**МГТУ им. Н.Э. Баумана**

**ВЫЧИСЛИТЕЛЬНЫЕ АЛГОРИТМЫ**

**Лабораторный практикум №3**

**по теме: «*Построение и программная реализация алгоритма сплайн-интерполяции табличных функций*.»**

***Студент: Нгуен Фыок Санг***

***Группa: ИУ7И-46***

***Преподаватель: Градов В.М.***

2020

**Цель работы**. Получение навыков владения методами интерполяции таблично заданных функций с помощью кубических сплайнов.

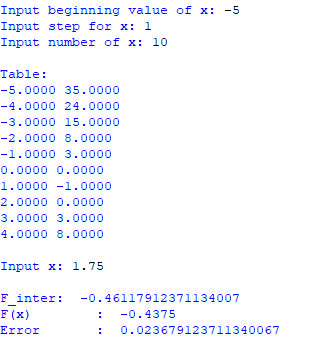
**Исходные данные.**

1. Таблица функции с количеством узлов N

2. Значение аргумента x.

**Результат работы программы.**

Значения y(x).



**Алгоритм:**

Кубический сплайн — это кривая, состоящая из „состыкованных“ полиномов третьей степени (y (IV ) (x) = 0). В точках стыковки значения и производные двух соседних полиномов равны.

На каждом отрезке [xi-1, xi], (i = 1..N) функция φ(x) есть полином третьей степени φi(x) в виде: φi(x) = ai + bi(x – xi – 1) + ci(x – xi -1)2 + di(x – xi – 1)3

* yi – 1 = ai (1)
* yi = ai +bihi + cihi2 + dihi3 (2)
* φ’i(x) = bi + 2ci(x – xi – 1) + 3di(x – xi -1)2
* φ’’i(x) = 2ci + 6di(x – xi – 1)
* bi+1 = bi + 2cihi + 3dihi2 (3)
* ci+1 = ci + 3dihi (4)
* φ’1(x0) = 0 => c1 = 0 (5)
* φ’’N(xN) = 0 => cN + 3dN = 0 (6)

Решение системы:

С1 = 0

hi-1ci – 1 + 2(hi – 1+hi)ci + hici+1 = 3()

cN-1 = 0

* Aiy­i -1 – Biyi + Diyi+1 = -Fi (7)
* K0y0 + M0y1 = P0 (7’)
* KNyN +MNyN+1 PN (7’’) , (i = 1..N – 1)
* yi = ξi+1yi+1 +ηi+1 (8)
* Ai ξiyi +Aiηi – Biyi +Diyi+1 = -Fi
* Yi = (9)
* ξi+1 = , ηi+1 = (10)

Найдем начальное значение коэффициент:

Из (7’) => y0 =

Из (8) => y0  = ξ1y1 +η1

Отсюда ξ1 = , η1 =

ci = ξi+1ci+1+ ηi+1 , c1 = 0

c1 = ξ2c2+ η2 , т.е η2  = 0, c2 = 0

ai = yi-1

di =

bi =

yi = ai +bihi + cihi2 + dihi3

**Ответы на вопросы:**

1. **Выписать значения коэффициентов сплайна, построенного на двух точках .**

Сплайн построенный на 2 точках и есть прямая линия. Соответственно коэффициенты с и d становятся равными нулю(с = 0, d = 0). Коэффициент а[i] = y[i — 1]. А коэффициент

b = (y[i] — y[i-1]/(x[i] — x[i -1])).

1. **Выписать все условия для определения коэффициентов сплайна, построенного на 3-х точках**

Для определения коэффициенты сплайна для 3-х точек, необходимо чтоб выполнялись следующие условия:

* ξ2 ``(x0) = 0
* ξ2 ``(x2) = 0
* η1`(x1) = η2`(x1)
* η1``(x1) = η2``(x1)
* ξ1(x0) = y0
* ξ1(x1) = y1
* ξ2(x1) = y1
* ξ2(x2) = y2

1. **Определить начальные значения прогоночных коэффициентов, если принять, что для коэффициентов сплайна справедливо C1=C2**.

