - self.diff(k - 1, i)) / (self.x[i + k] -
44             ↪ self.x[i])
45
46
47 def newton(self, v):
48     sum = self.y[0]
49     for i in range(1, self.n):
50         term = 1.0
51         for j in range(i):
52             term *= v - self.x[j]
53         sum += self.diff(i, 0) * term
54     return sum
55
56
57 def gauss(self, v, h):
58     a = len(self.y) // 2
59     if v > self.x[a]:
60         t = (v - self.x[a]) / h
61         n = len(self.defy)
62         pn = self.defy[a][0] + t * self.defy[a][1] + ((t * (t - 1)) / 2) * self.defy
63             ↪ [a - 1][2]
64         tn = t * (t - 1)
65         for i in range(3, n):
66             if i % 2 == 1:
67                 n = int((i + 1) / 2)
68                 tn *= (t + n - 1)
69                 pn += ((tn / factorial(i)) * self.defy[a - n + 1][i])
70             else:
71                 n = int(i / 2)
72                 tn *= (t - n)
73                 pn += ((tn / factorial(i)) * self.defy[a - n][i])
74
75     elif v < self.x[a]:
76         t = (v - self.x[a]) / h
77         n = len(self.defy)

```

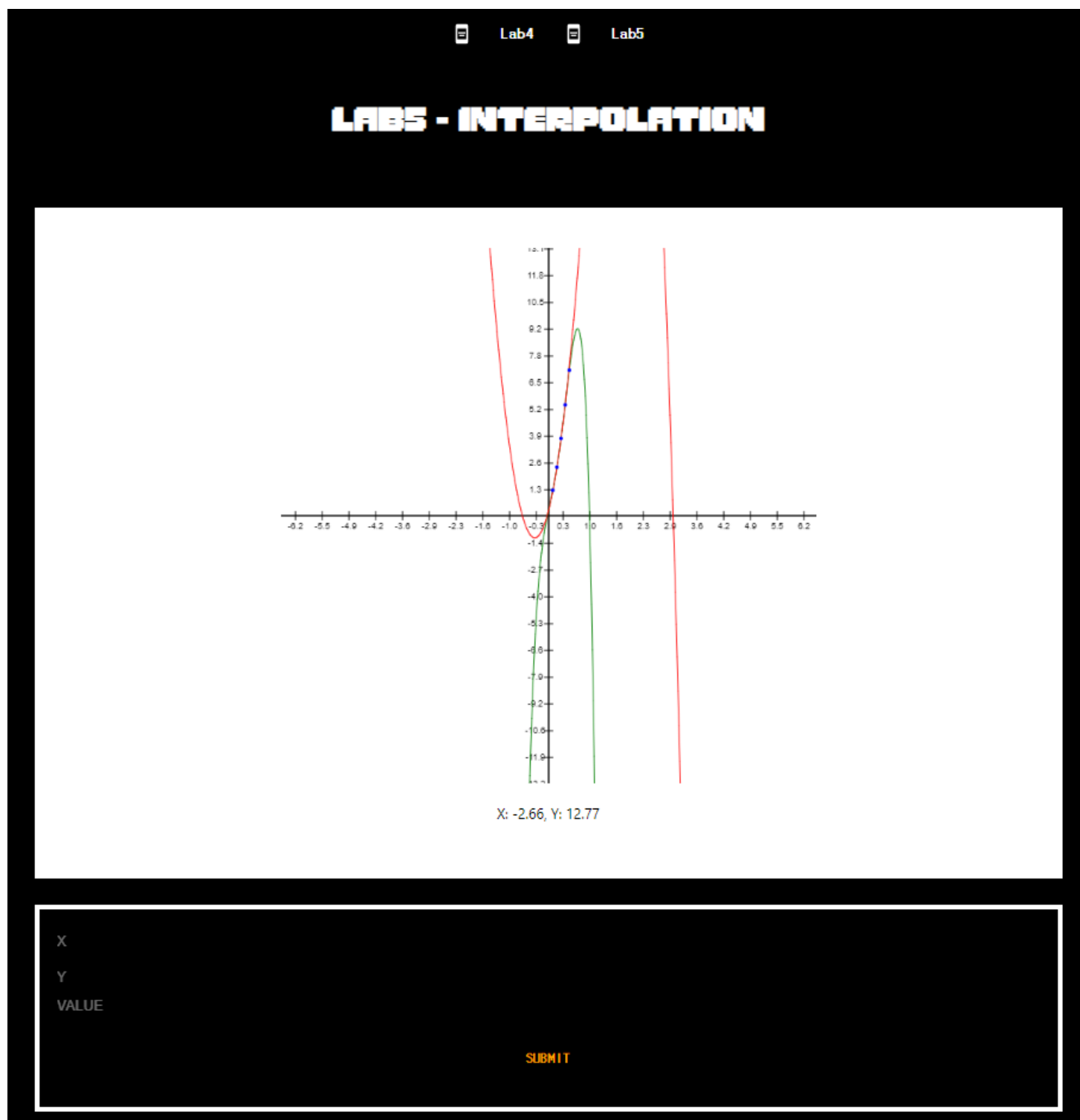
```

75     pn = self.defy[a][0] + t * self.defy[a - 1][1] + ((t * (t + 1)) / 2) * self.
