Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Методы численного анализа»

ОТЧЕТ

к лабораторной работе №7

на тему:

«ИНТРЕПОЛЯЦИЯ СПЛАЙНАМИ»

БГУИР 1-40 04 01

Выполнил студент группы 253505
БЕКАРЕВ Станислав Сергеевич

(дата, подпись студента)

Проверил доцент кафедры информатики
АНИСИМОВ Владимир Яковлевич

(дата, подпись преподавателя)

Минск 2023

Содержание

Цель работы

- изучить построение кубических интерполяционных сплайнов ;
- составить программу построения кубических интерполяционных сплайнов;
- выполнить тестовые примеры и проверить правильность работы программы

ЗАДАНИЕ. Произвести интерполирование кубическими сплайнами приведенных в таблице функций. Вычислить значение сплайна в точке $x = 0.5*(b-a)$.

Значение сплайна в точке $x = 0.5*(b-a)$ записать в качестве ответа. Сравнить его со значением функции в соответствующей точке.

№ варианта	Функция $f(x)$	Интервал $[a,b]$ 50	Число узлов	Значение в точке $x = 0.5*(b-a)$
3.	$\sqrt{x}$	[0,4]	5	1,4065

Вариант 3

Исходные данные

```
Изначальная функция F(x): sqrt(x)
Узловые точки:
[0.0000 0.8000 1.6000 2.4000 3.2000 4.0000 ]
```

Программная реализация

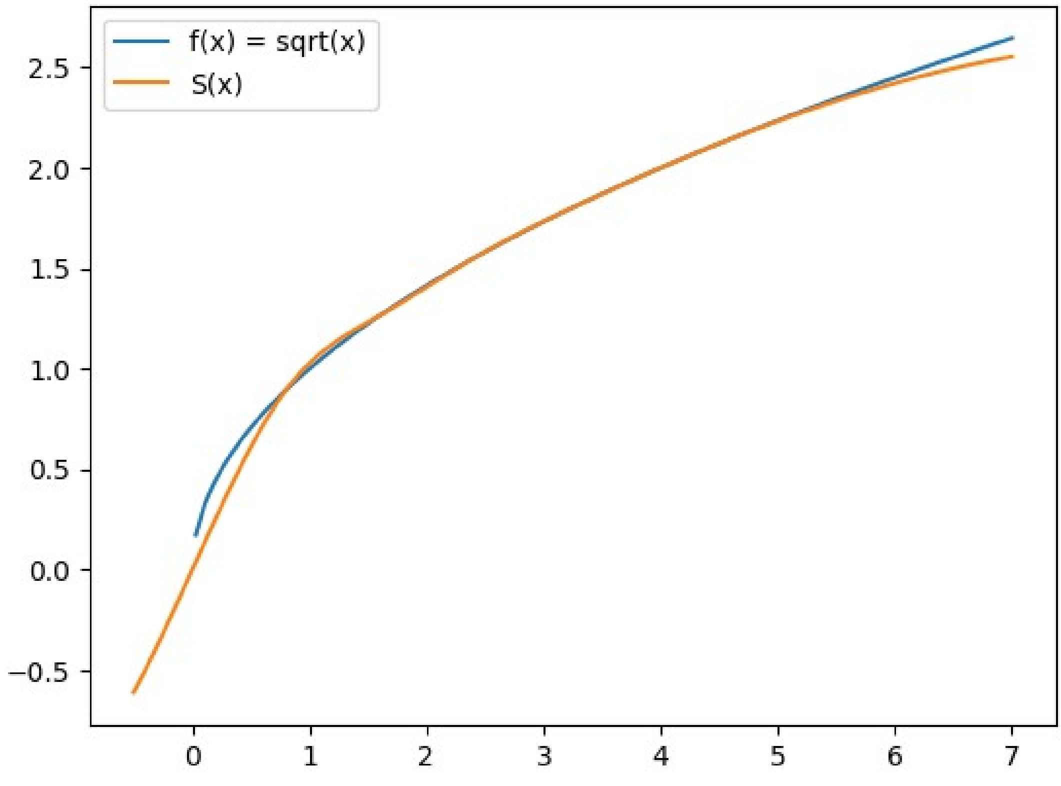
```
def spline_method(left, right, point_amount, func):
    point_amount += 2
    X = [], Y = [], A = [], B = [], C = [], D = [], S = []
    alpha = []
    betta = []
    for i in range(point_amount):
        X.append(left + (right - left)/(point_amount - 1) * i)
        Y.append(func.subs(x, X[i]))

    alpha.append(-1 * (X[2] - X[1]) / (2 * (X[2] - X[0])))
    betta.append((((Y[2] - Y[1]) / (X[2] - X[1])) - ((Y[1] -
Y[0]) / (X[1] - X[0]))) / (X[2] - X[0]) * 3 / 2)
    for i in range(1, point_amount - 2):
        a = (X[i + 1] - X[i]) / 3
        b = 2 * ((X[i + 2] - X[i]) / 3)
        c = (X[i + 2] - X[i + 1]) / 3
        d = (((Y[i + 2] - Y[i + 1]) / (X[i + 2] - X[i + 1])) -
((Y[i + 1] - Y[i]) / (X[i + 1] - X[i])))
        alpha.append(-1 * c / (a * alpha[i - 1] + b))
        betta.append((d - a * betta[i - 1]) / (a * alpha[i - 1]
+ b))
    alpha.reverse()
    betta.reverse()
    C.append(betta[0])
    for i in range(1, point_amount - 1):
        C.append(alpha[i - 1] * C[i - 1] + betta[i - 1])
    C.append(0.)
    C.reverse()
    C.append(0.)
    for i in range(point_amount - 1):
        B.append((Y[i + 1] - Y[i]) / (X[i + 1] - X[i]) - C[i] *
(X[i + 1] - X[i]) - ((C[i + 1] - C[i]) * (X[i + 1] - X[i]) / 3))
        D.append((C[i + 1] - C[i]) / (3 * (X[i + 1] - X[i])))
        A.append(Y[i])
    for i in range(point_amount - 1):
        f = A[i] + B[i]*(x - X[i]) + C[i]*(x - X[i])**2 +
D[i]*(x - X[i])**3
        S.append((f, x <= X[i+1]))
    S = Piecewise(*S)
    return S, X
```

