Лабораторная работа № 4

студента группы ИТз-221

Дмитриева Дмитрия Анатольевича

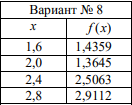
*Выполнение: \_\_\_\_\_\_\_\_\_ Защита: \_\_\_\_\_\_\_\_\_\_\_*

Интерполирование функций

*Цель работы*: найти значения функций с помощью интерполяционных многочленов Ньютона.

**Содержание работы**

1. Изучить теоретический материал.

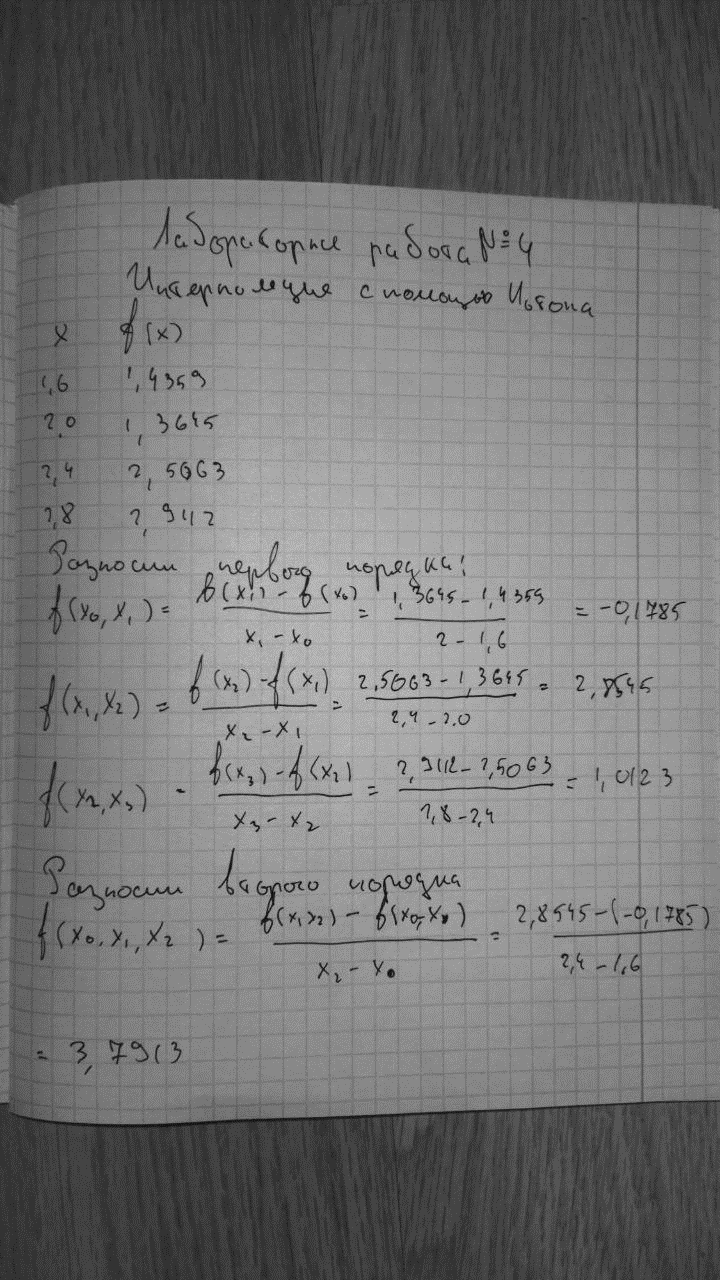


1. Выбрать индивидуальное задание согласно варианту, вычислить вручную значения функции для точки а1 с применением интерполяционного многочлена Ньютона.
2. Создать исходный модуль программы на языке высокого уровня Паскаль, реализующий вычисление значений функции при заданных значениях аргумента.
3. Оформить отчет.

**Ход работы:**

***Вариант - 8***

1. Изучил теоретический материал.
2. Вручную вычислил значения функции для точки а1 с применением интерполяционного многочлена Ньютона (квадратичной интерполяции) (рис. 1).



