**Программа**

//Выполнить аппроксимацию заданной табличной функции y(x) на отрезке x∈[xmin; xmax].

//7.Сплайн-интерполяция

#include <iostream>

#include <vector>

#include <cmath>

#include <fstream>

using namespace std;

void main() {

//вводим координаты точек

ifstream input;

input.open("Data.txt", ios::in);

int size;

input >> size;

float\* x = new float[size];

for (int i = 0; i < size; i++)

input >> x[i];

const float min = x[0];

const float max = x[size-1];

float\* y = new float[size];

for (int i = 0; i < size; i++)

input >> y[i];

input.close();

//начинаем разбиение отрезка x --> xx

vector<float> xx; //получим сейчас

vector<float> yy; //решение задачи, которое необх. найти

float step = 0.1;

float value = x[0];

int num = (max - min) / step; //число всех точек

for (float i = 0; i <= num; i++) {

xx.push\_back(value);

value += step;

}

int size\_xx = xx.size();

//приращение арг. в т. по x и у

vector<float> dx, dy;

for (int i = 0; i < size - 1; i++) {

float temp\_dx = x[i + 1] - x[i];

dx.push\_back(temp\_dx);

float temp\_dy = y[i + 1] - y[i];

dy.push\_back(temp\_dy);

}

//инициализируем матрицу H и заполняем её нулями

float\*\* matrH;

matrH = new float\* [size];

for (int i = 0; i < size; i++)

matrH[i] = new float[size];

for (int i = 0; i < size; i++)

for (int j = 0; j < size; j++)

matrH[i][j] = 0;

//инициализируем и обнуляем вектор Y

float\* vecY = new float[size];

for (int i = 0; i < size; i++)

vecY[i] = 0;

//инициализируем и обнуляем вектор M

float\* vecM = new float[size];

for (int i = 0; i < size; i++)

vecM[i] = 0;

//начинаем нахождение элементов матрицы H и вектора Y по формулам

matrH[0][0] = 1;

matrH[size - 1][size - 1] = 1;

for (int i = 1; i < size - 1; i++) {

matrH[i][i - 1] = dx[i - 1];

matrH[i][i] = 2 \* (dx[i - 1] + dx[i]);

matrH[i][i + 1] = dx[i];

vecY[i] = 3 \* (dy[i] / dx[i] - dy[i - 1] / dx[i - 1]);

}

// Hm=y ==> (H|y)=(m):

//методом прогонки

int N1 = size - 1;

float yYy;

float\* A = new float[size];

float\* B = new float[size];

yYy = matrH[0][0];

A[0] = -matrH[0][1] / yYy;

B[0] = vecY[0] / yYy;

for (int i = 1; i < N1; i++) {

yYy = matrH[i][i] + matrH[i][i - 1] \* A[i - 1];

A[i] = -matrH[i][1] / yYy;

B[i] = (vecY[i] - matrH[i][i - 1] \* B[i - 1]) / yYy;

}

vecM[N1] = (vecY[N1] - matrH[N1][N1 - 1] \* B[N1 - 1]) / (matrH[N1][N1] + matrH[N1][N1 - 1] \* A[N1 - 1]);

for (int i = N1 - 1; i >= 0; i--)

vecM[i] = A[i] \* vecM[i + 1] + B[i];

//завершили метод прогонки.

//вывод матрицы H и массивов M и Y

cout << "matrH:" << endl;

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++)

cout << matrH[i][j] << " ";

cout << endl;

}

cout << "\n\tvecY:\t||\tvecM"<<endl;

for (int i = 0; i < size; i++)

cout <<"\t"<< vecY[i] << "\t||\t"<<vecM[i]<<endl;

//находим коэффициенты сплайн-интерполяции по формулам

vector<float> ai, bi, ci, di;

for (int i = 0; i < size - 1; i++) {

ai.push\_back(y[i]);

di.push\_back((vecM[i + 1] - vecM[i]) / (3 \* dx[i]));

bi.push\_back(dy[i] / dx[i] - dx[i] \* (2 \* vecM[i] + vecM[i + 1]) / 3);

ci.push\_back(vecM[i]);

}

//подинтервалы

vector<int> x\_, xx\_;

for (int i = 0; i < size; i++) {

int temp = x[i] / 0.1;

x\_.push\_back(temp);

}

for (int i = 0; i < size\_xx; i++) {

int temp = xx[i] / 0.1;

xx\_.push\_back(temp);

}

//в каждом подинтервале x[j] <= x[i] <= x[j+1] ...

for (int i = 0; i < size\_xx; i++) {

int k = 0;

for (int j = 0; j < size - 1; j++) {

if (xx\_[i] >= x\_[j] && xx\_[i] <= x\_[j + 1]) {

k = j;

break;

}

else if (xx[i] == x[size - 1])

k = size - 1;

}

//... создаем уравнение и находим значение интерполирующей функции

float temp = ai[k] + bi[k] \* (xx[i] - x[k]) + vecM[k] \* pow((xx[i] - x[k]), 2) + di[k] \* pow((xx[i] - x[k]), 3);

yy.push\_back(temp);

}

ofstream output;

output.open("Spline3.txt");

for (int i = 0; i < size\_xx; i++)

output << xx[i] << '\t' << yy[i] << std::endl;

output.close();

