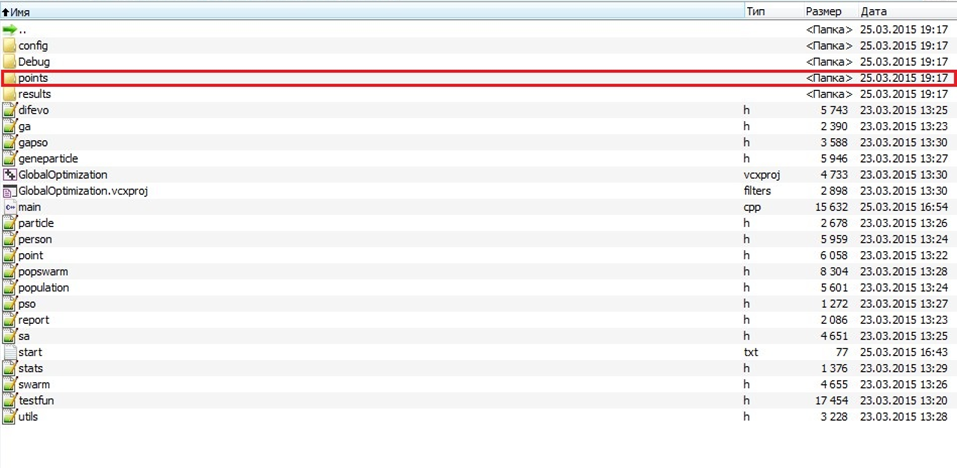
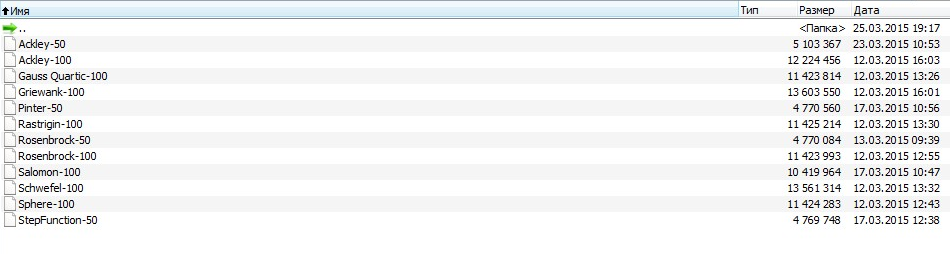
**Методическое пособие по работе с фреймворком и бенчмарком глобальной оптимизации**

# 1. ТЕСТИРОВАНИЕ МЕТОДОВ

Для тестирования методов необходимы наборы стартовых точек и файлы параметров. Первое находится в папке points:





Каждый файл предназначен для определенной функции и размерности. Это отображено в названии.

Параметры состоят из общего файла - start.txt - и индивидуального, содержащего особый набор для каждого метода. В start.txt прописаны:

- **mcode** порядковый номер метода;

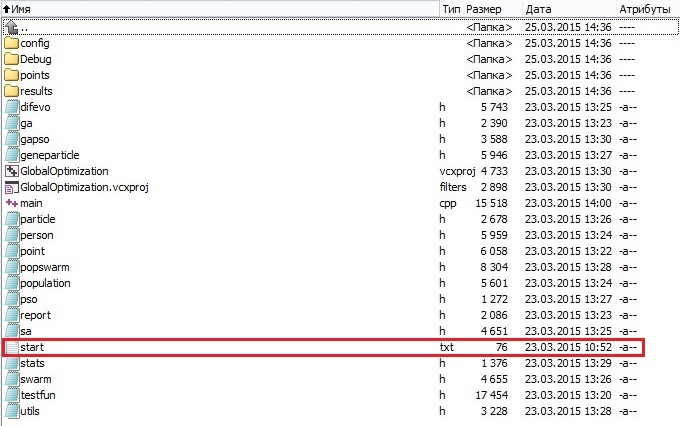
- **fcode** порядковый номер функции бенчмарка;

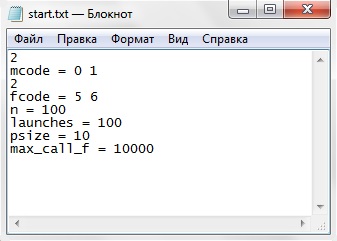
- **n** размерность задачи;

- **launches** число запусков для усреднения;

