

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ им.
ПАТРИСА ЛУМУМБЫ**

**Факультет физико-математических и естественных наук
Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

дисциплина: Вычислительные методы

Студент:

Мажитов Магомед Асхабович

Группа:

НКНбд-01-21

МОСКВА

2023 г.

Цель:

Написать программу для расчета полинома Ньютона.

Теоретическая справка:

1. Построить равномерное разбиение отрезка $[a, b]$ на $N = 10$ частей точками $a = x_0, x_1, \dots, x_N = b$.
2. Рассчитать значения функции $f(x)$ в узлах интерполяции: $y_0 = f(x_0), y_1 = f(x_1), \dots, y_N = f(x_N)$.
3. Построить интерполяционный полином Ньютона $P_N(x) = A_0 + A_1(x - x_0) + A_2(x - x_0)(x - x_1) + \dots + A_i(x - x_0)\dots(x - x_{i-1}) + \dots + A_N(x - x_0)\dots(x - x_{N-1})$ согласно значениям из п.1, 2, также посчитать погрешность интерполяции в точке x_j .
4. Построить равномерное разбиение отрезка $[a, b]$ из задания на $M = 3N$ частей точками $a = \bar{x}_0, \bar{x}_1, \dots, \bar{x}_M = b$.
5. Посчитать значения исходной функции $f(x)$ из задания и построенного в п.3 полинома Ньютона $P_N(x)$ в точках $\bar{x}_0, \bar{x}_1, \dots, \bar{x}_M$, полученных в п.4, также посчитать погрешность интерполяции в точке \bar{x}_j .
6. Подобрать такое значение N , при котором максимальная погрешность меньше 0,1

Листинг:

```
#include <iostream>
#include <cmath>
#include <iomanip>

using namespace std;

//sqrt(x) - x, a = 0, b = 2.

double* split(double a, double b, int n){ //функция для разбиения отрезка на N
равных отрезков
    auto* s = new double [n]; //создаем массив для записи точек
    s[0] = a; //начальная и конечная точка константы
    double step = (b - a)/n; //считаем шаг
    for(int i = 1; i <= n; i++){ //проходим по циклу и к предыдущей точке
прибавляем шаг
        s[i] = s[i-1] + step;
    }
    return s; //возвращаем массив из узлов интерполяции
}

double* function(double* x, int n){ //функция для подсчета функции в узлах
интерполяции
```

```

    auto* f = new double [n]; //массив для записи функций
    for(int i = 0; i <= n; i++){//проходим по циклу и считаем функции в узлах
интерполяции
        f[i] = sqrt(x[i]) - x[i];
    }
    return f; //возвращаем массив из функций
}

double newtonPolinomial(const double *x, const double *y, int n, double
xx){//функция для подсчета полинома в точке
    double omg, newt = y[0], a;
    for(int i = 1; i < n; i++) { //проходим по циклу и считаем полином
        a = 0.0;//задаем начальное значение a
        for(int j = 0; j <= i; j++) { //проходим по циклу и считаем a
            omg = 1.0;//задаем начальное значение омеги
            for(int k = 0; k <= i; k++) { //проходим по циклу и считаем значение
омеги
                if (k != j) {
                    omg *= (x[j] - x[k]); //умножаем предыдущее значение омеги с
разницей между точками разбиения
                }
            }
            a += (y[j] / omg); //складываем предыдущее значение a на частное j-
ого элемента функции и значение омеги
        }
        for (int j = 0; j < i; j++) { //проходим по циклу и считаем новые
значения a для полинома
            a *= (xx - x[j]);
        }
        newt += a; //складываем значение a с предыдущим значением полинома
    }
    return newt; //возвращаем значение полинома
}

double* newtonPolinomialAll(double* x, double* y, int n){//функция для подсчета
полинома в каждой точке
    auto* newton = new double [n]; //создаем массив для записи полинома в каждой
точке
    for(int i = 0; i <= n; i++){//проходим по циклу и в каждой точке считаем
значения полинома и записываем в массив
        newton[i] = newtonPolinomial(x, y, n, x[i]);
    }
    return newton; //возвращаем массив
}

void print(double* pol, int n, double* x, double* y){//функция для красивого
вывода
    cout << "Dividing a segment into " << n << " parts: " << endl;
    for(int i = 0; i <= n; i++){
        cout << setprecision(4) << x[i] << setw(20);
        cout << y[i] << setw(20);
        cout << pol[i] << setw(20);
        cout << abs(y[i] - pol[i]) << setprecision(20) << endl;
    }
}

int main() {
    int n = 10, m = 3*n;
    double a = 0, b = 2;
    double *x = split(a, b, n);
    double *y = function(x, n);
    double *x1 = split(a, b, m);
    double *y1 = function(x1, m);
    double* ne = newtonPolinomialAll(x, y, n);

```

```

print(ne, n, x, y);
cout << endl;
double* nel = newtonPolynomialAll(x1, y1, m);
print(nel, m, x1, y1);
return 0;
}

```

Работа программы:

```

Консоль отладки Microsoft Visual Studio
Dividing a segment into 10 parts:
0      0      0      0
0.2    0.2472  0.2472  0
0.4    0.2325  0.2325  5.551e-17
0.6    0.1746  0.1746  1.388e-16
0.8    0.09443 0.09443  2.776e-17
1      0      5.551e-16  5.551e-16
1.2    -0.1046 -0.1046  2.276e-15
1.4    -0.2168 -0.2168  4.996e-15
1.6    -0.3351 -0.3351  1.377e-14
1.8    -0.4584 -0.4584  4.341e-14
2      -0.5858 -0.4286  0.1572

Dividing a segment into 30 parts:
0      0      0      0
0.06667 0.1915  0.1915  0
0.1333  0.2318  0.2318  2.776e-17
0.2      0.2472  0.2472  1.943e-16
0.2667  0.2497  0.2497  0
0.3333  0.244  0.244  4.718e-16
0.4      0.2325  0.2325  4.996e-16
0.4667  0.2165  0.2165  3.941e-15
0.5333  0.197  0.197  1.293e-14
0.6      0.1746  0.1746  2.09e-14
0.6667  0.1498  0.1498  1.771e-14
0.7333  0.123  0.123  3.347e-13
0.8      0.09443 0.09443  1.808e-12
0.8667  0.06428 0.06428  6.668e-12
0.9333  0.03276 0.03276  2.041e-11
1      0      -5.585e-11  5.585e-11
1.067  -0.03387 -0.03387  1.416e-10
1.133  -0.06875 -0.06875  3.478e-10
1.2    -0.1046  -0.1046  8.768e-10
1.267  -0.1412  -0.1412  2.355e-09
1.333  -0.1786  -0.1786  6.873e-09
1.4    -0.2168  -0.2168  2.13e-08
1.467  -0.2556  -0.2556  6.737e-08
1.533  -0.2951  -0.2951  2.092e-07
1.6    -0.3351  -0.3351  6.241e-07
1.667  -0.3757  -0.3757  1.777e-06
1.733  -0.4168  -0.4168  4.823e-06
1.8    -0.4584  -0.4584  1.252e-05
1.867  -0.5004  -0.5004  3.105e-05
1.933  -0.5429  -0.543  7.308e-05
2      -0.5858  -0.5105  0.07529

C:\Users\magom\source\repos\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe (процесс 2352) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно:

```

При $N=10$ погрешность интерполяции равна 0, таким образом подходит условию пункта №6.

Вывод:

В ходе работы я реализовал на языке C++ полином Ньютона.