Полученные результаты

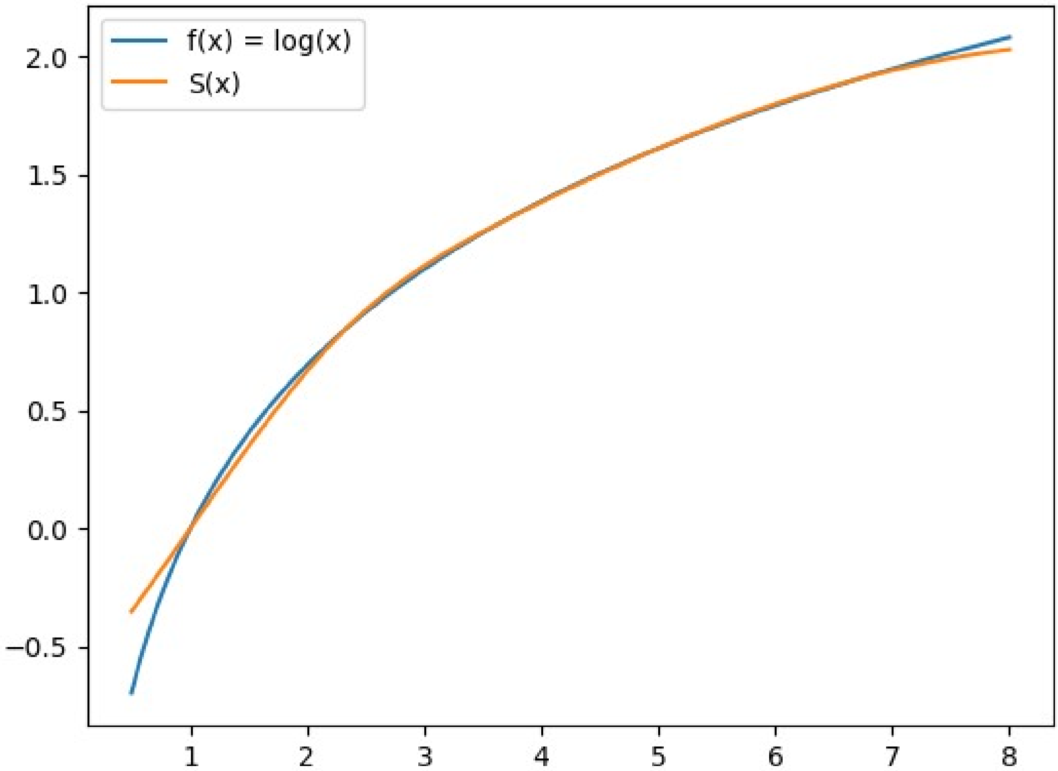
```
Изначальная функция F(x): sqrt(x)
Узловые точки:
[0.0000 0.8000 1.6000 2.4000 3.2000 4.0000 ]
Значения в точке x = (b - a) * 0.5: (2.0)
S(x) = 1.40654720409865
F(x) = 1.41421356237310
Разность F(x) - S(x) в данной точке:
0.00766635827444739
Значения в тестовой точке x: (1.0)
S(x) = 1.02762154181583
F(x) = 1.00000000000000
Разность F(x) - S(x) в данной точке:
-0.0276215418158334
```

