|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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.**

**«Интерполяция сплайнами»**

Студент **Леонов Владислав Вячеславович**

Группа **ИУ7-46Б**

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Леонов В.В.

*подпись, дата фамилия, и.о.*

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Градов В.М.

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Москва, 2021 г.*

**Оглавление**

[Цель работы 3](#_Toc68467393)

[Исходные данные 3](#_Toc68467394)

[Описание алгоритма 3](#_Toc68467395)

[Код программы 4](#_Toc68467396)

[Результаты работы 6](#_Toc68467397)

[Ответы на контрольные вопросы 7](#_Toc68467398)

# Цель работы

Построение и программная реализация алгоритма сплайн-интерполяции табличных функций.

# Исходные данные

1. Таблица функции с количеством узлов N. Задать с помощью формулы в диапазоне [0..10] c шагом 1.

|  |  |
| --- | --- |
| x | y |
| 0 | 0 |
| 1 | 1 |
| 2 | 4 |
| … | … |
| 9 | 81 |
| 10 | 100 |

# Описание алгоритма

Для решения поставленной задачи используется кубический сплайн – кривая, состоящая из множества полиномов третьей степени. Для определения коэффициентов полинома необходимо решить некоторую систему алгебраических уравнений, составные уравнения которой строятся по следующим соображениям:

1. В узлах многочлена и функции должны совпадать.
2. Во внутренних узлах значения первой и второй производной должны совпадать для обеспечения «гладкого» вида кривой.
3. Во внешних узлах полагаем, что вторая производная имеет нулевое значение.

Т.к. матрица данной системы трехдиагональна, то такую систему удобно решить методом прогонки.

# Код программы

Код программы представлен ниже.

|  |
| --- |
| Файл ***main.py*** |
| **import** io\_lib  **import** spline\_lib      **def** main() -> None:  data = io\_lib.data\_read(io\_lib.FILENAME\_IN)  splines = spline\_lib.calc\_splines(data)  x = float(input("X\_Value:"))  **for** i **in** range(1, len(splines)):  **if** splines[i].is\_x\_correct(x):  y = splines[i].value(x)  **print**("Y\_Value:", y)  **break**    **if** \_\_name\_\_ == "\_\_main\_\_":  main() |

|  |
| --- |
| Файл ***io\_lib.py*** |
| FILENAME\_IN = "data.txt"      **def** data\_read(filename: str) -> list:  data: list = []  f = open(filename, "r")  **for** line **in** f:  data.append(str\_to\_num\_arr(line))  f.close()  **return** data      **def** str\_to\_num\_arr(string: str) -> list:  **return** [float(value) **for** value **in** string.split(" ")] |

|  |
| --- |
| Файл ***spline\_lib.py*** |
| **class** Cube\_Spline():  **def** \_\_init\_\_(self, x\_l, x\_r, a, b, c, d):  self.x\_l = x\_l  self.x\_r = x\_r  self.a = a  self.b = b  self.c = c  self.d = d    **def** value(self, x):  **return** self.a + self.b \* (x - self.x\_l) + self.c \* (  x - self.x\_l)\*\*2 + self.d \* (x - self.x\_l)\*\*3    **def** is\_x\_correct(self, x):  **return** self.x\_l <= x < self.x\_r      **def** calc\_splines(data: list) -> list[Cube\_Spline]:  N = len(data) - 1    **def** \_h(i: int) -> float:  **return** data[i][0] - data[i - 1][0]    **def** \_ydelta(i: int) -> float:  **return** data[i][1] - data[i - 1][1]    **def** \_f(i: int) -> float:  **return** 3 \* (\_ydelta(i) / \_h(i) - \_ydelta(i - 1) / \_h(i - 1))    **def** \_xi(i: int) -> float:  **return** -\_h(i - 1) / (\_h(i - 2) \* XI\_LIST[i - 1] + 2 \*  (\_h(i - 1) + \_h(i - 2)))    **def** \_eta(i: int) -> float:  **return** (\_f(i - 1) - \_h(i - 2) \* ETA\_LIST[i - 1]) / (  \_h(i - 2) \* XI\_LIST[i - 1] + 2 \* (\_h(i - 1) + \_h(i - 2)))    **def** \_a():  **for** i **in** range(N):  splines.append(  Cube\_Spline(data[i][0], data[i + 1][0], data[i][1], 0, 0, 0))    **def** \_xi\_and\_eta():  **for** i **in** range(2, N + 1):  XI\_LIST.append(\_xi(i + 1))  ETA\_LIST.append(\_eta(i + 1))    **def** \_c():  splines[N].c = ETA\_LIST[N + 1]  **for** i **in** range(1, N):  splines[N - i].c = XI\_LIST[N - i + 1] \* splines[N - i +  1].c + ETA\_LIST[N - i + 1]    **def** \_b():  splines[N].b = \_ydelta(N) / \_h(N) - \_h(N) \* 2 \* splines[N].c / 3  **for** i **in** range(1, N):  splines[i].b = \_ydelta(i) / \_h(i) - \_h(i) \* (splines[i + 1].c -  2 \* splines[i].c) / 3    **def** \_d():  splines[N].d = -splines[N].c / (3 \* \_h(N))  **for** i **in** range(1, N):  splines[i].d = (splines[i + 1].c - splines[i].c) / (3 \* \_h(N))    splines: list[Cube\_Spline] = [" "]  XI\_LIST: list[float] = [0.0, 0.0, 0.0]  ETA\_LIST: list[float] = [0.0, 0.0, 0.0]  \_a()  \_xi\_and\_eta()  \_c()  \_b()  \_d()  **return** splines |

# Результаты работы

1. Значения функции при и .

|  |  |
| --- | --- |
| x | y |
| 0.5 | 0.34150552 |
| 5.5 | 30.91885359 |

2. Интерполяция полиномом Ньютона 3 степени при тех же значениях аргумента.

|  |  |
| --- | --- |
| x | y |
| 0.5 | 0.25 |
| 5.5 | 30.25 |

3. Истинное значение функции.

|  |  |
| --- | --- |
| x | y |
| 0.5 | 0.25 |
| 5.5 | 30.25 |

Таким образом, в данном конкретном случае полином Ньютона обеспечивает лучшее приближение, чем интерполяция сплайнами. Однако, как известно, коэффициенты полинома Ньютона вычисляются с большой погрешностью при больших степенях, точность расчета может заметно снизится, поэтому может быть лучше использовать интерполяцию кубическими сплайнами.

# Ответы на контрольные вопросы

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

Пусть задано два узла со значениями в них и соответственно.

***2. Выписать все условия для определения коэффициентов сплайна, построенного на трех точках.***

Пусть задано 3 узла со значениями в них , и соответственно.

***3. Определить начальные значения прогоночных коэффициентов, если принять, что для коэффициентов сплайна справедливо .***

***4. Написать формулу для определения последнего коэффициента сплайна , чтобы можно было выполнить обратный ход метода прогонки, если в качестве граничного условия задано , где k, m, p – заданные числа.***