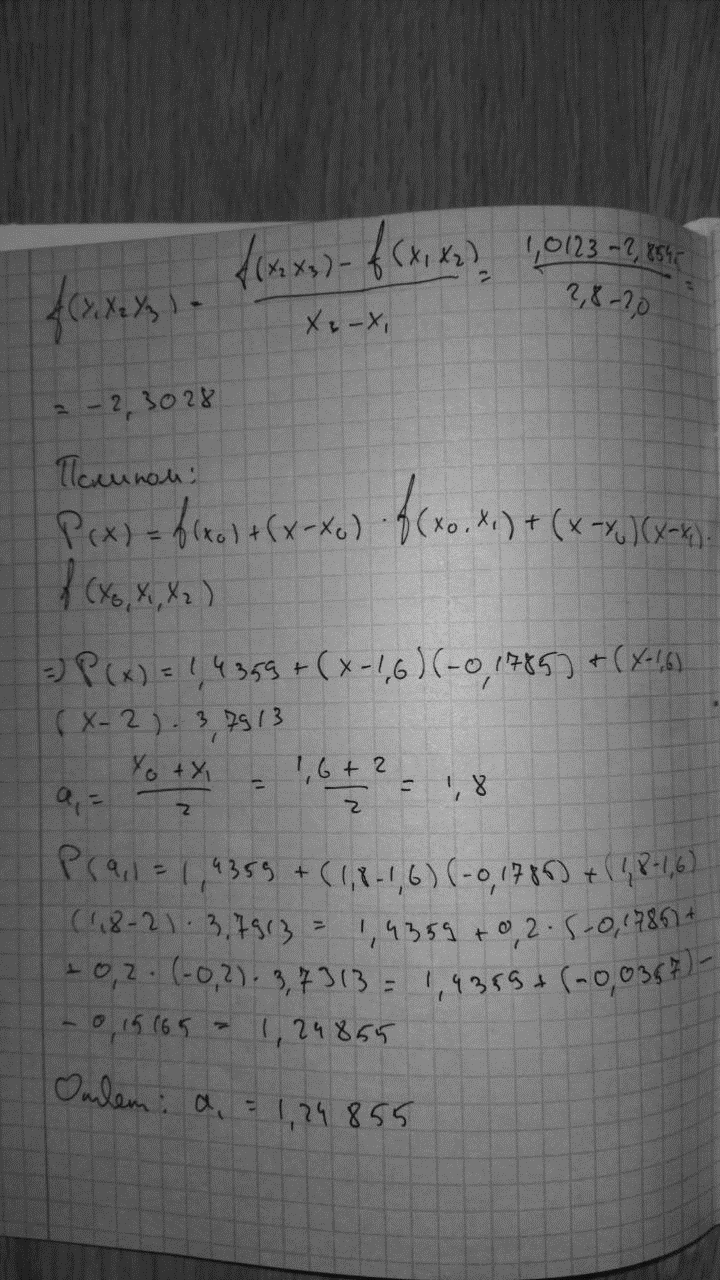


Рисунок 1 – Ручной расчет данный

1. Создал исходный модуль программы на языке высокого уровня C# (Приложение А), реализующий программу для вычисления значения функции для точки а1 с применением интерполяционного многочлена Ньютона. (рис. 2). Сверил результаты, значения сошлись.

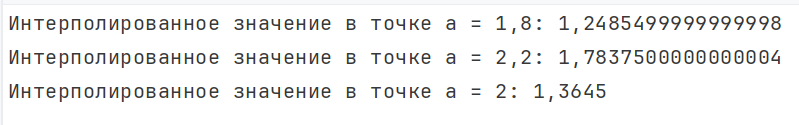


Рисунок 2 – Результат выполнения программы

1. Создал блок-схему будущей программы (Приложение Б)

**Контрольные вопросы**:

1. Задача интерполирования функции заключается в нахождении приближенного значения функции в точках, которые не принадлежат исходным данным, с использованием математического приближения. Это позволяет создать модель, которая может быть использована для оценки значений функции в промежуточных точках.
2. Конечная разность n-го порядка представляет собой последовательное вычисление разностей между значениями функции, начиная с разностей первого порядка, а затем переходя к разностям более высокого порядка. Пример разности первого порядка — это разница между значениями функции для двух соседних точек, а разность второго порядка — это разница между разностями первого порядка.
3. Разделенная разность n-го порядка — это разница между функциями, поделённая на соответствующие разности между их аргументами. Это понятие используется для построения интерполяционных многочленов и позволяет эффективно вычислять значения функции между заданными точками.
4. Интерполяционный многочлен Ньютона для конечных разностей строится с использованием последовательных разностей, начиная с первого порядка и продвигаясь к более высоким. Этот многочлен постепенно добавляет новые члены, что позволяет учитывать изменения функции с увеличением точности интерполяции.
5. Интерполяционный многочлен Ньютона для разделенных разностей использует последовательные разности, при этом каждый член включает разницу между значениями функции и их аргументами. Это позволяет эффективно вычислять более точные значения функции, особенно когда количество точек увеличивается.
6. Линейная интерполяция заключается в нахождении приближенного значения функции между двумя точками, используя прямую, которая соединяет эти точки. Это наиболее простой метод интерполяции, который используется для нахождения значений в промежуточных точках.
7. Другими методами интерполяции являются сплайны, метод наименьших квадратов, полиномиальная интерполяция, метод Лагранжа и другие. Эти методы используют различные подходы и могут быть выбраны в зависимости от задачи и требуемой точности.
8. Погрешность интерполяционной формулы Ньютона оценивается с помощью разностей между значением функции и значением, вычисленным по интерполяционному многочлену. Погрешность зависит от количества точек и порядка разностей, а также от величины промежутков между точками.

**Вывод**: научился определять значения функций с помощью интерполяционных многочленов Ньютона.

ПРИЛОЖЕНИЕ А

Код приложения

class Program

{

static void Main()

{

double[] x = { 1.6, 2.0, 2.4 };

double[] xFunction = { 1.4359, 1.3645, 2.5063 };

double function01 = (xFunction[1] - xFunction[0]) / (x[1] - x[0]);

double function12 = (xFunction[2] - xFunction[1]) / (x[2] - x[1]);

double function012 = (function12 - function01) / (x[2] - x[0]);

double a1 = (x[0] + x[1]) / 2;

double a2 = (x[1] + x[2]) / 2;

double a3 = (x[0] + x[2]) / 2;

double[] aList = { a1, a2, a3 };

foreach (var a in aList)

{

double result = xFunction[0] + (a - x[0]) \* function01 + (a - x[0]) \* (a - x[1]) \* function012;

Console.WriteLine($"Интерполированное значение в точке a = {a}: {result}");

}

}

}

ПРИЛОЖЕНИ Б

Блок схема