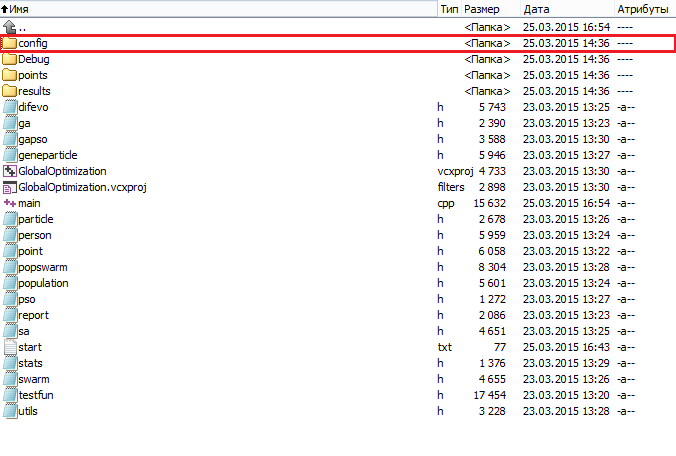
- **psize** количество точек;

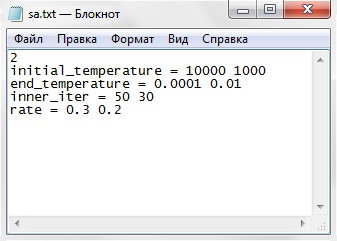
- **max\_call\_f** максимальное число вызовов ЦФ.



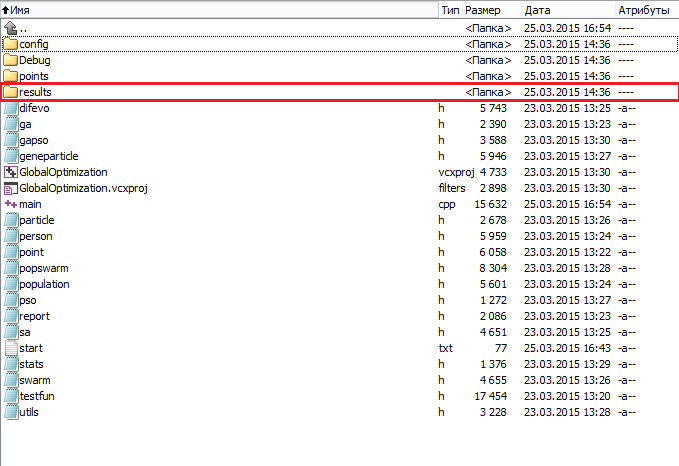


В папке config расположены файлы параметров, которые не вошли в общий список. Название соответствует методу. Например, метод имитации отжига – файл sa.txt, параметры - initial\_temperature стартовая температура, end\_temperature итоговая температура, inner\_iter коэффициент понижения температуры, rate шаг.



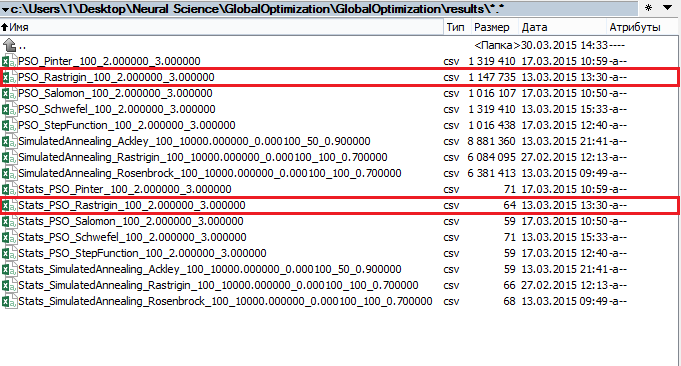


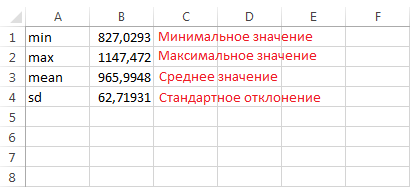
Результаты запуска сохраняются в 2х .csv файлах в папке results.



Имя первого файла состоит из названия метода, функции, размерности и параметров через «\_». Первая строка содержит количество вызовов тестовой функции на каждой итерации. Каждая следующая строка – каждый отдельный запуск, в ячейке которого записано значение функции на соответствующей итерации. Последняя строка – усредненные по столбцам значения.

