**Разработка Web-сервиса для решения математических операций**

**Цель работы:** Разработать web-сервис, который будет предоставлять возможность пользователю решать следующие математические задачи:

* Решение системы дифференциальных уравнений.
* Решение определенных интегралов.

**Задачи:** Данная работа подразумевает решение нескольких задач:

* Разработка математической библиотеки для решения универсального математического выражения.
* Разработка математической библиотеки для решения системы дифференциальных уравнений.
* Разработка математической библиотеки для решения интегралов.
* Разработка Web-сервиса для решения математических задач, перечисленных в списке для цели работы, с использованием предсавленных выше математических библиотек.

1. **Разработка математической библиотеки для решения системы дифференциальных уравнений**

Библиотека должна решать универсальную систему дифференциальных уранений следующими методами:

1. Метод Эйлера.
2. Метод Рунге-Кутты 2.
3. Метод Рунге-Кутты 4.
4. Метод Прогноза-коррекции.
5. Экстраполяционный метод Адамса 2, 3 и 4 порядков.
6. Метод Милна.

Для решения системы дифференциальныз уравнений библиотека должна принимать на вход следующие параметры:

1. Список дифференциальных уравнений в виде их правых частей:

,

где – правые части должны быть представлены в списке на вход.

1. Список параметров , представленных в левой части системы дифференциальных уравнений.
2. Переменная времени с ее название и начальным значением.
3. Список констант системы дифференциальных уравнений.
4. Конечное время расчетов.
5. Шаг вычислений по времени.

Библиотека должна возвращать документ в формате Microsoft Excel, в которм будут представлены следующие данные:

1. Начальная информация (включает все стартовые параметры и список методов, выбранных для расчетов).
2. Таблица общих результатов для сранения точности вычислений, пример представлен в таблице 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Euler | EulerAsyc | ForecastCorrection | ForecastCorrectionAsync |
| y | 2.405909636 | 2.405909636 | 2.408021685 | 2.408021685 |
| y1 | 4.482809632 | 4.482809632 | 4.48617188 | 4.48617188 |

Таблица 1. Пример общих результатов вычислений.

1. Таблица результатов времени вычисления каждым из методов и прилоденная к ней диаграмма. Пример представлен в таблице 2.

|  |  |  |  |
| --- | --- | --- | --- |
| Euler | EulerAsyc | ForecastCorrection | ForecastCorrectionAsync |
| 0.008 | 0.09 | 0.029 | 0.191 |

Таблица 2. Пример результатов времени вычисления.

Пример диаграммы придставлен на рисунке 1.

Рисунок 1. Диаграмма времени вычисления.

1. Для каждого метода создается отдельный лист Microsoft Excel, в котором указываются следующие значения:
   1. Конечные значения параметров.
   2. Время вычислений для данного метода.
   3. Таблица значений в каждую единицу времени. Сопровождается графиком изменения значений параметров во времени. Пример графика представлен на рис.2.

Рисунок 2. Пример графика именения значений параметров во времени.

Конструирование программного обеспечения для математической библиотеки решения системы дифференциальных уравнений.

На рис.3 представлена общая диаграмма классов, описывающая архитектуру данного приложения.

