|  |  |
| --- | --- |
| 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** |

**Факультет «ИУ9»**

**Отчет по лабораторной работе № 2**

**«Сплайн-интерполяция»**

**по курсу «Численные методы»**

Выполнила **Балтаева М.**

Группа **62Б**

Преподаватель **Домрачева А.Б.**

**2023г.**

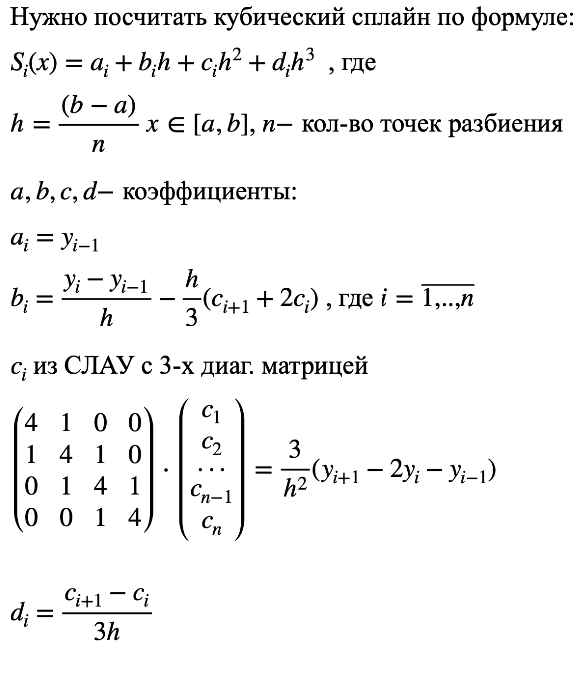
**Цель работы**

Целью данной лабораторной работы является написание программы для вычисления сплайнов по функции на рассматриваемом отрезке и с заданным количеством разбиений

**Индивидуальный вариант**

Номер варианта 2

**Постановка задачи**



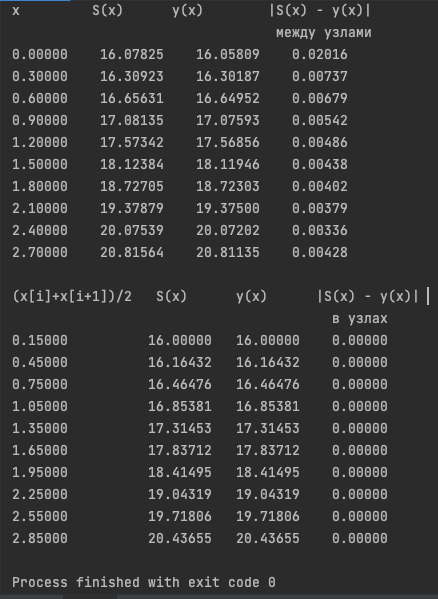
**Реализация**

def func(x):  
 return x \* (x \*\* 0.5) + 16  
  
def spline(i, x, xi, a\_coef, b\_coef, c\_coef, d\_coef):  
 return a\_coef[i] + b\_coef[i] \* (x - xi[i]) + c\_coef[i] \* ((x - xi[i]) \*\* 2) + d\_coef[i] \*( (x - xi[i]) \*\* 3)  
  
def main():  
 from\_val = 0  
 to\_val = 3  
 n = 10  
 h = (to\_val - from\_val) / n  
  
 curx = from\_val  
 yi = [0 for k in range(0, n+1)]  
 i = 0  
  
 while (curx <= to\_val):  
 yi[i] = (curx \* ((curx) \*\* 0.5) + 16)  
 curx += h  
 i += 1  
 # print("y: ", yi)  
  
 xi = [0 for k in range(0, n+1)]  
  
 curx = from\_val  
 i = 0  
  
 while (curx <= to\_val):  
 xi[i] = curx  
 curx += h  
 i += 1  
 # print('x:', xi)  
  
 a\_coef = [0 for k in range(0, n+1)]  
 b\_coef = [0 for k in range(0, n+1)]  
 c\_coef = [0 for k in range(0, n+1)]  
 d\_coef = [0 for k in range(0, n+1)]  
 d\_res = [0 for k in range(0, n+1)]  
 for i in range(0, n):  
 d\_res[i] = (yi[i+1] - 2 \* yi[i] + yi[i-1]) / (h \* h)  
  
