# Общая информация

Программа calc\_tool выполняет вычисления на основе данных, представленных во входном файле и выводит результат в файл или на консоль. Calc\_tool — консольное приложение с использованием Qt.

Формат командной строки:

Calc\_tool.exe inputFile outputFile

inputFile — имя файла с входными данными.

outputFile — имя выходного файла. Если выходной файл не указан, вывод осуществляется на консоль.

Программа выполняет расчеты в следующих режимах

1. Вычисление определенного интеграла.
2. Решение обыкновенного дифференциального уравнения
3. Решение алгебраического уравнения.

# Формат входного файла.

Входной файл — текстовый, первая строка — идентификатор режима. Допустимые идентификаторы: itg, ode, ale — для режимов 1, 2 и 3 соответственно. Далее идут специфические для каждого из режимов данные. В файле могут быть пустые строки, их следует игнорировать. Ниже описаны специфические записи режимов.

## Вычисление интеграла

|  |  |
| --- | --- |
| Подынтегральное выражение | X^2 |
| Интервал интегрирования 1 | 0 1 |
| … |  |
| Интервал интегрирования N | 1 1.5 |

Пример файла: test\_itg.txt

## Решение обыкновенного дифференциального уравнения

|  |  |
| --- | --- |
| Правая часть | X\*y |
| Начальное условие | 0 1 |
|  | 1 |
| … | … |
|  | 3 |

Пример файла test\_ode.txt

## Решение алгебраического уравнения

|  |  |
| --- | --- |
| Левая часть | X^2-4 |
| Интервал/начальная точка 1 | 0 3 |
| … | … |
| Интервал/начальная точка N | 1 |

Пример файла test\_ale.exe

# Описание заданий

## Задание 1. Режим вычисления интеграла.

Следует

1. Прочитать данные из входного файла
2. Написать процедуры вычисления интеграла по формулам трапеций и Симпсона.
3. Вычислить интегралы от заданной функции для каждого интервала.
4. Вывести в выходной файл таблицу значений

|  |  |  |
| --- | --- | --- |
|  | Ф-ла трапеций | Ф-ла Симпсона |
| Интервал 1 | … | … |
| … | … | … |
| Интервал N | … | … |

Для вычисления подынтегрального выражения следует пользоваться объектами класса ExpressionCalc (см. задание 4).

Режим следует реализовать в функции Integrate.

## Задание 2. Режим решения ОДУ

Следует

1. Прочитать данные из входного файла.
2. Написать процедуры интегрирования задачи Коши для уравнения : явную и неявную схемы Эйлера.
3. Интегрированием уравнения найти значения функции в заданных точках.
4. Вывести в выходной файл таблицу значений

|  |  |  |
| --- | --- | --- |
|  | Явная схема | Неявная схема |
|  |  |  |
| … | … | … |
|  |  |  |

Для вычисления подынтегрального выражения следует пользоваться объектами класса ExpressionCalc (см. задание 4).

Для реализации неявной схемы Эйлера нужно решать алгебраическое уравнение. Для этого следует воспользоваться функциями FindRoot, написанными в рамках Задания 3.

Режим следует реализовать в функции SolveODE.

## Задание 3. Режим решения алгебраического уравнения.

Следует

1. Прочитать данные из входного файла.
2. Написать процедуры нахождения корней алгебраического уравнения : метод половинного деления и метод касательных.
3. Если во входном файле задан интервал, найти корень уравнения методом половинного деления, если начальная точка — методом касательных.
4. Вывести в выходной файл список найденных корней.

Для вычисления подынтегрального выражения следует пользоваться объектами класса ExpressionCalc (см. задание 4).

Режим следует реализовать в функции SolveAlgEqv.

Процедуры нахождения корня алгебраического уравнения следует реализовать в соответствующих функциях FindRoot.

## Задание 4. Калькулятор символьных выражений.

Следует написать класс, вычисляющий значение заданного символьного выражения. Класс описан в заголовочном файле expression\_calc.h и называется ExpressionCalc. Символьное выражение передается объекту ExpressionCalc через функцию SetExpression или с помощью конструктора. В последующем значение выражения вычисляется функцией Calculate. Например, пусть требуется вычислить значение функции в точках . Объекту ExpressionCalc необходимо передать строку выражения “x^3+1”, что может быть сделано так:

ExpressionCalc e;

e.SetExpression( “x^3+1” );

или так:

ExpressionCalc e( “x^3+1” );

Для вычисления значений функции следует вызвать Calculate для каждого x. Массив имен параметров (первый аргумент Calculate) будет содержать один элемент “x”, массив значений (второй аргумент) — соответствующее значение x (1,2 или 3).

Требования к калькулятору.

Калькулятор должен понимать , как минимум, следующие элементы синтаксиса:

1. Арифметические операции +,-,\*,/.
2. Возведение в степень ^ (x^y)
3. Скобки ()
4. Унарный минус

Калькулятор должен быть рассчитан на произвольное число параметров, передаваемых в функцию Calculate.