**Лабораторная работа №1 “Численное интегрирование”**

**по Вычислительной Математике**

**Постановка задачи**

* Вычислить определенный интеграл, используя различные численные методы и алгоритмы их реализации;
* Провести сравнительный анализ полученных;
* Сделать вывод.

**Формула**



**Цель работы**

Систематизация алгоритмов для решения задач, связанных с интегрированием и написание общей программы для их реализации.

**Ход работы**

**1) Разработка модуля с программным меню**

Содержание меню:

Основной файл содержит две функции: основную и подменю для интегрирования. В основной функции предлагается выбрать тип решаемой задачи. В подменю пользователь выбирает метод вычисления интеграла и алгоритм нахождения результата.

**2) Разработка модуля с основными функциями вычисления**

Существующие функции вычисления интеграла:

a) Метод прямоугольников левых частей

1) с постоянным шагом

2) двойной пересчет

b) Метод прямоугольников правых частей с постоянным шагом

c) Метод трапеций с постоянным шагом

d) Метод Симпсона (парабол) с постоянным шагом

**Функциональная часть программы**

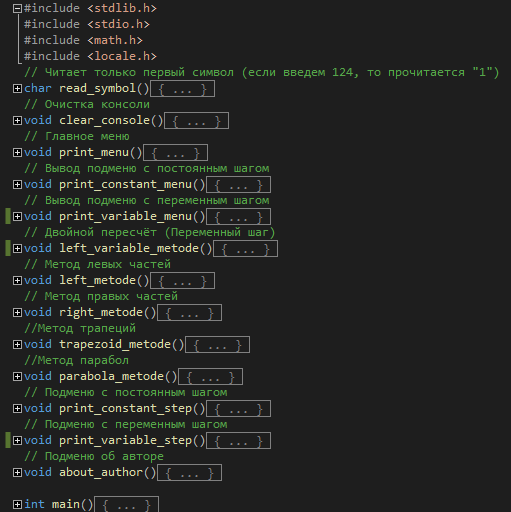
**Функционал меню**:

Пользователь может передвигаться по программе, выбирая варианты, нужные ему для выполнения задачи требуемым методом. Передвижение происходит посредством ввода номеров соответствующих пунктов в консоль.

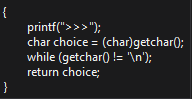
**Функционал модуля с вычислительными алгоритмами:**

Модуль позволяет произвести все необходимые вычисление, запрошенные пользователем посредством меню.

**Код программы**



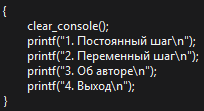
**Функция char read\_symbol()** – считываетпервый символ введенный с клавиатуры



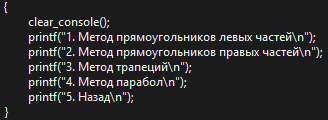
**Функция clear\_console()** – удаляет предыдущий код



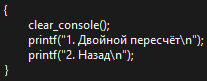
**Функция print\_menu()** – выводит основное меню на экран



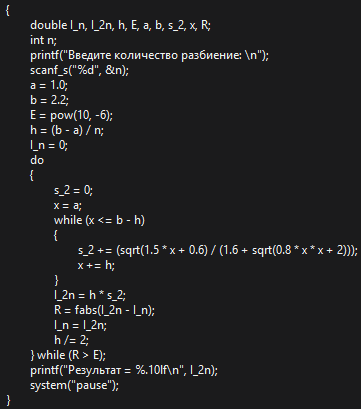
**Функция print\_constant\_menu()** – выводит меню выбора метода с постоянным шагом



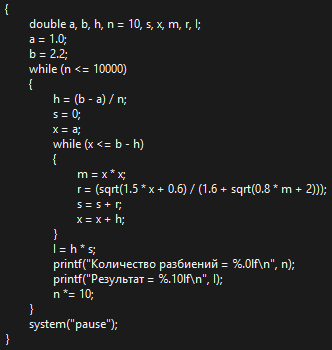
**Функция print\_variable\_menu()** – выводит меню выбора метода с переменным шагом