Графики функций:



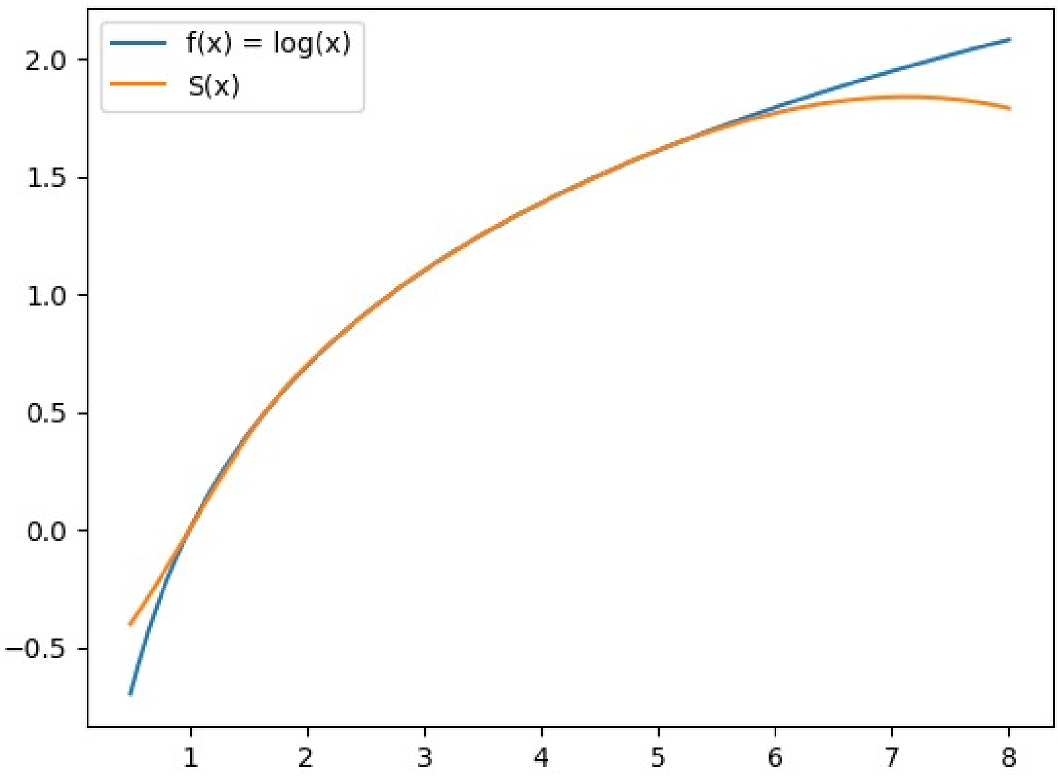
Тестовый пример 1. Четыре узла

```
Изначальная функция F(x): log(x)
Узловые точки:
[1.0000 2.3333 3.6667 5.0000 ]
Значения в точке x = (b - a) * 0.5: (2.0)
S(x) = 0.667211569287322
F(x) = 0.693147180559945
Разность F(x) - S(x) в данной точке:
0.0259356112726233
Значения в тестовой точке x: (2.0)
S(x) = 0.667211569287322
F(x) = 0.693147180559945
Разность F(x) - S(x) в данной точке:
0.0259356112726233
График функции:
```