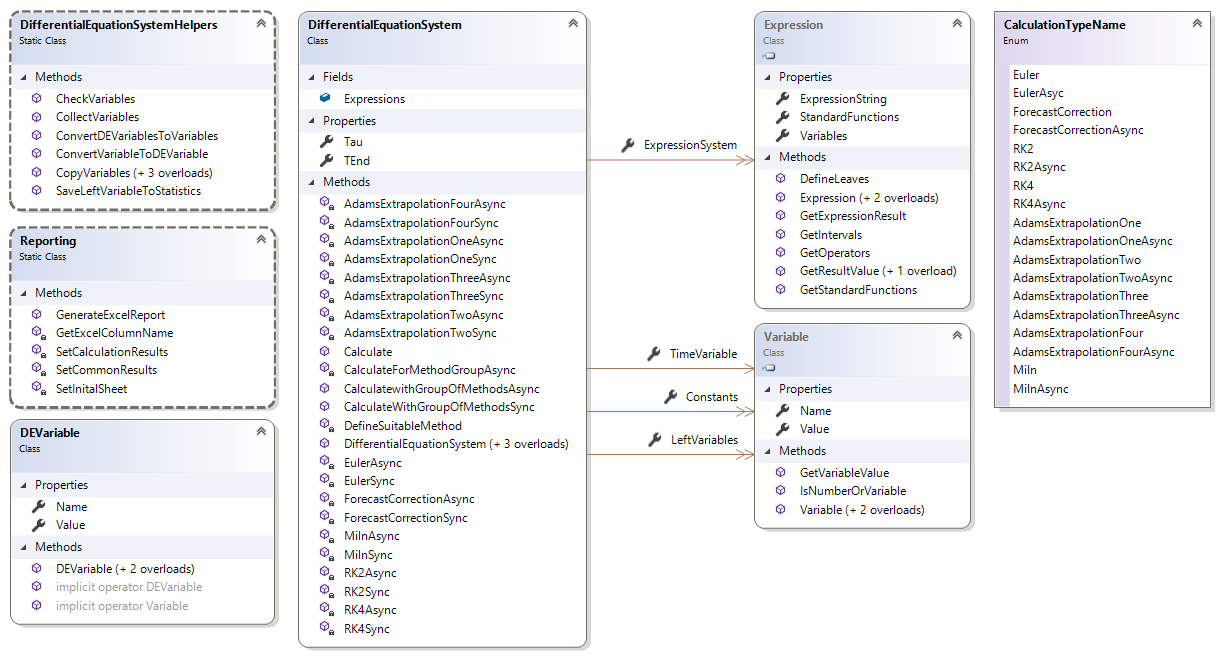


Рис.3. Диаграмма классов матемаической библиотеки решения системы дифференциальных уравнений.

В данной реализации объект класса «DifferentialEquationSystem» предоставляет функционал для решения системы дифференциальных уравнений, ниже представленны свойства данного класса (Таблица 3).

|  |  |
| --- | --- |
| Название | Назначение |
| public List<Expression> ExpressionSystem | Список, состоящий из правых частей системы дифференциальных уравнений. |
| public List<Variable> LeftVariables | Параметры системы дифференциальных уравнений (переменные, находящиеся в левой части уравнений). |
| public Variable TimeVariable | Параметр, обозначающий время в системе дифференциальных уравнений (Название и начальное значение.) |
| public List<Variable> Constants | Константы системы дифференциальных уравнений. |
| public double Tau | Шаг по времени, необходимый совершить за одну итерацию при расчете системы дифференциальных уравнений. |
| public double TEnd | Конечное время расчета системы дифференциальных уравнений. |
| public List<string> Expressions | Списк правых частей системы дифференциальных уравнений в формате строки. |

Таблица 3. свойства класса «DifferentialEquationSystem»

Публичные конструкторы класса «DifferentialEquationSystem»

1. public DifferentialEquationSystem() - Конструктор по умолчанию, инициализирует объект класса
2. public DifferentialEquationSystem(List<string> expressions, List<DEVariable> leftVariables, List<DEVariable> constants, DEVariable timeVariable, double tEnd, double tau) – Конструктор инициализирует объект и заполняет его начальными значениями для системы начального уравнения, где:
   1. List<string> expressions – правые части системы дифференциальных уравнений.
   2. List<DEVariable> leftVariables – переменные из левых частей системы дифференциальных уравнений.
   3. List<DEVariable> constants – константы системы дифференциальных уравнений.
   4. DEVariable timeVariable – переменная времени системы дифференциальных уравнений.
   5. double tEnd – конечное время расчетов системы дифференциальных уравнений.
   6. double tau – шаг по времени за одну итерацию расчета системы дифференциальных уравнений.
3. public DifferentialEquationSystem(DifferentialEquationSystem differentialEquationSystem) – Конструктор инициализирует объект и заполняет его полями другого экземпляра данного класса.