}

**Пример (содержимое файла “Data.txt”):**

10

0 3 5 7 9 11 12 13 14 15

0 1.2 1.7 2.0 2.1 2.0 1.8 1.2 1.0 1.6

**Вывод (содержимое файла “Spline3.txt”):**

0 0

0.1 0.0420432

0.2 0.0840727

0.3 0.126075

0.4 0.168036

0.5 0.209943

0.6 0.251782

0.7 0.293539

0.8 0.3352

0.9 0.376752

1 0.418182

1.1 0.459475

1.2 0.500618

1.3 0.541598

1.4 0.5824

1.5 0.623011

1.6 0.663418

1.7 0.703607

1.8 0.743564

1.9 0.783275

2 0.822727

2.1 0.861907

2.2 0.9008

2.3 0.939393

2.4 0.977673

2.5 1.01562

2.6 1.05324

2.7 1.09049

2.8 1.12738

2.9 1.16389

3 1.2

3.1 1.2357

3.2 1.2579

3.3 1.28623

3.4 1.31414

3.5 1.34162

3.6 1.36867

3.7 1.39529

3.8 1.42147

3.9 1.44721

4 1.4725

4.1 1.49734

4.2 1.52173

4.3 1.54566

4.4 1.56913

4.5 1.59213

4.6 1.61466

4.7 1.63672

4.8 1.6583

4.9 1.67939

5 1.7

5.1 1.72012

5.2 1.73974

5.3 1.75885

5.4 1.77745

5.5 1.79554

5.6 1.8131

5.7 1.83013

5.8 1.84662

5.9 1.86256

6 1.87795

6.1 1.89279

6.2 1.90705

6.3 1.92075

6.4 1.93386

6.5 1.94639

6.6 1.95833

6.7 1.96966

6.8 1.98039

6.9 1.99051

7 2

7.1 2.00887

7.2 2.02107

7.3 2.03066

7.4 2.03964

7.5 2.04799

7.6 2.05572

7.7 2.06284

7.79999 2.06934

7.89999 2.07523

7.99999 2.08051

8.09999 2.08518

8.2 2.08924

8.3 2.09269

8.4 2.09553

8.5 2.09778

8.6 2.09942

8.7 2.10046

8.8 2.1009

8.9 2.10075

9 2.1

9.1 2.09866

9.2 2.10536

9.3 2.10706

9.4 2.10806

9.5 2.10832

9.6 2.10782

9.7 2.10651

9.8 2.10436

9.9 2.10133

10 2.09739

10.1 2.0925

10.2 2.08663

10.3 2.07973

10.4 2.07179

10.5 2.06276

10.6 2.0526

10.7 2.04128

10.8 2.02876

10.9 2.01501

11 2

11.1 1.99307

11.2 1.98441

11.3 1.97357

11.4 1.96011

11.5 1.9436

11.6 1.92361

11.7 1.89969

11.8 1.8714

11.9 1.83832

12 1.8

12.1 1.74393

12.2 1.68349

12.3 1.62

12.4 1.55476

12.5 1.48909

12.6 1.42429

12.7 1.36167

12.8 1.30254

12.9 1.24821

13 1.19999

13.1 1.14211

13.2 1.09192

13.3 1.04969

13.4 1.0157

13.5 0.990222

13.6 0.973524

13.7 0.96588

13.8 0.967564

13.9 0.978848

14 1

14.1 1.03109

14.2 1.07131

14.3 1.11964

14.4 1.17508

14.5 1.2366

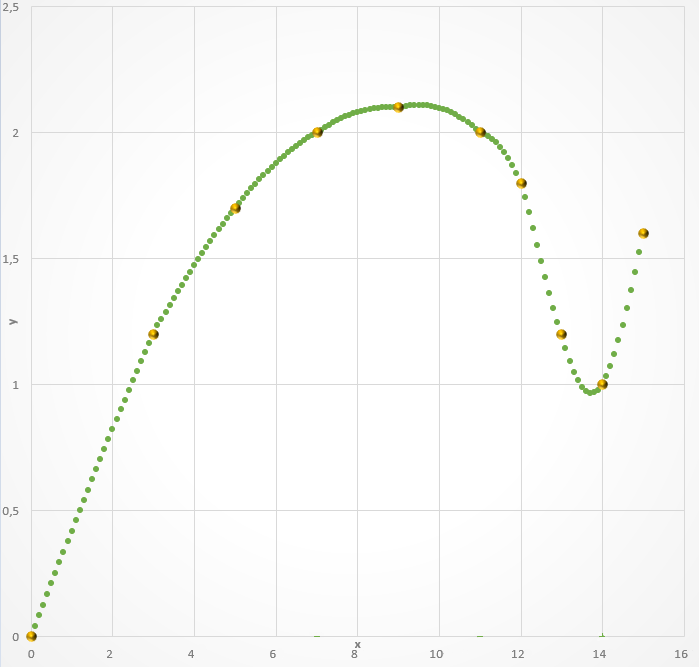
14.6 1.3032

14.7 1.37385

14.8 1.44755

14.9 1.52328

15 1.60002

**Визуализация примера в Excel:**  
  


**Теория:**

Теорией пользовался из учебника *«Самарский А.А., Гулин А.В. - Численные методы».*

**Изображение выглядит как текст

Автоматически созданное описание**

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Таким образом, мы можем построить систему линейных уравнений с m в качестве неизвестного, и мы знаем значения нескольких неизвестных, когда известны условия конечной точки.

*Конечное условие*

В соответствии с диапазоном значений **i** существует **n-1** формул, но существует **n + 1** неизвестных величин **c**. Чтобы решить эту систему уравнений, необходимы два других уравнения. Следовательно, необходимо добавить некоторые ограничения на дифференцирование двух концов **x[0]** и **x[n].** Воспользуемся следующим ограничением – свободной (естественной) границей:

Концы головы и хвоста не подвергаются воздействию силы, которая заставляет их сгибаться, то есть [image](https://images0.cnblogs.com/blog/477176/201301/26192708-ffd40879b19e41b1930c153fc653c171.png), Специально выражено как **c[0]=0** с участием **c[n]=0.**

Тогда решаемая система уравнений может быть записана в виде: **Hc=f**, где

H=

c=, f=