Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы численного анализа

**ОТЧЁТ**

по лабораторной работе №7

Интерполяция сплайнами

Выполнил:

студент группы 253502

Шишко В. В.

Руководитель:

доцент

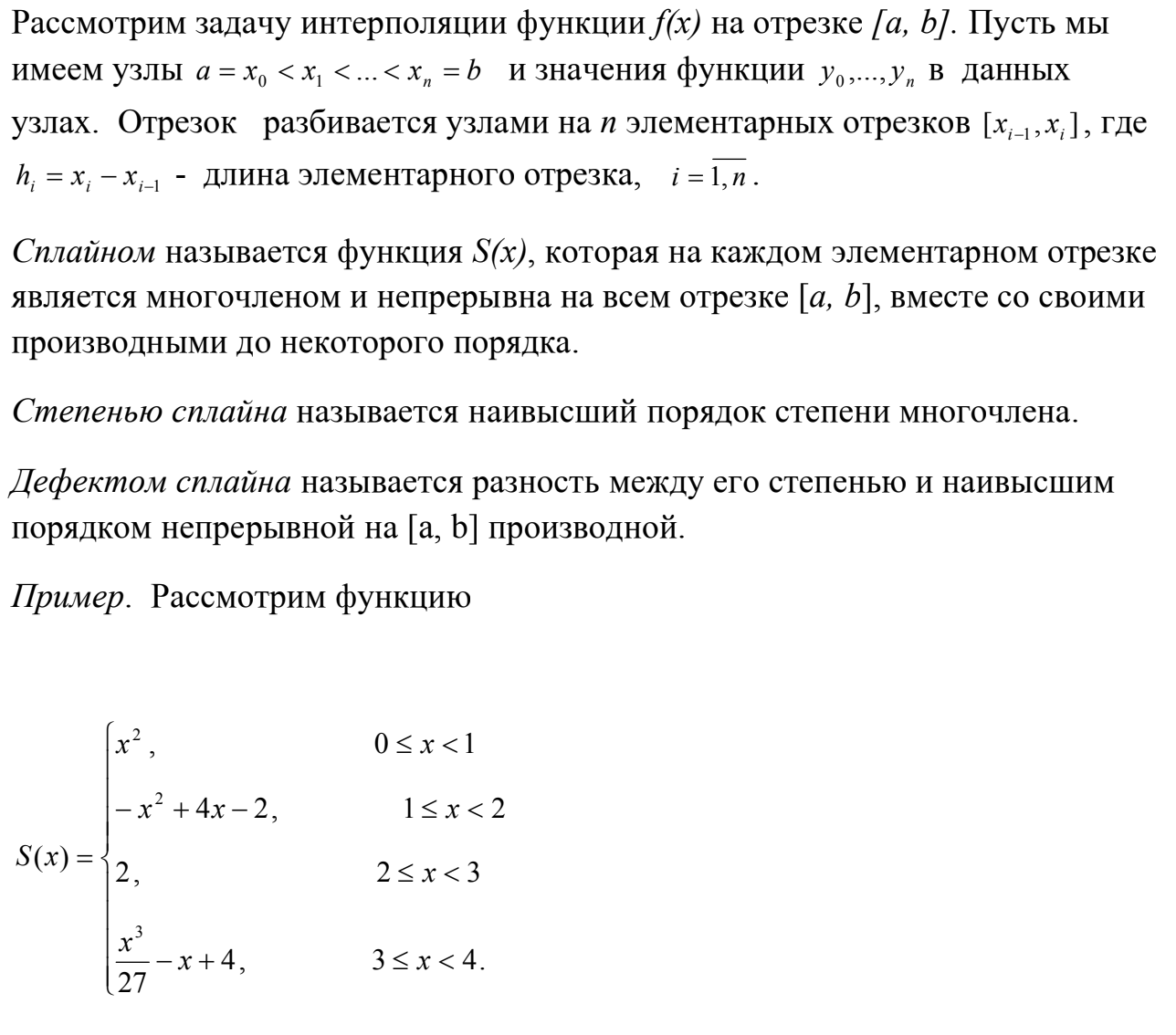
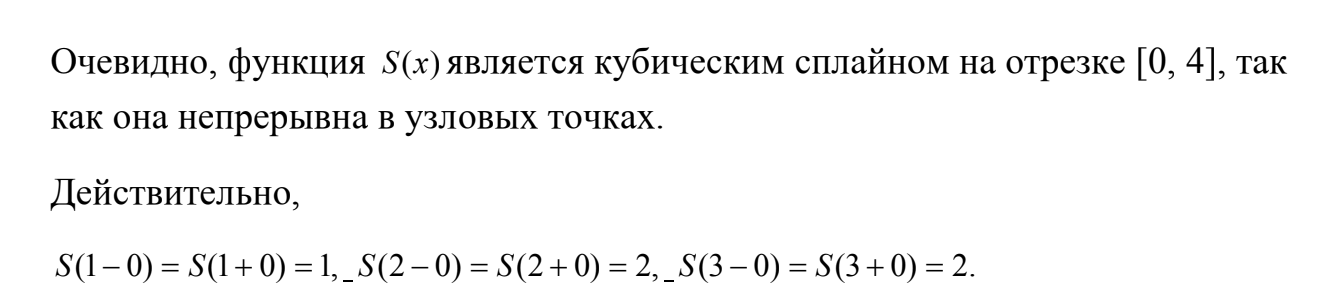
Анисимов В.Я.