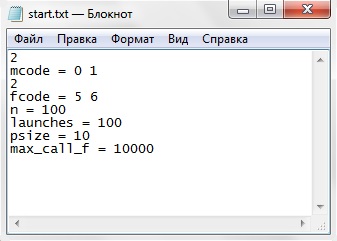
Имя второго файла начинается со «Stats\_». В этом файле отображаются основные статистики по запускам: минимальное, максимальное, среднее значения и стандартное отклонение достигнутого минимума.



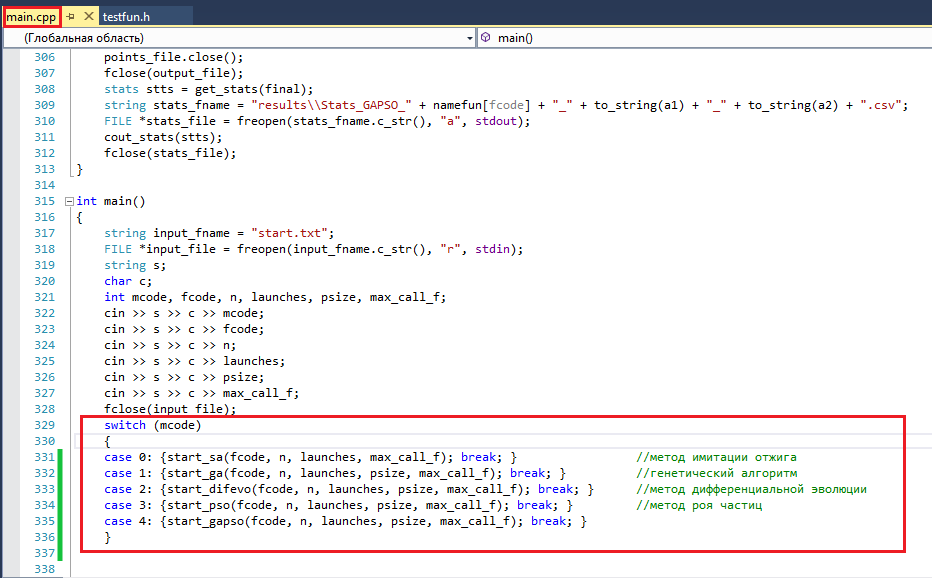


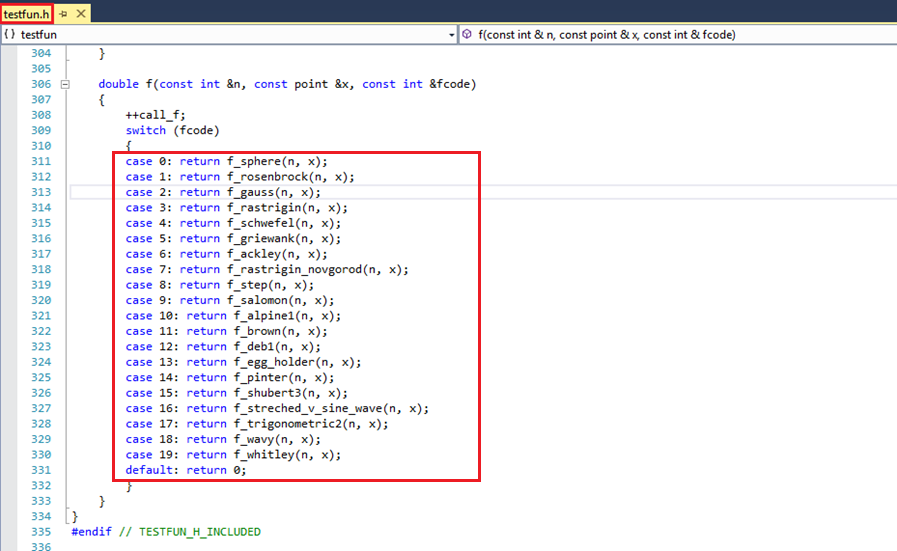
Для того чтобы протестировать конкретный метод на выбранной функции, необходимо сделать следующее:

1. прописать в start.txt общие параметры. Перед mcode и fcode нужно указать количество методов и функций соответственно, а сами значения параметров указать после через пробел. Например, для запуска метода имитации отжига и генетического алгоритма на 2 функциях griewank и ackley файл будет таким:

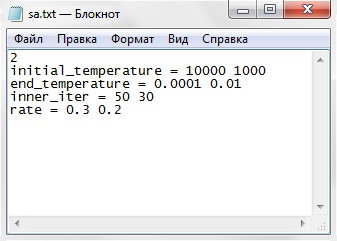


Для того чтобы узнать порядковый номер метода можно посмотреть главную функцию main() в main.cpp, а номер функции - функцию вызова f в testfun.h.