Основные методы расчета системы дифференциальных уравнений класса «DifferentialEquationSystem»

1. public double Calculate(CalculationTypeName calculationType, out List<DEVariable> results, List<List<DEVariable>> variablesAtAllStep = null) – Метод производит расчет системы дифференциальных уравнений одним указанным методом, где:
   1. CalculationTypeName calculationType – метод расчета системы дифференциальных уравнений.
   2. out List<DEVariable> results – список параметров, куда будут сохранены конечные значения системы дифференциальных уравнений.
   3. List<List<DEVariable>> variablesAtAllStep = null – таблица значений параметров системы дифференциальных уравнений, указаных в каждую единицу времени расчета.

Метод возвращает время выполнения программы в секундах.

1. public Dictionary<CalculationTypeName, double> CalculateWithGroupOfMethodsSync(List<CalculationTypeName> calculationTypes, out Dictionary<CalculationTypeName, List<DEVariable>> results, Dictionary<CalculationTypeName, List<List<DEVariable>>> variablesAtAllSteps = null) – Метод синхронно производит расчет системы дифференциальных уравнений несколькими указанными методами, где:
   1. List<CalculationTypeName> calculationTypes – список методов решения системы дифференциальных уравнений.
   2. out Dictionary<CalculationTypeName, List<DEVariable>> results – конечные значения системы дифференциальных уравнений для каждого метода расчета.
   3. Dictionary<CalculationTypeName, List<List<DEVariable>>> variablesAtAllSteps = null – таблица значений параметров системы дифференциальных уравнений, указаных в каждую единицу времени расчета для каждого расчетного метода.

Метод возвращает время выполнения программы в секундах для каждого расчетного метода.

1. public Dictionary<CalculationTypeName, double> CalculateWithGroupOfMethodsAsync(List<CalculationTypeName> calculationTypes, out Dictionary<CalculationTypeName, List<DEVariable>> results, Dictionary<CalculationTypeName, List<List<DEVariable>>> variablesAtAllSteps = null) – Метод асинхронно производит расчет системы дифференциальных уравнений несколькими указанными методами, где:
   1. List<CalculationTypeName> calculationTypes – список методов решения системы дифференциальных уравнений.
   2. out Dictionary<CalculationTypeName, List<DEVariable>> results – конечные значения системы дифференциальных уравнений для каждого метода расчета.
   3. Dictionary<CalculationTypeName, List<List<DEVariable>>> variablesAtAllSteps = null – таблица значений параметров системы дифференциальных уравнений, указаных в каждую единицу времени расчета для каждого расчетного метода.

Метод возвращает время выполнения программы в секундах для каждого расчетного метода.

Пример использования библиотеки расчетов системы дифференциальных уравнений на языке c# в консольном приложении .NET

Ниже представлен пример расчета системы дифференциальных уравнений методом Эйлера. Дифференциальное уравнение имеет вид:

Начальные значения имеют вид:

static void VerifyDifferentialEquations()

{

List<string> expressions = new List<string>

{

"2 \* y1 - y + time \* exp(time)",

"y1"

};

List<DEVariable> leftVariables = new List<DEVariable>

{

new DEVariable("y", 2),

new DEVariable("y1", 1),

};

DEVariable timeVariable = new DEVariable("time", 0);

DifferentialEquationSystem.DifferentialEquationSystem differentialEquationSystem = new DifferentialEquationSystem.DifferentialEquationSystem(expressions, leftVariables, null, timeVariable, 1.5, 0.001);

List<List<DEVariable>> perTime = new List<List<DEVariable>>();

double calcTime = 0;

List<DEVariable> resultEuler;

calcTime = differentialEquationSystem.Calculate(CalculationTypeName.Euler, out resultEuler, perTime);

}

Результат будет сохранен в переменную resultEuler, таблица значений в каждый момент времени расчета - perTime. Метод вернет время выполнения расчетов.

1. **Разработка математической библиотеки для решения определенных интегралов**

Библиотека должна решать универсальный определенный интеграл следующими методами:

1. Метод правых прямоугольников.
2. Метод левых прямоугольников.
3. Метод прямоугольников в среднем.
4. Метод трапеций.
5. Метод парабол (Симпсона).

Для решения интегралов система должна принимать на вход следующие значения:

1. Подынтегральное выражение.
2. Начальное значение интеграла.
3. Конечное значение интеграла.
4. Количество шагов разбиения.