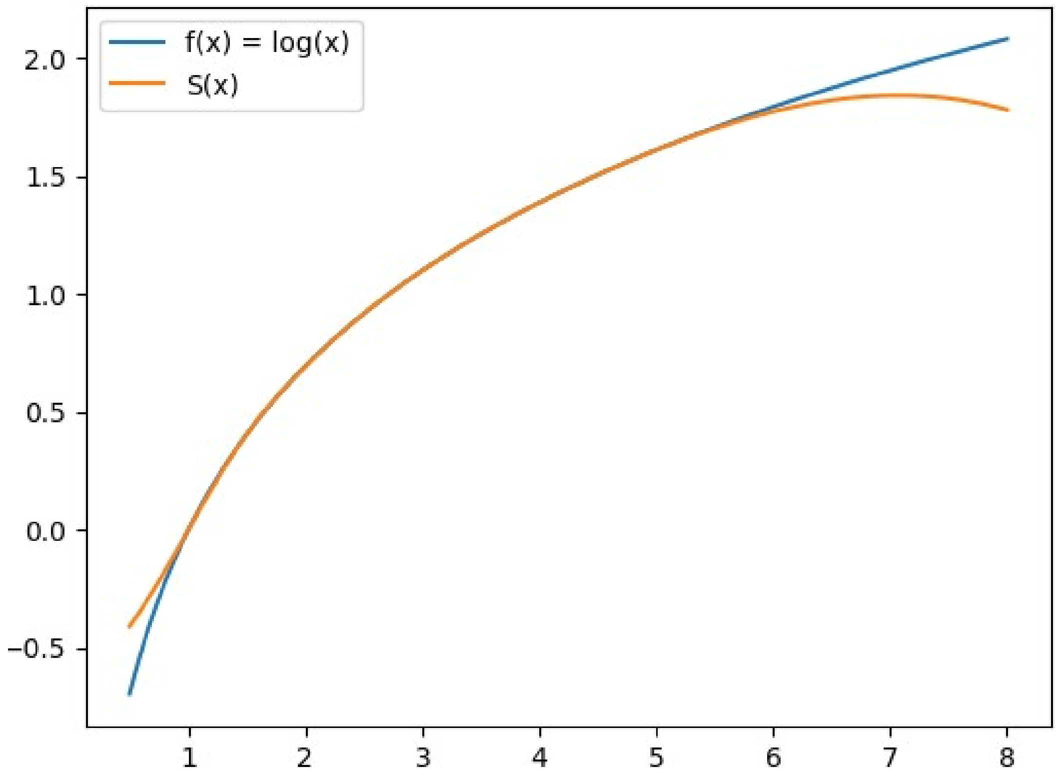
Тестовый пример 2. Семь узлов

```
Изначальная функция F(x): log(x)
Узловые точки:
[1.0000 1.6667 2.3333 3.0000 3.6667 4.3333 5.0000 ]
Значения в точке x = (b - a) * 0.5: (2.0)
S(x) = 0.697835515205212
F(x) = 0.693147180559945
Разность F(x) - S(x) в данной точке:
-0.00468833464526641
Значения в тестовой точке x: (2.0)
S(x) = 0.697835515205212
F(x) = 0.693147180559945
Разность F(x) - S(x) в данной точке:
-0.00468833464526641
График функции:
```



Тестовый пример 3. Десять узлов

```
Изначальная функция F(x): log(x)
Узловые точки:
[1.0000 1.4444 1.8889 2.3333 2.7778 3.2222 3.6667 4.1111 4.5556 5.0000 ]
Значения в точке x = (b - a) * 0.5: (2.0)
S(x) = 0.692581523790041
F(x) = 0.693147180559945
Разность F(x) - S(x) в данной точке:
0.000565656769904388
Значения в тестовой точке x: (2.0)
S(x) = 0.692581523790041
F(x) = 0.693147180559945
Разность F(x) - S(x) в данной точке:
0.000565656769904388
График функции:
```



Вывод

В ходе выполнения лабораторной работы я изучил построения кубических интерполяционных сплайнов, написал программу их реализации на языке Python, правильность работы программы проверил на тестовых примерах.

На основании тестов можно сделать следующие выводы:

- Достоинством построения кубических интерполяционных сплайнов является то, что на каждом из отрезков (сплайнов) многочлен имеет определенную степень(в данном случае третью степень), что облегчает нахождение значения функции в каждой из точек.
- Точность интерполяции функции увеличивается с ростом количества узлов (точек Xi) отобранных для построения кубических интерполяционных сплайнов