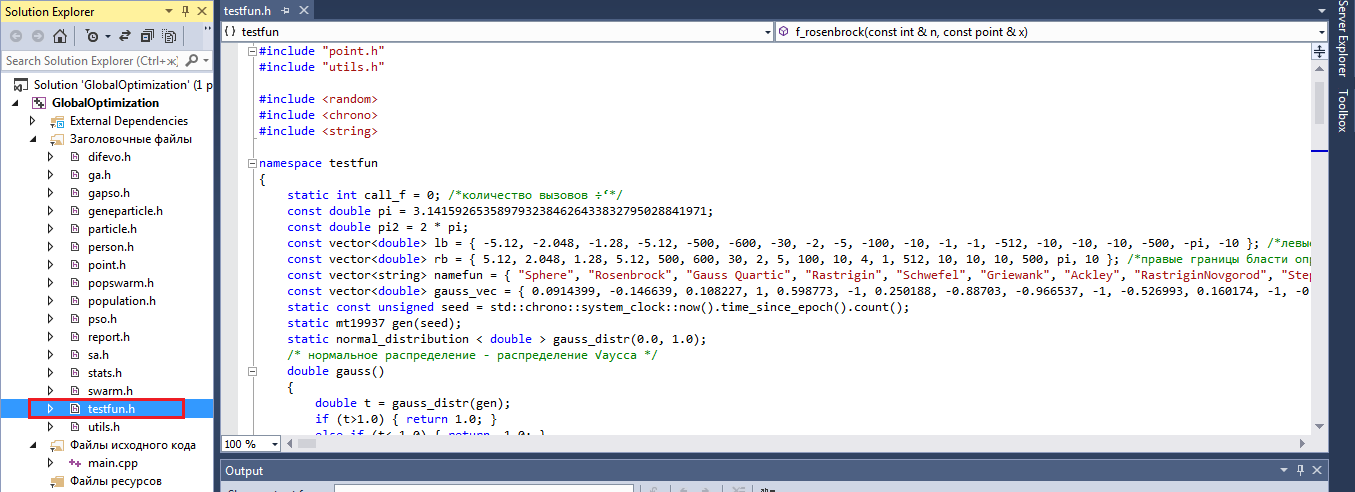
1. Изменить данные в соответствующих файлах параметров методов. Перед набором указать количество, сами значения после через пробел.



1. Запустить программу и ждать окончания работы.   
   В случае отсутствия необходимого набора стартовых точек для нужной тестовой функции и/или размерности задачи он будет сгенерирован автоматически и сохранен.
2. В папку result добавятся 2 файла с результатами.

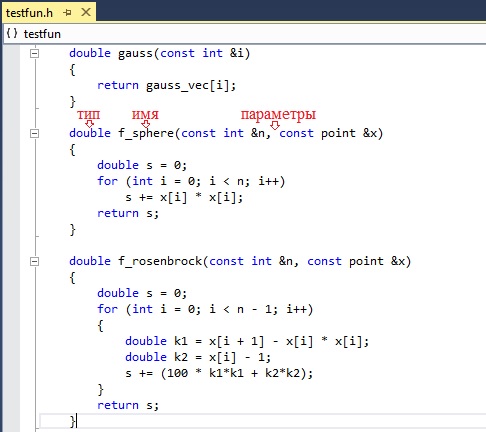
## 2. ДОБАВЛЕНИЕ ТЕСТОВЫХ ФУНКЦИЙ

Все данные об используемых тестовых функциях, такие как сам программный код, границы области определения и название, содержатся в tesfun.h.