    ↪ defy[a - 1][2]
76     tn = t * (t + 1)
77     for i in range(3, n):
78         if i % 2 == 1:
79             n = int((i + 1) / 2)
80             tn *= (t + n - 1)
81         else:
82             n = int(i / 2)
83             tn *= (t - n)
84
85         fact = factorial(i)
86         pn += (tn / fact) * self.defy[a - n][i]
87     else:
88         raise ValueError("Error in Gauss")
89
90     return pn
91
92 def bessel(self, v, h):
93     n = len(self.x) - 1
94     center = n // 2
95     a = self.x[center]
96     t = (v - a) / h
97     result = (self.defy[center][0] + self.defy[center+1][0]) / 2 + (t - 1/2) * self.defy[
    ↪ center][1] + t * (t - 1) / 2 * (self.defy[center - 1][2] + self.defy[center][2])
    ↪ / 2
98     term = t * (t - 1) / 2
99     for k in range(3, n + 1):
100         if k % 2 == 0:
101             term /= (t - 1/2)
102             term *= (t + (k // 2 - 1)) * (t - (k // 2)) / k
103             result += term * (self.defy[center - 1 - (k // 2 - 1)][k] + self.defy[
    ↪ center - (k // 2 - 1)][k]) / 2
104         else:
105             term *= (t - 1/2) / k
106             result += term * self.defy[center - k // 2][k]
107
108     return result
109
110 def sterling(self, v, h):
111     n = len(self.x) - 1
112     center = n // 2
113     a = self.x[center]
114     t = (v - a) / h
115
116     result = self.defy[center][0] + t * (self.defy[center - 1][1] + self.defy[
    ↪ center][1]) / 2 + t**2 / 2 * self.defy[center - 1][2]
117     term = t**2 / 2
118     for k in range(3, n):
119         if k % 2 == 0:
120             term *= t / k
121             result += term * self.defy[center - k // 2][k]
122         else:
123             term *= (t**2 - int(k / 2)**2) / (k * t)
124             result += term * (self.defy[center - k // 2 - 1][k] + self.defy[center - k
    ↪ // 2][k]) / 2

```

125
126 `return result`

4.2 Пример работы программы



2

0.32

SUBMIT FROM FILE

12

1



7

3.4

1

SUBMIT FUNCTION

y_i	$\Delta^1 y_i$	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$
1.25	1.13	0.28000000000000025	-0.04000000000000036	-0.15000000000000124
2.38	1.4100000000000001	0.2400000000000002	-0.19000000000000128	0
3.79	1.6500000000000004	0.04999999999998934	0	0
5.44	1.6999999999999993	0	0	0
7.14	0	0	0	0

Lagrange	Newton 	Gauss 	Sterling	Bessel
4.1047199999999995	4.10472	4.10472	4.104480000000001	4.1058

5 Github

[Ссылка на GitHub](#)

6 Вывод

В этой работе я ознакомился с различными методами интерполяции по точкам