Вариант 4

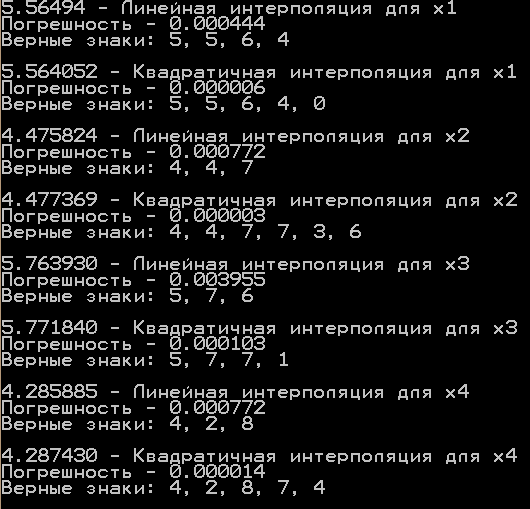
Используя первую или вторую интерполяционную формулу Ньютона, вычислить значения функции при данных значениях аргумента. При составлении таблицы разностей контролировать вычисления.

|  |  |
| --- | --- |
| x | y |
| 0,180 | 5,61543 |
| 0,185 | 5,46693 |
| 0,190 | 5,32634 |
| 0,195 | 5,19304 |
| 0,200 | 5,06649 |
| 0,205 | 4,94619 |
| 0,210 | 4,83170 |
| 0,215 | 4,72261 |
| 0,220 | 4,61855 |
| 0,225 | 4,51919 |
| 0,230 | 4,42422 |
| 0,235 | 4,33337 |

Значения аргумента:

|  |  |  |  |
| --- | --- | --- | --- |
| X1 | X2 | X3 | X4 |
| 0,1817 | 0,2275 | 0,175 | 0,2375 |

Результаты работы программы:



Программа

#include <iostream>

#include <vector>

#include <string>

#include <math.h>

#include <iomanip>

using namespace std;

void vern\_znaki(double pogr, double res)

{

int count = 0;

while (((int)(pogr \* 10)) == 0)

{

count++;

pogr \*= 10;

}

if (((int)(pogr \* 10)) > 5)

count--;

res \*= pow(10, count);

int r = (int)res;

int \*mas = new int[1000];

int it = 0;

while (r != 0)

{

mas[it++] = r % 10;

r /= 10;

}

cout << "Верные знаки: ";

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

{

if (i > 0)

cout << mas[i] << ", ";

else

cout << mas[i];

}

cout << endl << endl;

}

int main()

{

setlocale(LC\_ALL, "Russian");

vector<double> X(12);

vector<double> Y(12);

X[0] = 0.180f;

X[1] = 0.185f;

X[2] = 0.190f;

X[3] = 0.195f;

X[4] = 0.200f;

X[5] = 0.205f;

X[6] = 0.210f;

X[7] = 0.215f;

X[8] = 0.220f;

X[9] = 0.225f;

X[10] = 0.230f;

X[11] = 0.235f;

Y[0] = 5.61543f;

Y[1] = 5.46693;

Y[2] = 5.32634f;

Y[3] = 5.19304f;

Y[4] = 5.06649f;

Y[5] = 4.94619f;

Y[6] = 4.83170f;

Y[7] = 4.72261f;

Y[8] = 4.61855f;

Y[9] = 4.51919f;

Y[10] = 4.42422f;

Y[11] = 4.33337f;

vector <double> x(4);

x[0] = 0.1817f;

x[1] = 0.2275f;

x[2] = 0.175f;

x[3] = 0.2375f;

double raz11 = (Y[1] - Y[0]) / (X[1] - X[0]);

double raz12 = (Y[2] - Y[1]) / (X[2] - X[1]);

double raz21 = (raz12 - raz11) / (X[2] - X[0]); //2 разд разность для начальных

double raz110 = (Y[10] - Y[9]) / (X[10] - X[9]);

double raz111 = (Y[11] - Y[10]) / (X[11] - X[10]);

double raz22 = (raz111 - raz110) / (X[11] - X[9]); //2 разд разность для конечных

double raz13 = (Y[3] - Y[2]) / (X[3] - X[2]);

double raz23 = (raz13 - raz12) / (X[3] - X[1]);

double raz31 = (raz23 - raz21) / (X[3] - X[0]); //3 разд разность для начальных

double raz109 = (Y[9] - Y[8]) / (X[9] - X[8]);

double raz24 = (raz110 - raz109) / (X[10] - X[8]);

double raz32 = (raz22 - raz24) / (X[11] - X[8]); // 3 разд разность для конечных

for (int i = 0; i < 4; i++)

{

int count = 0;

double Pn1, Pn2, pogr1, pogr2;

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

{

if (x[i] > X[j])

{

count++;

}

}

if (count > (X.size()) / 2)

{

Pn1 = Y[11] + raz110 \* (x[i] - X[11]);

Pn2 = Y[11] + raz110 \* (x[i] - X[11]) + raz22 \* (x[i] - X[11]) \* (x[i] - X[10]);

pogr1 = (abs(raz22) / 2) \* (abs(x[i] - X[10]) \* (x[i] - X[11]));

pogr2 = (abs(raz32) / 6) \* (abs(x[i] - X[9]) \* (x[i] - X[10]) \* (x[i] - X[11]));

}

else

{

Pn1 = Y[0] + raz11 \* (x[i] - X[0]);

Pn2 = Y[0] + raz11 \* (x[i] - X[0]) + raz21 \* (x[i] - X[0]) \* (x[i] - X[1]);

pogr1 = (abs(raz21) / 2) \* (abs((x[i] - X[0]) \*(x[i] - X[1])));

pogr2 = (abs(raz31) / 6) \* (abs((x[i] - X[0]) \* (x[i] - X[1]) \* (x[i] - X[2])));

}

cout << Pn1 << " - Линейная интерполяция для x" << i + 1 << endl;

cout << fixed << setprecision(6) << "Погрешность - " << abs(pogr1) << endl;

vern\_znaki(abs(pogr1), Pn1);

cout << Pn2 << " - Квадратичная интерполяция для x" << i + 1 << endl;

cout << fixed << setprecision(6) << "Погрешность - " << abs(pogr2) << endl;

vern\_znaki(abs(pogr2), Pn2);

}

system("pause");

}