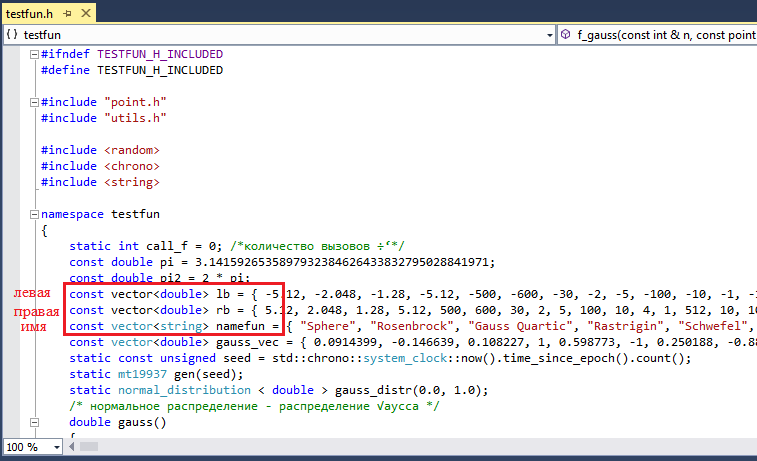


Для того чтобы добавить тестовую функцию, необходимо сделать следующее:

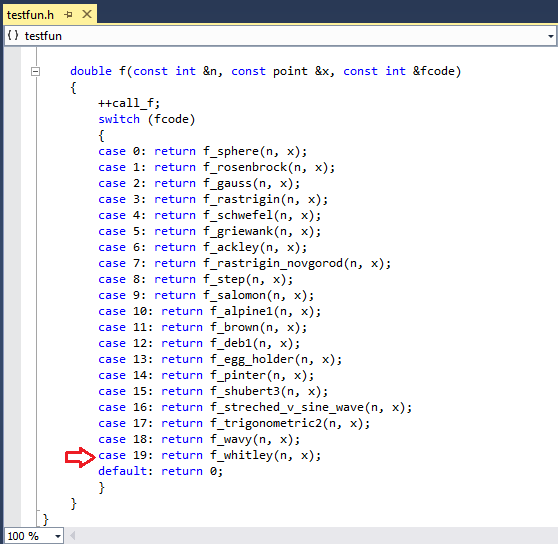
1. Добавить код функции в пространство имён. Имя функции состоит из «f\_»+название функции.



1. В векторы границ lb и rb добавить значения левой и правой границ области определения. В вектор namefun – название функции, заключенное в «».



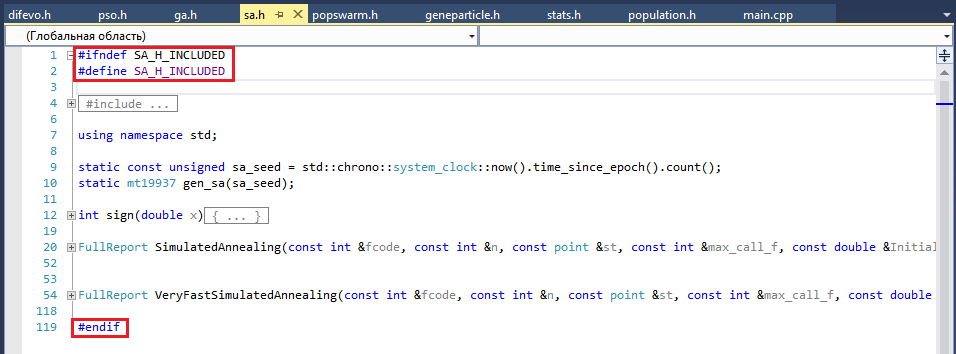
1. Прописать в главной функции вызова f новую функцию со соответствующим порядковым номером (последний +1). Порядковые номера функции в вызове и в векторах границ, названия должны совпадать!



## 3. Добавление метода

Для того чтобы добавить свой метод необходимо:

1. создать нужное количество заголовочных файлов (.h) для выбранного метода. Имена должны кратко отображать суть. Например, имитация отжига SimulatedAnnealing -sa.h. В нем будет реализован непосредственно сам метод.
2. Дальше необходимо прописать программный код. В самом начале - стражи включения #ifndef и #define, после которых название заголовочного файла + «\_H\_INCLUDED», в конце – закрытие.