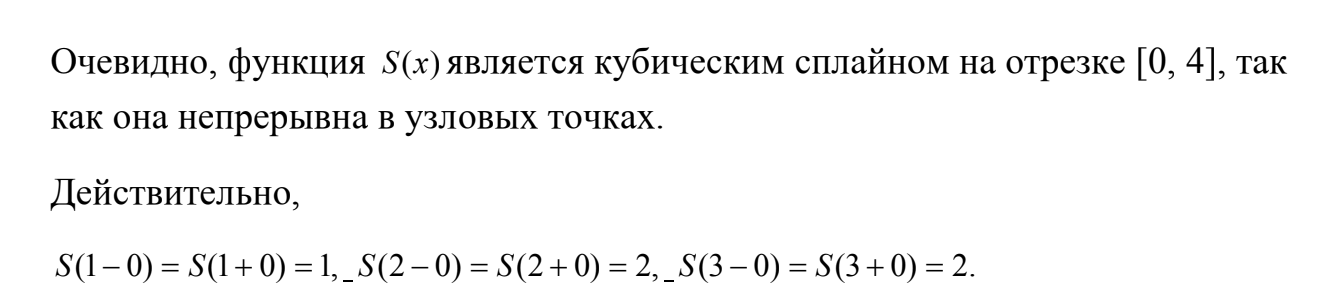
Минск 2023

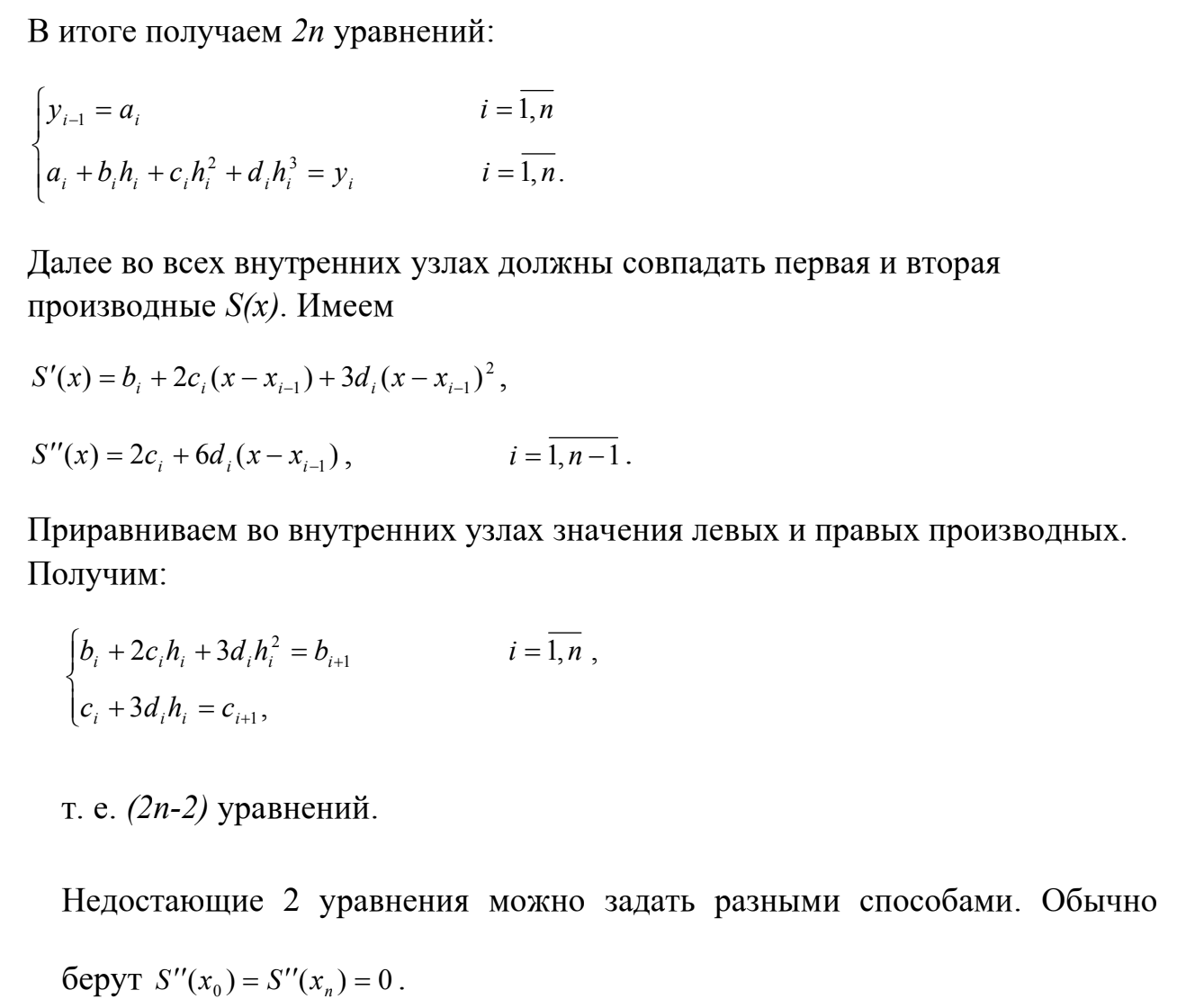
1. **Цель работы**

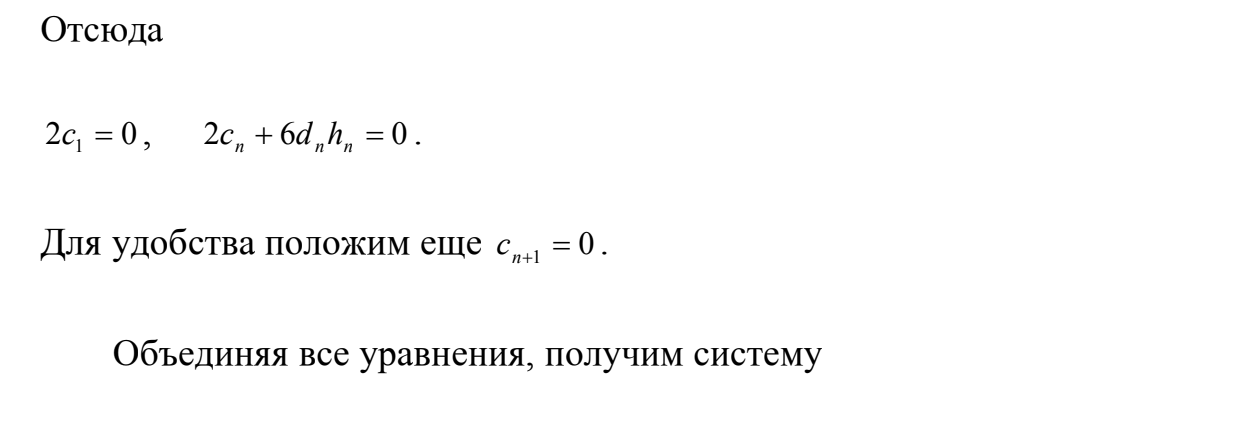