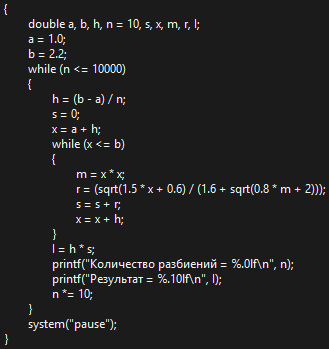
**Функция left\_variable\_metode()** – вычисление «Двойной пересчёт»



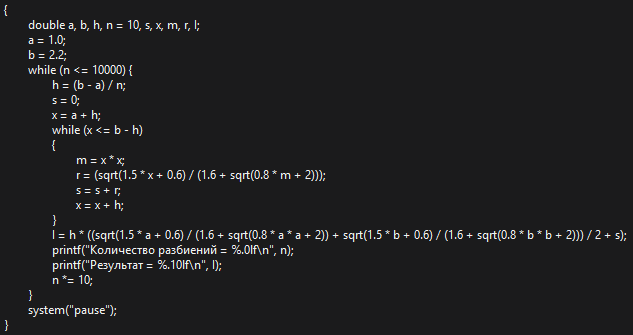
**Функция left\_metode()** – вычисление методом левых частей прямоугольников



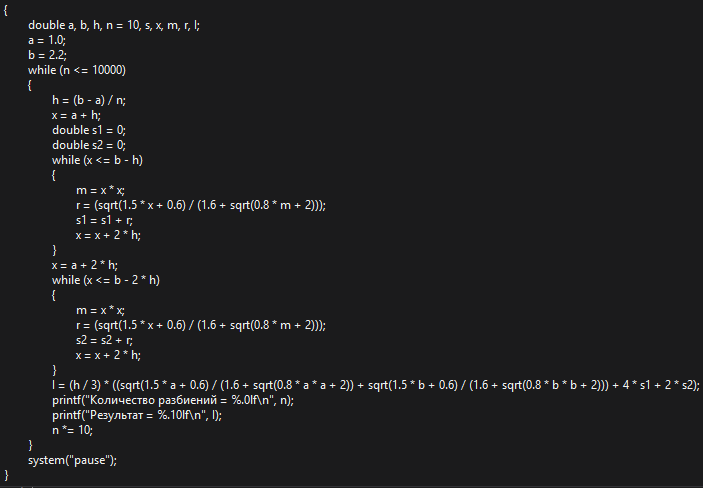
**Функция right\_metode()** – вычисление методом правых частей прямоугольников



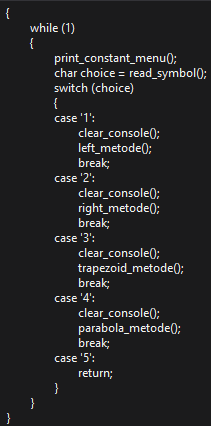
**Функция trapezoid\_metode()** – вычисление методом трапеций



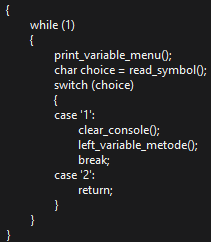
**Функция parabola\_metode()** – вычисление методом Симпсона



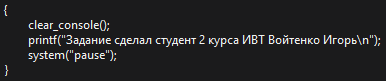
**Функция print\_constant\_step()** – выводит меню с постоянным шагом и просит ввести значение



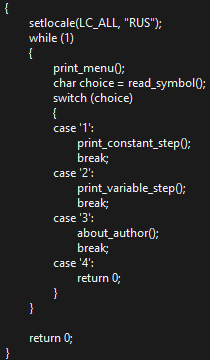
**Функция print\_variable\_step()** – выводит меню с переменным шагом и просит ввести значение