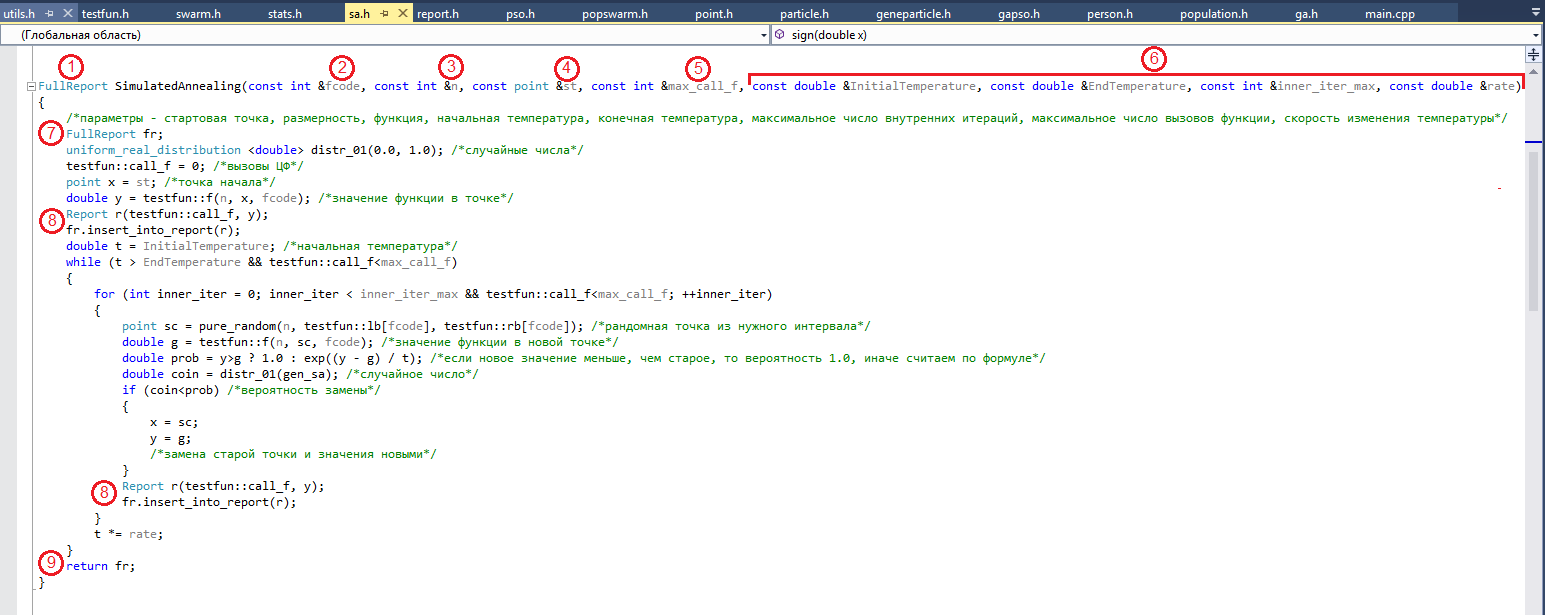


Дальше сам метод. Он должен быть реализован в виде функции, которая возвращает объект класса FullReport (1). Эта функция имеет следующие обязательные параметры:

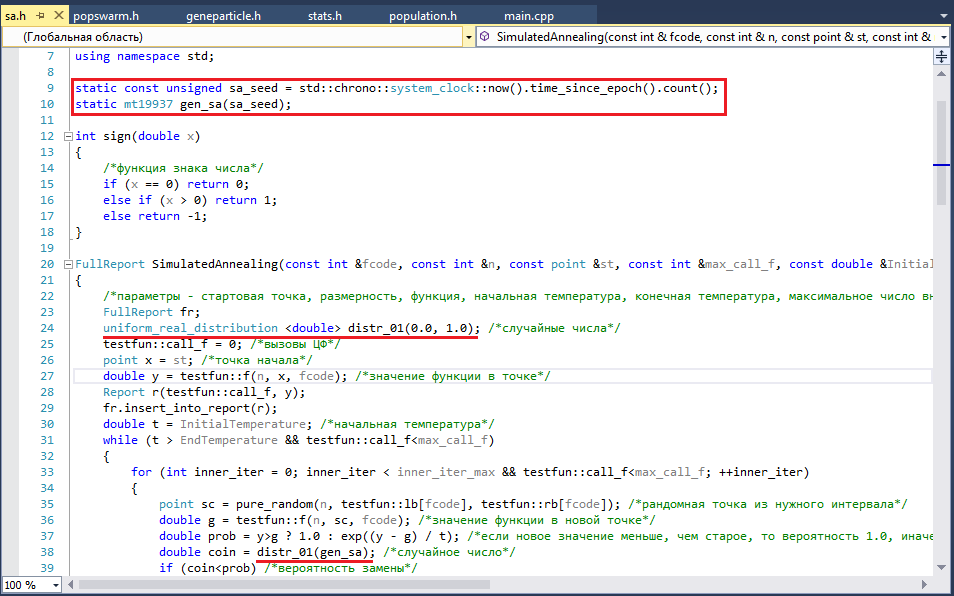
* порядковый номер целевой функции (2);
* размерность задачи (3);
* стартовая точка типа point (для одноточечных методов) или вектор стартовых точек (для многоточечных методов) (4);
* максимальное число вызовов целевой функции (5).