Изучить построение кубических интерполяционных сплайнов.

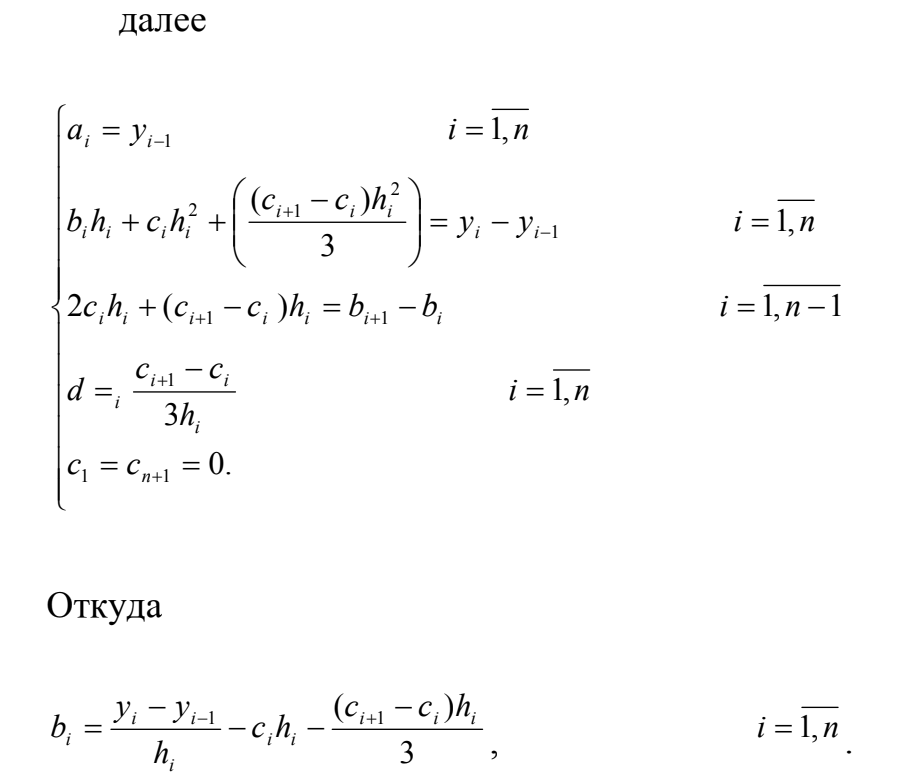
Реализовать метод построение Кубического сплайна.