  
 alpha = [0 for k in range(n)]  
 beta = [0 for k in range(n)]  
 y = 4  
  
 alpha[1] = - 1 / y  
 beta[1] = d\_res[1] / y  
  
 for i in range(2, n):  
 y = 4 + 1 \* alpha[i - 1]  
 alpha[i] = -1 / y  
 beta[i] = (d\_res[i] - 1 \* beta[i - 1]) / y  
 # print(beta, alpha)  
  
 c\_coef[0] = 0  
 c\_coef[n] = 0  
 c\_coef[n-1] = beta[n-1]  
 for i in range(n-2, 0, -1):  
 c\_coef[i] = alpha[i] \* c\_coef[i+1] + beta[i]  
 # print(c\_coef)  
  
 for i in range(0, n-1):  
 a\_coef[i] = yi[i]  
 b\_coef[i] = ((yi[i+1] - yi[i]) / h) - (h / 3) \* (c\_coef[i+1] + 2 \* c\_coef[i])  
 d\_coef[i] = (c\_coef[i+1] - c\_coef[i]) / (3 \* h)  
 a\_coef[n-1] = yi[n-1]  
 b\_coef[n-1] = ((yi[n] - yi[n-1]) / h) - (2 \* h / 3) \* c\_coef[n-1]  
 d\_coef[n-1] = - c\_coef[n] / (3 \* h)  
  
 print('x S(x) y(x) |S(x) - y(x)| ')  
 print(' между узлами ')  
 for i in range(0, n):  
 spl = spline(i, xi[i], xi, a\_coef, b\_coef, c\_coef, d\_coef)  
  
 x = from\_val + (i+1/2) \* h  
 y = func(from\_val + (i+1/2) \* h)  
  
 # print(y)  
 spl1 = spline(i, x, xi, a\_coef, b\_coef, c\_coef, d\_coef)  
 # print(spl1, end=", ")  
 diff = abs(spl1 - y)  
 diff2 = abs(spl - yi[i])  
 # print(xi[i], spl, y, diff, sep=" ")  
 print(f"{xi[i]:-2.5f} {spl1:-10.5f} {y:-10.5f} {diff:-10.5f}")  
  
 print()  
 print('(x[i]+x[i+1])/2 S(x) y(x) |S(x) - y(x)| ')  
 print(' в узлах ')  
 for i in range(0, n):  
 spl = spline(i, xi[i], xi, a\_coef, b\_coef, c\_coef, d\_coef)  
  
 x = from\_val + (i+1/2) \* h  
 nel = (xi[i] +xi[i+1]) / 2  
 y = func(x)  
  
 # print(y)  
 spl1 = spline(i, x, xi, a\_coef, b\_coef, c\_coef, d\_coef)  
 # print(spl1, end=", ")  
 diff = abs(spl1 - y)  
 diff2 = abs(spl - yi[i])  
 # print(xi[i], spl, y, diff, sep=" ")  
 print(f"{x:-2.5f} {spl:-10.5f} {yi[i]:-10.5f} {diff2:-10.5f}")  
main()

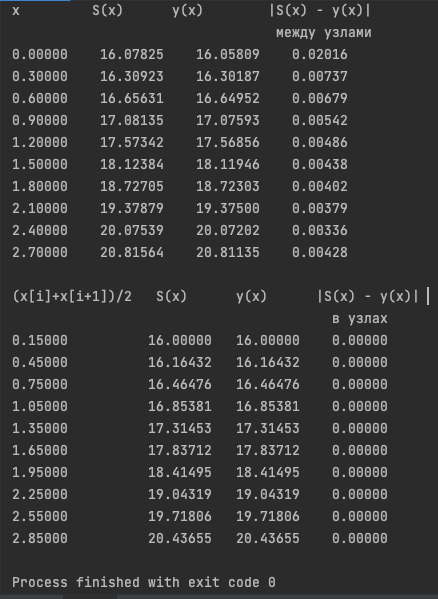
**Тестирование**

Вывод программы при n = 10

Между узлами



В узлах



**Вывод**

Реализована программа, вычисляющая сплайны с помощью метода прогонки, однако имеет место математическая погрешность. Как можно заметить, в узлах значения сплайна и функции совпадают.