Остальные параметры определяются разработчиком метода (6). Все параметры могут быть расположены в произвольном порядке. Рекомендуется передавать параметры по ссылке (&) и указывать перед типом const.

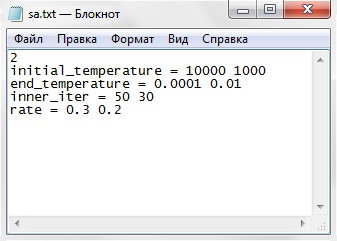
Тело функции начинается с создания объекта класса FullReport (7)– он содержит информацию для отчетности по всему одинарному запуску, такую как количество вызовов тестовой функции и её значение на каждом шаге. Также создается объект класса Report–туда записываются данные по одной итерации. Добавление его в полный отчет производится с помощью метода insert\_into\_report(8). Функция разрабатываемого метода возвращает объект класса FullReport (9), созданный в начале.



Рекомендуется использовать генератор псевдослучайных чисел из библиотеки <random.h> Вихрь Мерсенна - обеспечивает быструю генерацию высококачественных псевдослучайных чисел.

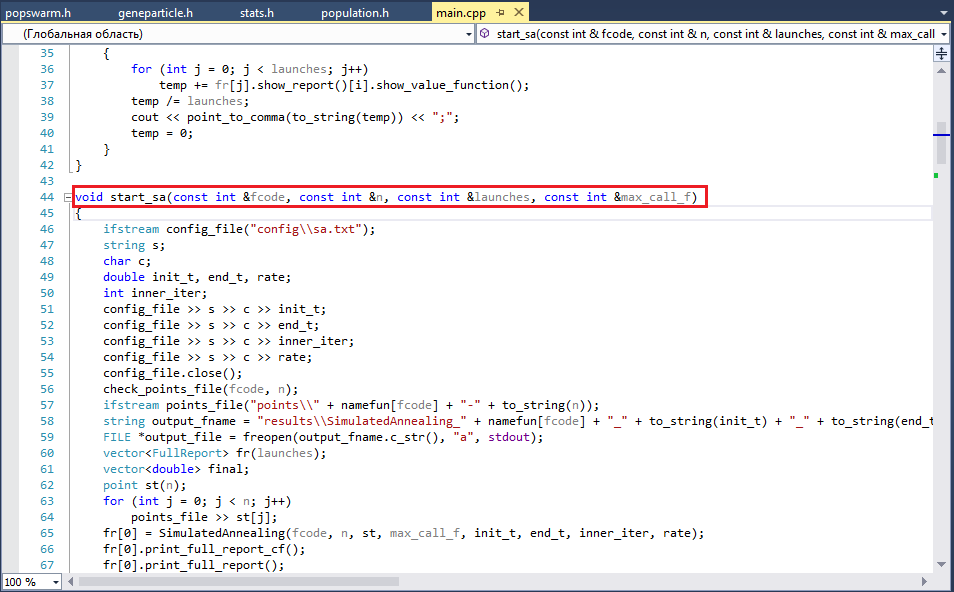


1. В папку config добавьте конфигурационный файл формата txt реализованного метода со соответствующим названием. В него пропишите все параметры алгоритма в формате: название « = » значение.



1. В main.cpp добавить функцию метода. Имя состоит из «start\_» + название метода. Эта функция должна иметь следующие параметры:

* порядковый номер целевой функции;
* размерность задачи;
* число запусков для усреднения;
* максимальное число вызовов целевой функции.



1. Добавить вызов функции в main(),подключить заголовочный файл метода.