Конструирование программного обеспечения для математической библиотеки решения интегралов.

На рис. 4. представлена диаграмма классов проекта решения интегралов.

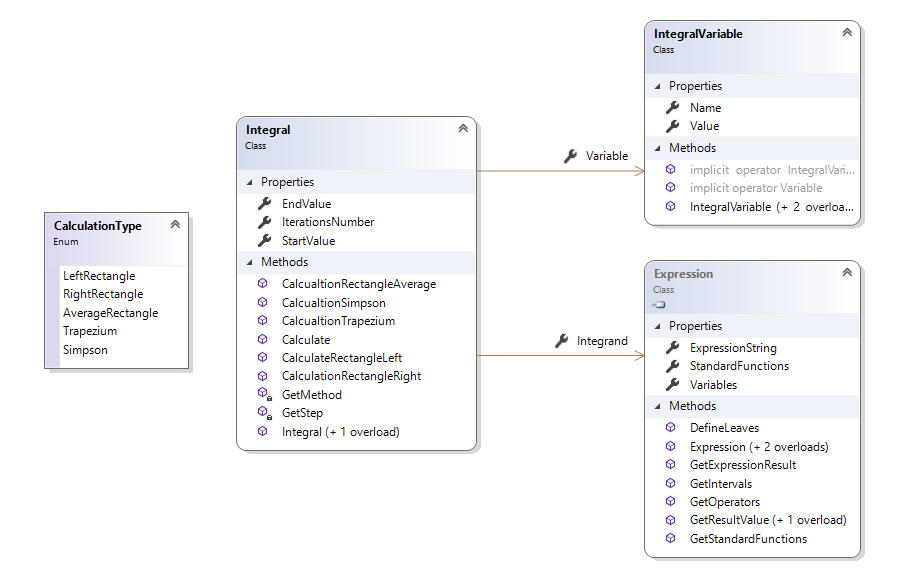


Рис.4. Диаграмма классов проекта решения интегралов.

В данной реализации объект класса «Integral» предоставляет функционал для решения интегралов, ниже представленны свойства данного класса (таблица 4).

|  |  |
| --- | --- |
| Название | Назначение |
| public Expression Integrand | Подынтегральное выражение |
| public double StartValue | Начальное значение определенного интеграла |
| public double EndValue | Конечное значение определенного интеграла |
| public int IterationsNumber | Количество итераций |
| public IntegralVariable Variable | Название переменной (имя). |

Таблица 4. Свойства класса «Integral»

Конструкторы класса «Integral» Конструктор

1. public Integral() – стандартный конструктор, инициализирует объект класса.
2. public Integral(string integrandExpression, double startValue, double endValue, int iterations, string parameterName) – Конструктор задает исходные параметры определенного интеграа для дальнейшего его решения, где:
   1. string integrandExpression – подэнтегральное выражение.
   2. double startValue – начальное значение интеграла.
   3. double endValue – конечное значение интеграла.
   4. int iterations – количество итераций расчета.
   5. string parameterName – имя интегрального параметра.

Пример использования библиотеки для расчета интегралов в консольном приложении .NET на языке c#.

Ниже представлен пример расчета интеграла методом трапеций, определенный интеграл имеет вид:

Решение имеет следующий вид:

string integrand = "x\*x";

double startValue = -1;

double endValue = 1;

int iterationsNumber = 1000;

string parameterName = "x";

Integral.Integral integral = new Integral.Integral(integrand, startValue, endValue, iterationsNumber, parameterName);

double resultTrapezium = integral.Calculate(CalculationType.Trapezium);

Сonsole.WriteLine($"Trapezium: {resultTrapezium}");

Console.ReadKey();

1. **Разработка библиотеки решения универсального математического выражения**

Библиотека должна решать заданное математическое выражение с указанными значениями параметрами данного выражения.

На вход данной библиотеке подаются следующие параметры:

1. Математическое выражение, которое необходимо расчитать.
2. Список переменных математического выражения и их значений.

Ниже представлена диаграмма классов библиотеки решения математических выражений (рис.5).