Для того чтобы найти начальные прогоночные коэффициенты и приняв в условие то, что с1 = с2, имеем: Воспользуемся формулой y [i]= ξ [i+1] \* y [i+1] + η [i+1].

Для поиска коэффициента сплайна, знаем что с[i] и y[i] совпадают, то есть прямо пропорциональны.

Отсюда и получаем выражение: с[i]= ξ [i+1] \* с [i+1] + η [i+1]. Следовательно получаем с1 = ξ2 \* с2 + η2. Зная, что с1 = с2, получаем что с1 = ξ2 \* с1+ η2. Отсюда следует что ξ = 1 и η = 0.

1. **Написать формулу для определения последнего коэффициента сплайна СN, чтобы можно было выполнить обратный ход метода прогонки, если задано kCN-1+mCN=p, где k,m и p - заданные числа.**

Преобразовав формулу указанной в пункте 3 получим, c[n — 1] = ξ[i] \* c[n] + η[i].

Зная начальное условие k \* c[n—1] = -m \* c[n] +p, получим что k = 1, m = -1, p = 1.

Подставив эти вычисленные значения в начальное условие получим, что С[n] = С[n -1] — 1.

**Код программы:**

from math import cos

def f(x):

return x\*x - 2 \* x

def get\_table(x\_beg, step, amount):

x\_tbl = [x\_beg + step\*i for i in range(amount)]

y\_tbl = [f(x) for x in x\_tbl]

return x\_tbl, y\_tbl

def print\_table(x, y):

length = len(x)

for i in range(length):

print("%.4f %.4f" % (x[i], y[i]))

print()

def interpolate(x, y, x\_value):

n = len(x)

i\_near = min(range(n), key = lambda i: abs(x[i] - x\_value)) # index of nearest value

h = [0 if not i else x[i] - x[i - 1] for i in range(n)] # step value

A = [0 if i < 2 else h[i-1] for i in range(n)]

B = [0 if i < 2 else -2 \* (h[i - 1] + h[i]) for i in range(n)]

D = [0 if i < 2 else h[i] for i in range(n)]

F = [0 if i < 2 else -3 \* ((y[i] - y[i - 1]) / h[i] - (y[i - 1] - y[i - 2]) / h[i - 1]) for i in range(n)]

si = [0 for i in range(n + 1)]

eta = [0 for i in range(n + 1)]

for i in range(2, n):

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

eta[i + 1] = (A[i] \* eta[i] + F[i]) / (B[i] - A[i] \* si[i])

c = [0 for i in range(n + 1)]

for i in range(n - 2, -1, -1):

c[i] = si[i + 1] \* c[i + 1] + eta[i + 1]

a = [0 if i < 1 else y[i-1] for i in range(n)]

b = [0 if i < 1 else (y[i] - y[i - 1]) / h[i] - h[i] / 3 \* (c[i + 1] + 2 \* c[i]) for i in range(n)]

d = [0 if i < 1 else (c[i + 1] - c[i]) / (3 \* h[i]) for i in range(n)]

return a[i\_near] + b[i\_near] \* (x\_value - x[i\_near - 1]) + c[i\_near] \* ((x\_value - x[i\_near - 1]) \*\* 2) + d[i\_near] \* ((x\_value - x[i\_near - 1]) \*\* 3)

x\_beg = float(input("Input beginning value of x: "))

x\_step = float(input("Input step for x: "))

x\_amount = int(input("Input number of x: "))

x\_tbl, y\_tbl = get\_table(x\_beg, x\_step, x\_amount)

print("\nTable:")

print\_table(x\_tbl, y\_tbl)

x = float(input("Input x: "))

# Results

found = interpolate(x\_tbl, y\_tbl, x)

print("\nF\_inter: ", found)

print("F(x) : ", f(x))

print("Error : ", fabs(f(x) - found), "\n")