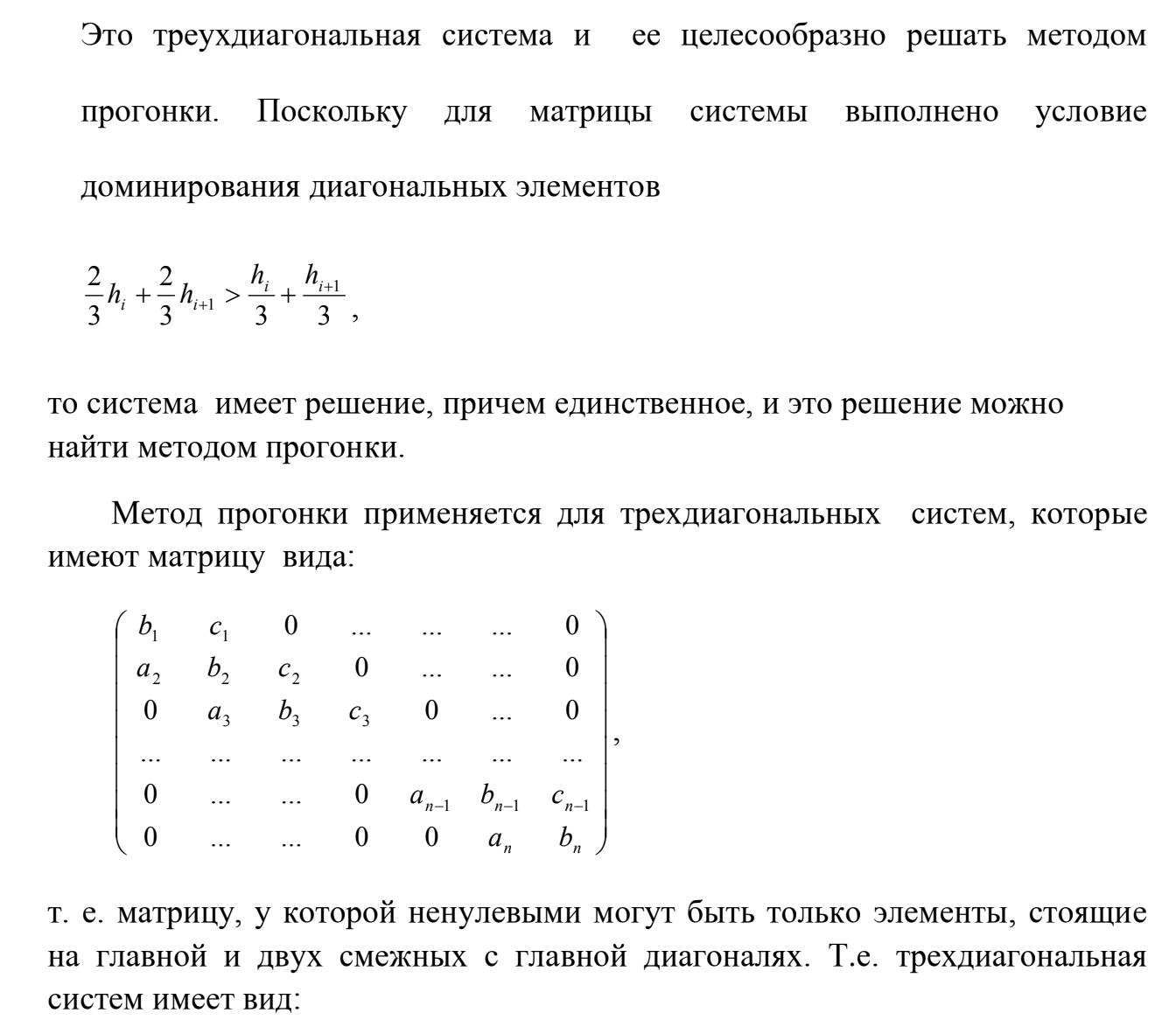
1. **Краткие теоретические сведения**

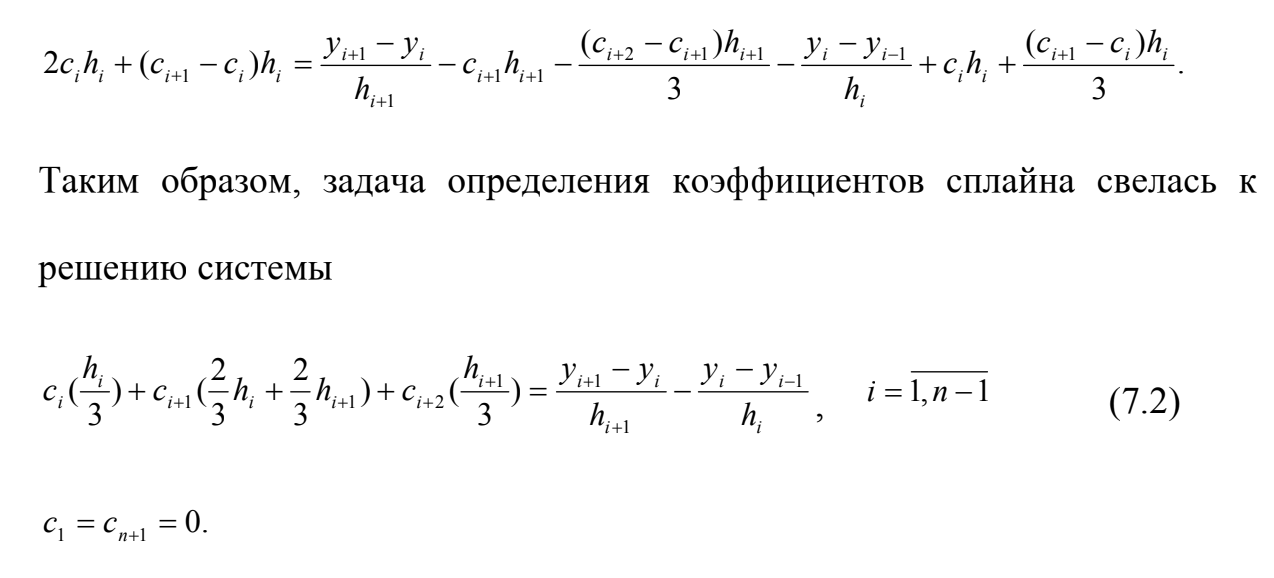


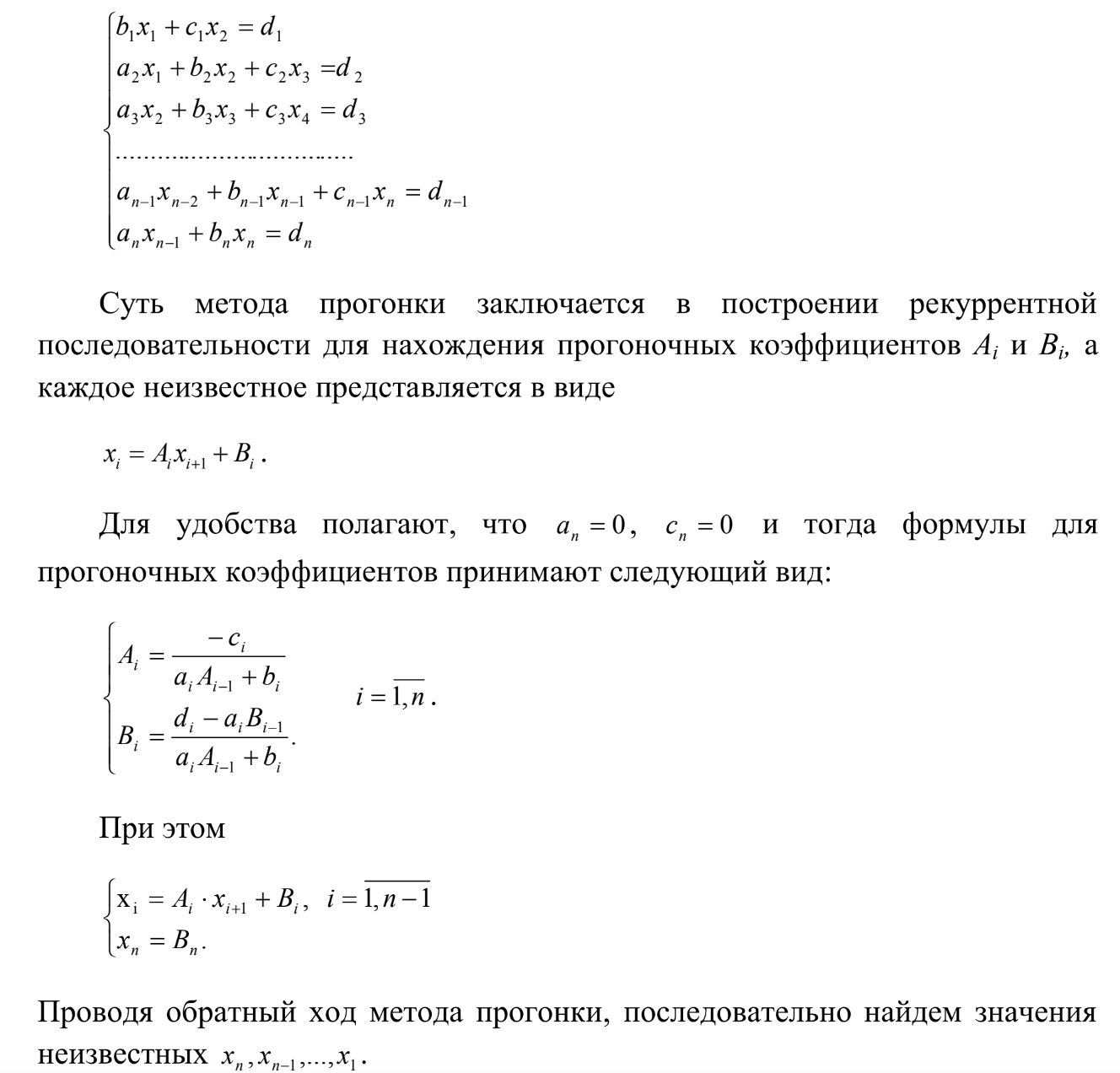




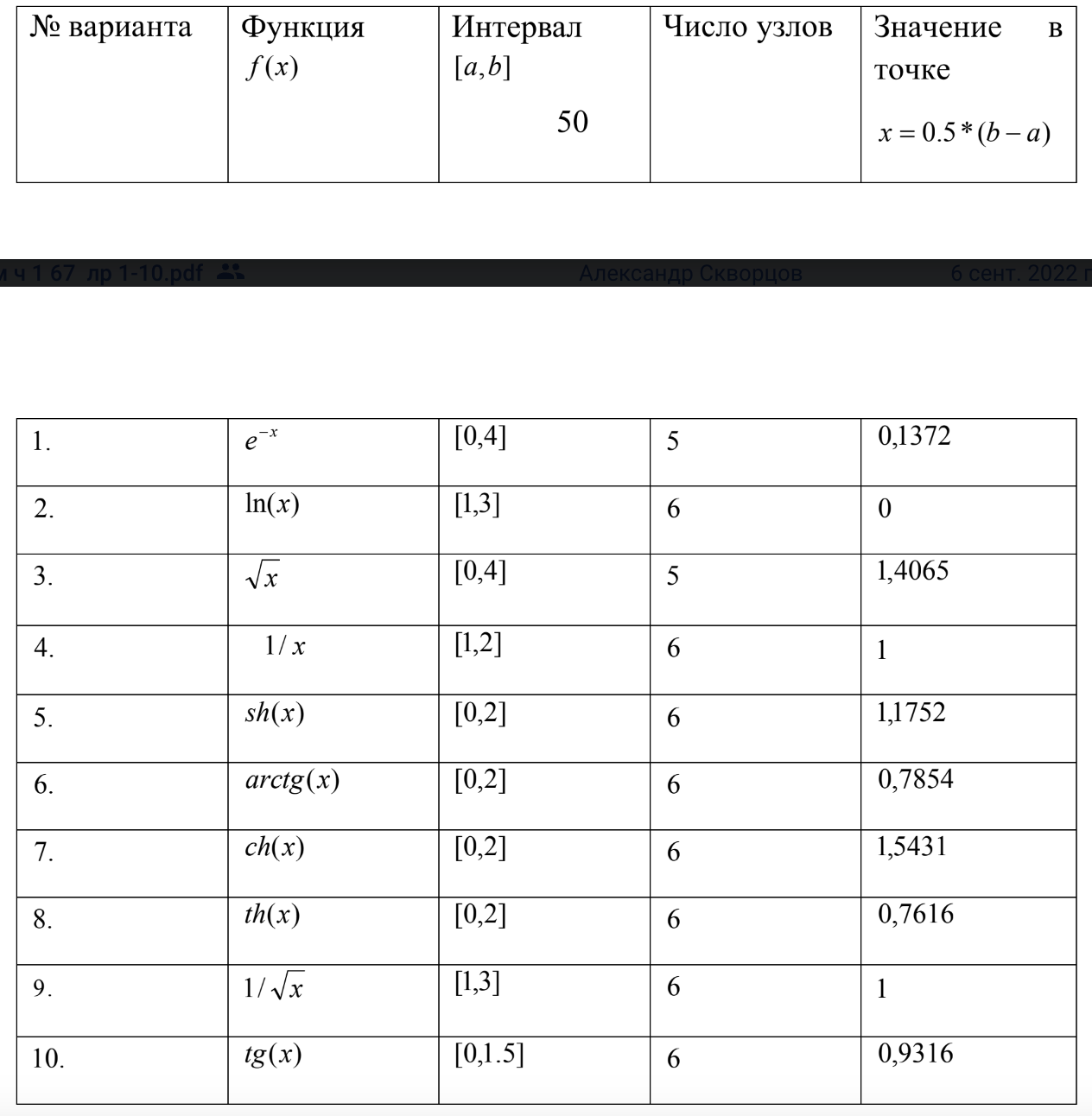






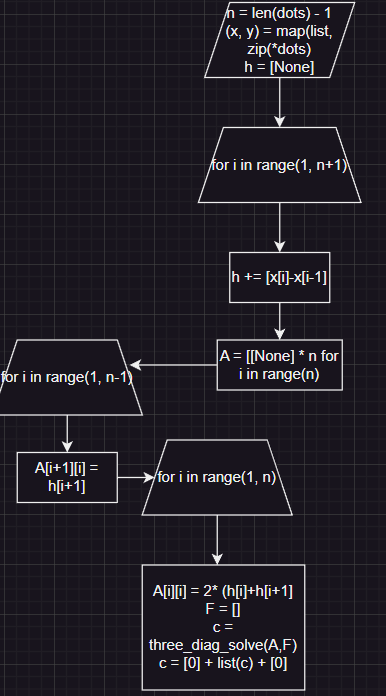


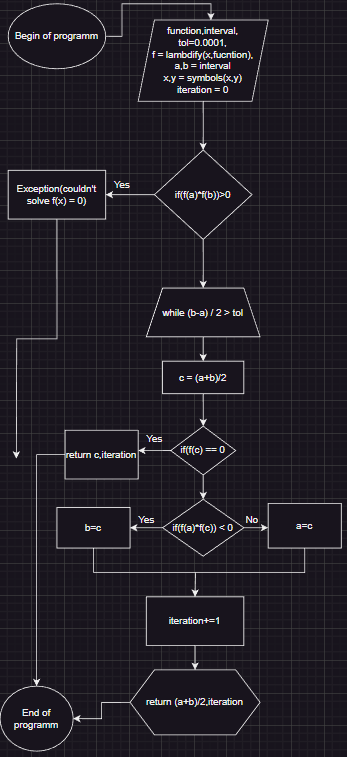
**Задача**

Произвести интерполирование кубическими сплайнами приведенных в таблице функций. Вычислить значение сплайна в точке . Значение сплайна в точке записать в качестве ответа. Сравнить его со значением функции в соответствующей точке.

1. **Программная реализация**

*cubic\_spline*





def three\_diag\_solve(A, b):

    A = A.copy()

    b = b.copy()

