**numerical\_methods(численные методы)**

Проект состоит из двух программ:

1. integrals\_function\_resolving;

Позволяет вычислять значение интеграла функции с заданной точностью. Содержит следующую структуру:

1. integrals

Библиотека, описывающая логику вычисления интеграла. Содержит следующие файлы:

1. ***resolving.py***

Содержит описание класса ***resolving***, являющегося контейнером для хранения ответа.

Объект класса содержит атрибуты:

* value (*float*) – вычисленное значение
* steps (*int*) – количество шагов, потребовавшееся для вычисления значения
* method\_name (*string*) – название метода, с помощью которого было вычислено значение

И классовый метод:

* ***from\_dict*** – преобразует json-строку вида {“value”: x, “steps”: n, “method\_name”: name} в объект класса ***resolving****.* В качестве параметра принимает словарь (*dict*). Возвращает объект класса ***resolving***

1. ***resolving\_methods.py***

Содержит описание класса ***integrals\_resolving***, являющегося материнским для остальных классов в модуле.

Объект класса содержит атрибуты:

* base\_function (*string*) – подынтегральная функция в виде строки, написанной с учетом синтаксиса Python 3.5
* error (*float*) – точность вычислений. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.001
* range (*list*) – границы интегрирования
* method\_name (*string*) – название метода
* debug (*bool*) – состояние режима отладки. В активном состоянии позволяет выводить все промежуточные вычисления

А также методы:

* ***f*** – принимает в качестве параметра число и подставляет его в base\_function. Возвращает число
* ***resolve*** – не принимает параметры. Производит вычисления по логике того или иного метода. Возвращает решение в виде объекта класса ***resolving***

В модуле содержатся дочерние от ***integrals\_resolving*** и переопределяющие только метод ***resolve*** классы:

* ***left\_rect*** – описывает логику вычислений методом левых прямоугольников
* ***right\_rect*** – описывает логику вычислений методом правых прямоугольников
* ***middle\_rect*** – описывает логику вычислений методом средних прямоугольников
* ***trapeze*** – описывает логику вычислений методом трапеций
* ***parabole*** – описывает логику вычислений методом парабол

1. solution

Библиотека, описывающая логику анализа эффективности методов и интерфейса программы. Содержит следующие файлы:

1. ***analysis.py***

Содержит описание класса ***analyzer***, реализующего логику анализа методов.

Объект класса содержит атрибуты:

* methods (*list*) – список методов, для которых необходимо провести анализ. При инициализации может передаваться как в виде списка, так и в виде последовательности элементов
* min\_error (*float*) – минимальная точность. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.001
* max\_error (*float*) – максимальная точность. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.01
* error\_step (*float*) – шаг изменения точности от минимальной до максимальной. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.001
* method\_names (*list*) – список названий методов, переданных в анализатор. Не передается при инициализации, т.к. создается внутри метода объекта
* method\_steps (*list*) – список шагов вычисления интеграла для каждой точности каждым из методов, переданных в анализатор. Не передается при инициализации, т.к. создается внутри метода объекта

А также методы:

* ***annalyze*** – собирает данные об эффективности каждого метода, вычисляя значение интеграла с точностью от минимальной до максимальной для каждого из них.
* ***graph*** – на основе собранных методом ***annalyze*** строит график эффективности методов
* ***run*** – вызывает методы ***annalyze*** и ***graph***, снимая тем самым нагрузку с пользователя, котору остается лишь вызвать метод ***run***

1. ***solution.py***

Содержит описание класса ***solution***, реализующего примитивный консольный интерфейс пользователя.

Объект класса содержит атрибуты:

* debug (*bool*) – состояние режима отладки. В активном состоянии позволяет выводить все промежуточные вычисления интеграла
* analyze (*bool*) – состояние режима анализатора. В активном состоянии проводит анализ эффективности и строит соответствующий график

А также методы:

* ***\_\_check\_choise*** – принимает в качестве параметра выбор в виде строки (*string*), обрабатывает ее и возвращает True или False, в зависимости от сделанного выбора (yes/y или no/n соответственно)
* ***\_\_make\_choise*** – принимает в качестве параметра заголовок (вопрос) (*string*). Выводит его на консоль и запрашивает ответ пользователя. Возвращает ответ пользователя в виде строки (*string*)
* ***run*** – реализует примитивный консольный интерфейс программы

1. main.py

Основной скрипт программы, который необходимо запустить пользователю для начала работы.

1. nonlineal\_equations\_resolving;

Позволяет вычислять значение корня функции с заданной точностью. Содержит следующую структуру:

1. equations

Библиотека, описывающая логику нахождения корней. Содержит следующие файлы:

1. ***resolving.py***

Содержит описание класса ***resolving***, являющегося контейнером для хранения ответа.