**Функция about\_author()** – выводит информацию об авторе



**Функция int main()** – выводит главное меню и просит ввести значение



**Список идентификаторов**

|  |  |  |
| --- | --- | --- |
| **Название переменной** | **Смысл** | **Тип** |
| **Общие переменные** | | |
| a | Нижний предел интегрирования | double |
| b | Верхний предел интегрирования | double |
| h | Шаг | double |
| x | Аргумент, по которому вычисляет интеграл | double |
| choice | Выбор меню | char |
| n | Число разбиений | int |
| **Переменный шаг** | | |
| I\_n | Интеграл с шагом h (переменный шаг) | double |
| I\_2n | Интеграл с шагом h/2 (переменный шаг) | double |
| E | Точность (переменный шаг) | double |
| s\_2 | Накопленная сумма (переменный шаг) | double |
| R | Модуль разности I\_n и I\_2n (переменный шаг) | double |
| **Постоянный шаг** | | |
| s | Накопленная сумма (постоянный шаг) | double |
| m | Промежуточная переменная (постоянный шаг) | double |
| r | Промежуточная переменная (постоянный шаг) | double |
| I | Интеграл (постоянный шаг) | double |
| s1 | Накопленная сумма для нечетный (метод Симпсона) | double |
| s2 | Накопленная сумма для четных (метод Симпсона) | double |
| **Функции** | | |
| char read\_symbol() | Функция проверяет первый символ, который введен | void |
| clear\_console() | Происходит очистка предыдущего текста в консоли | void |
| print\_menu() | Выводит на экран главное меню | void |
| print\_constant\_menu() | Выводит на экран подменю с постоянным шагом | void |
| print\_variable\_menu() | Выводит на экран подменю с переменным шагом | void |
| left\_variable\_metode() | Вычисление методом левых частей прямоугольников с переменным шагом (Двойной пересчет) | void |
| left\_metode() | Вычисление методом левых частей прямоугольников с постоянным шагом | void |
| right\_metode() | Вычисление методом правых частей прямоугольников с постоянным шагом | void |
| trapezoid\_metode() | Вычисление методом трапеций с постоянным шагом | void |
| parabola\_metode() | Вычисление методом Симпсона с постоянным шагом | void |
| print\_constant\_step() | Выводит подменю постоянного шага и просит ввести номер, выбранного алгоритма | void |
| print\_variable\_step() | Выводит подменю переменного шага и просит ввести номер, выбранного алгоритма | void |
| about\_author() | Выводит подменю с информацией об авторе | void |
| int main() | Выводит главное меню и просит ввести номер выбранного подменю | void |

**Таблица результатов**

**Постоянный шаг:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Название метода** | **n=10** | **n=100** | **n=1000** | **n=10000** |
| Метод левых частей прямоугольников | 0,5086216688 | 0,5642256179 | 0,5697871076 | 0,5704021432 |
| Метод правых частей прямоугольников | 0,5142854903 | 0,5648025437 | 0,5698448781 | 0,5704079211 |
| Метод трапеций | 0,5115106083 | 0,5645145118 | 0,5698159970 | 0,5704050321 |
| Метод парабол | 0,5704049318 | 0,5586267099 | 0,5704050322 | 0,5704050322 |

**Вывод:** чем больше количество разбиений, тем меньше шаг и тем точнее вычисления. Самым точным при 10 разбиениях оказался метод парабол, при остальных - трапеций.

**Переменный шаг:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Название метода** | **n = 10** | **n = 100** | **n = 1000** | **n = 10000** |
| Двойной пересчёт | 0,5704050102 | 0,5704044680 | 0,5704045808 | 0,5704045495 |

**Вывод:** алгоритм с переменным шагом точнее любого с постоянным шагом при любом количестве разбиений из-за нахождения шага конкретно для определенной точности

Второй алгоритм намного точнее при 10 разбиениях, но при 100 и 1000 разбиениях и точностях в 0.1 и 0.01 результаты одинаковы у обоих алгоритмов. При количестве разбиений от 10000 второй алгоритм точнее за счет большего приближение к реальному ответу и быстрее за счет отсутствия повторений вычислений в одних++ и тех же точках.

**Вывод**

В данной лабораторной работе были разработаны различные алгоритмы для нахождения интеграла (методы прямоугольников левых и правых частей, трапеций, Симпсона). Для метода прямоугольников левых частей был разработан алгоритм с переменным шагом.

Среди алгоритмов для постоянного шага наиболее точным является метод парабол (метод Симпсона). Он наиболее универсален. Наименее точен метод прямоугольников, он имеет наибольшую погрешность.

Алгоритм с переменным шагом позволяет рассчитывать интеграл, задавая не конкретный размер шагов, а рассчитывать его исходя из заданной точности вычисления.