    n = len(A)

    A[0][1] /= A[0][0]

    for i in range(1, n - 1):

        A[i][i + 1] /= (A[i][i] - A[i][i - 1] \* A[i - 1][i])

    b[0] /= A[0][0]

    for i in range(1, n):

        b[i] = (b[i] - A[i][i - 1] \* b[i - 1]) / (A[i][i] - A[i][i - 1] \* A[i - 1][i])

    x = np.zeros(n)

    x[-1] = b[-1]

    for i in range(n - 2, -1, -1):

        x[i] = b[i] - A[i][i + 1] \* x[i + 1]

    return x

def spline\_built(dots):

    n = len(dots) - 1

    (x, y) = map(list, zip(\*dots))

    # ищем h\_i = x\_i - x\_i-1. Для поиска коэффицентов естественного сплайна(для поиска a\_i, b\_i, c\_i....)

    h = [None]

    for i in range(1, n + 1):

        h += [x[i] - x[i - 1]]

    A = [[None] \* n for i in range(n)]

    for i in range(1, n):

        for j in range(1, n):

            A[i][j] = 0.0

    for i in range(1, n - 1):

        A[i + 1][i] = h[i + 1]

    for i in range(1, n):

        A[i][i] = 2 \* (h[i] + h[i + 1])

    for i in range(1, n - 1):

        A[i][i + 1] = h[i + 1]

    F = []

    for i in range(1, n):

        F += [3 \* ((y[i + 1] - y[i]) / h[i + 1] - (y[i] - y[i - 1]) / h[i])]

    A = [A[i][1:] for i in range(len(A)) if i]

c = three\_diag\_solve(A, F)

    c = [0.0] + list(c) + [0.0]

    def evaluate(xdot):

        for i in range(1, len(x)):

            if x[i - 1] <= xdot <= x[i]:

                val = 0

                val += y[i]

                b = (y[i] - y[i - 1]) / h[i] + (2 \* c[i] + c[i - 1]) \* h[i] / 3

                val += b \* (xdot - x[i])

                val += c[i] \* ((xdot - x[i]) \*\* 2)

                d = (c[i] - c[i - 1]) / (3 \* h[i])

                val += d \* ((xdot - x[i]) \*\* 3)

                return val

        return None

    def output():

        print("Cubic spline:", '\n')

        for i in range(1, len(x)):

            val = 0

            b = (y[i] - y[i - 1]) / h[i] + (2 \* c[i] + c[i - 1]) \* h[i] / 3

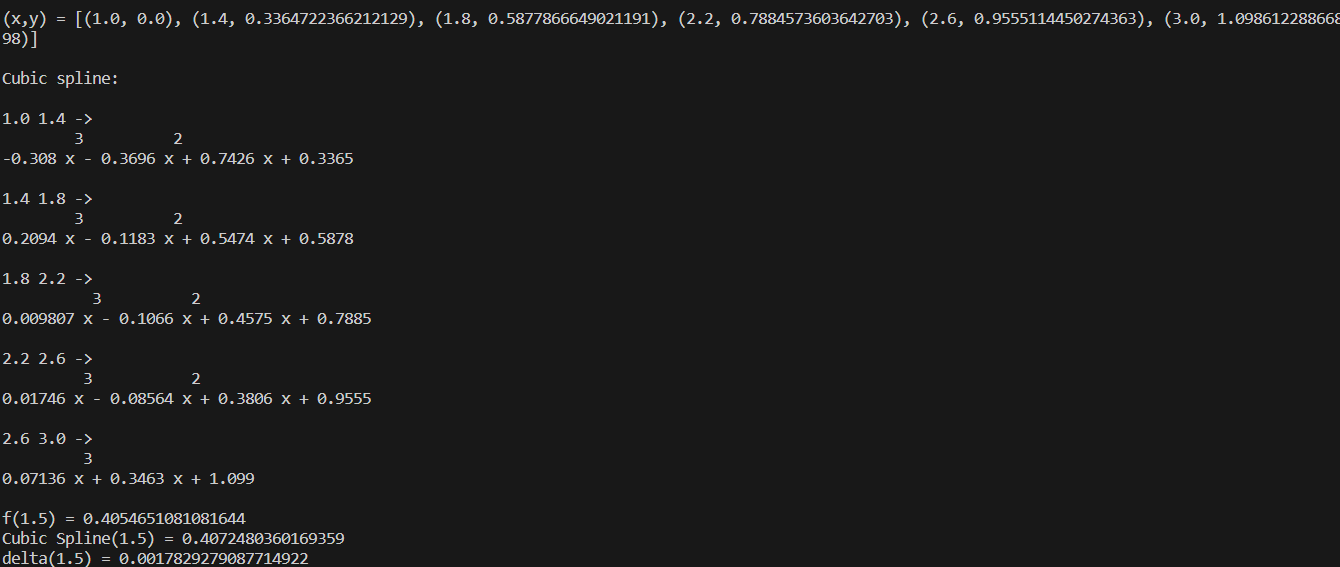
            d = (c[i] - c[i - 1]) / (3 \* h[i])