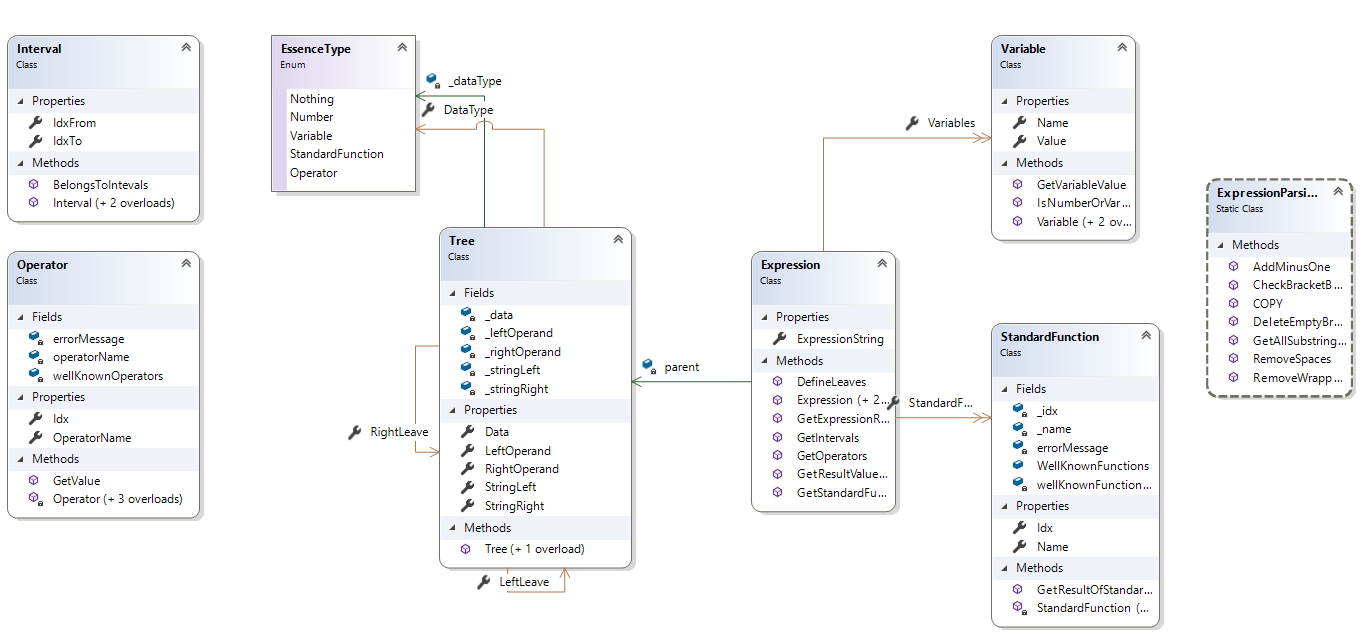


Рис.5. Диаграмма классов математической бибилотеки.

Объект класса “Expression” предоставляет набор инструментов для решения математического выражения.

Ниже пресдатвлены свойства данного класса.

|  |  |
| --- | --- |
| Название | Назначение |
| public List<StandardFunction> StandardFunctions | Список стандартных функций. |
| public List<Variable> Variables | Список переменных математического выражеия. |

Конструкторы класса “Expression”

1. public Expression(string expression, List<Variable> variables) – конструктор задает математическое выражение и его параметры.
2. public Expression(string expression) ) – конструктор задает математическое выражение без параметров.
3. public Expression() – стандартный конструктор без параметров.

Для расчета математического выражения используют следующие методы:

1. public double GetResultValue(List<Variable> variables) – Метод получения результата математического выражения. На вход подаются значения параматров данного выражения.
2. public double GetResultValue(Variable variable) – Метод получения результата математического выражения. На вход подается одно значение параматра данного выражения.

Пример использования математической библиотеки. Ниже представлен код программы консольного .NET приложения на языке C#, который производит вычисление математического выражения:

string expression = "(a + b) \* c";

List<Variable> vars = new List<Variable>

{

new Variable("a", 1),

new Variable("b", 3),

new Variable("c", 2)

};

double result = 8;

Expression e = new Expression(expression, vars);

Assert.AreEqual(result, e.GetResultValue(vars));