Объект класса содержит атрибуты:

* value (*float*) – вычисленное значение
* steps (*int*) – количество шагов, потребовавшееся для вычисления значения
* method\_name (*string*) – название метода, с помощью которого было вычислено значение

И классовый метод:

* ***from\_dict*** – преобразует json-строку вида {“value”: x, “steps”: n, “method\_name”: name} в объект класса ***resolving****.* В качестве параметра принимает словарь (*dict*). Возвращает объект класса ***resolving***

1. ***resolving\_methods.py***

Содержит описание класса ***equation\_resolving***, являющегося материнским для остальных классов в модуле.

Объект класса содержит атрибуты:

* base\_function (*string*) – подынтегральная функция в виде строки, написанной с учетом синтаксиса Python 3.5
* derivative (*string*) – первая производная подынтегральной функции в виде строки, написанной с учетом синтаксиса Python 3.5
* sqr\_derivative (*string*) – вторая производная подынтегральной функции в виде строки, написанной с учетом синтаксиса Python 3.5
* error (*float*) – точность вычислений. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.00001
* range (*list*) – границы интегрирования
* method\_name (*string*) – название метода
* debug (*bool*) – состояние режима отладки. В активном состоянии позволяет выводить все промежуточные вычисления

А также методы:

* ***f*** – принимает в качестве параметра число и подставляет его в base\_function. Возвращает число
* ***f1*** – принимает в качестве параметра число и подставляет его в derivative. Возвращает число.
* ***f2*** – принимает в качестве параметра число и подставляет его в sqr\_derivative. Возвращает число.
* ***resolve*** – не принимает параметры. Производит вычисления по логике того или иного метода. Возвращает решение в виде объекта класса ***resolving***
* ***check\_value*** – проверяет корректность введенных данных, наличие корня на отрезке. В случае обнаружения ошибочных данных генерируется исключение

В модуле содержатся дочерние от ***equation\_resolving*** и переопределяющие только метод ***resolve*** классы:

* ***bisection*** – описывает логику поиска корня методом половинного деления
* ***chord*** – описывает логику поиска корня методом хорд
* ***shearing*** – описывает логику поиска корня методом касательных
* ***combo*** – описывает логику поиска корня методом парабол

1. solution

Библиотека, описывающая логику анализа эффективности методов и интерфейса программы. Содержит следующие файлы:

1. ***analysis.py***

Содержит описание класса ***analyzer***, реализующего логику анализа методов.

Объект класса содержит атрибуты:

* methods (*list*) – список методов, для которых необходимо провести анализ. При инициализации может передаваться как в виде списка, так и в виде последовательности элементов
* min\_error (*float*) – минимальная точность. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.00001
* max\_error (*float*) – максимальная точность. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.0001
* error\_step (*float*) – шаг изменения точности от минимальной до максимальной. Необязательно передавать при инициализации, т.к. имеет значение по умолчанию 0.00001
* method\_names (*list*) – список названий методов, переданных в анализатор. Не передается при инициализации, т.к. создается внутри метода объекта
* method\_steps (*list*) – список шагов вычисления интеграла для каждой точности каждым из методов, переданных в анализатор. Не передается при инициализации, т.к. создается внутри метода объекта.

А также методы:

* ***annalyze*** – собирает данные об эффективности каждого метода, вычисляя значение интеграла с точностью от минимальной до максимальной для каждого из них.
* ***graph*** – на основе собранных методом ***annalyze*** строит график эффективности методов
* ***run*** – вызывает методы ***annalyze*** и ***graph***, снимая тем самым нагрузку с пользователя, котору остается лишь вызвать метод ***run***

1. ***solution.py***

Содержит описание класса ***solution***, реализующего примитивный консольный интерфейс пользователя.

Объект класса содержит атрибуты:

* debug (*bool*) – состояние режима отладки. В активном состоянии позволяет выводить все промежуточные вычисления интеграла
* analyze (*bool*) – состояние режима анализатора. В активном состоянии проводит анализ эффективности и строит соответствующий график

А также методы:

* ***\_\_check\_choise*** – принимает в качестве параметра выбор в виде строки (*string*), обрабатывает ее и возвращает True или False, в зависимости от сделанного выбора (yes/y или no/n соответственно)
* ***\_\_make\_choise*** – принимает в качестве параметра заголовок (вопрос) (*string*). Выводит его на консоль и запрашивает ответ пользователя. Возвращает ответ пользователя в виде строки (*string*)
* ***run*** – реализует примитивный консольный интерфейс программы

1. main.py

Основной скрипт программы, который необходимо запустить пользователю для начала работы.

**Легенда**

|  |  |
| --- | --- |
| **Математические функции и операторы** | **Функции и операторы в синтаксисе Python** |
| **+** | **+** |
| **-** | **-** |
| **\*** | **\*** |
| **/** | **/** |
| ^ | \*\* |
| ^ | math.pow(x) |
| √x | math.sqrt(x) |
| |x| | abs(x) |
| sin(x) | math.sin(x) |
| cos(x) | math.cos(x) |
| tg(x) | math.tan(x) |
| ln(x) | math.log(x) |