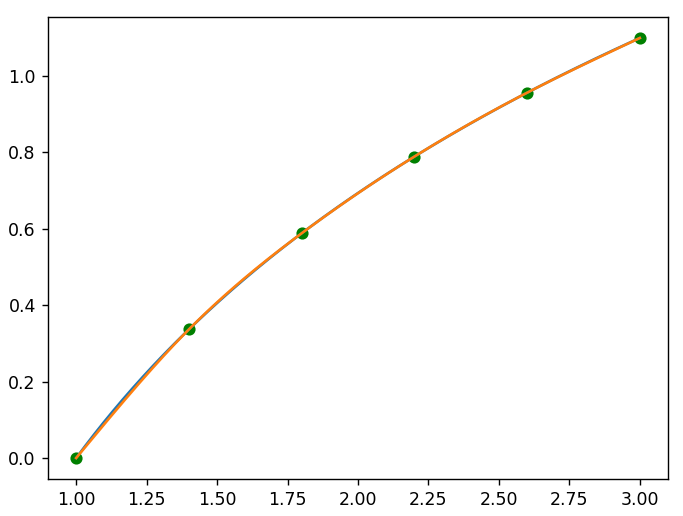
            print(x[i - 1], x[i], "->")

            print(np.poly1d([d, c[i], b, y[i]]), '\n')

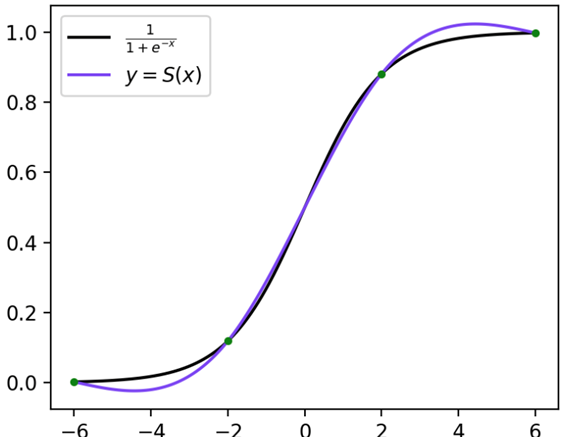
    return evaluate, output

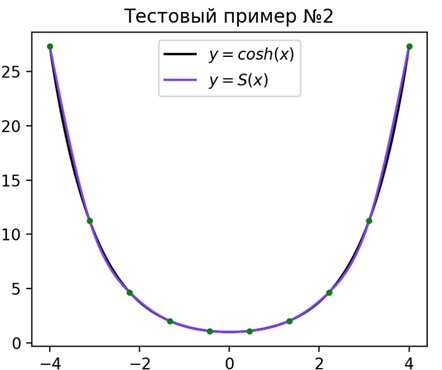
**Результаты работы программной реализации**

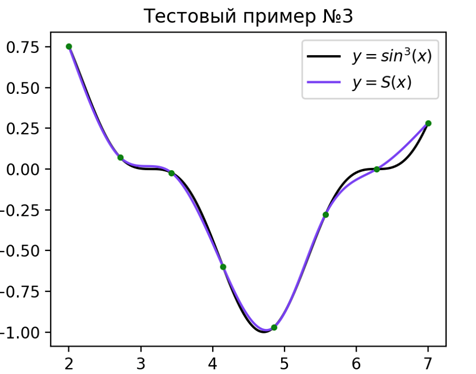
****



**Реализация тестовых примеров**







**ВЫВОД**

Таким образом, в ходе выполнения лабораторной работы была освоена

интерполяция функций с помощью кубических сплайнов. Составлена компьютерная программа, на тестовых примерах проверена правильность её

работы, построены кубические интерполяционные сплайны, вычислено значение функции в точке согласно заданному варианту. В заданной функции

и точке погрешность составляет меньшее значение, чем позволяют отобразить используемые средства для создания программы.