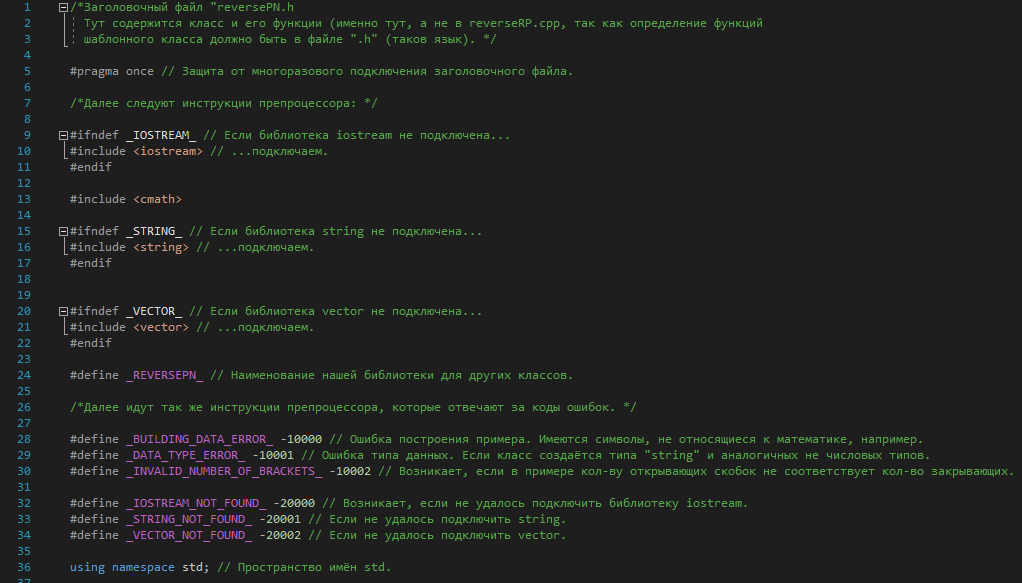
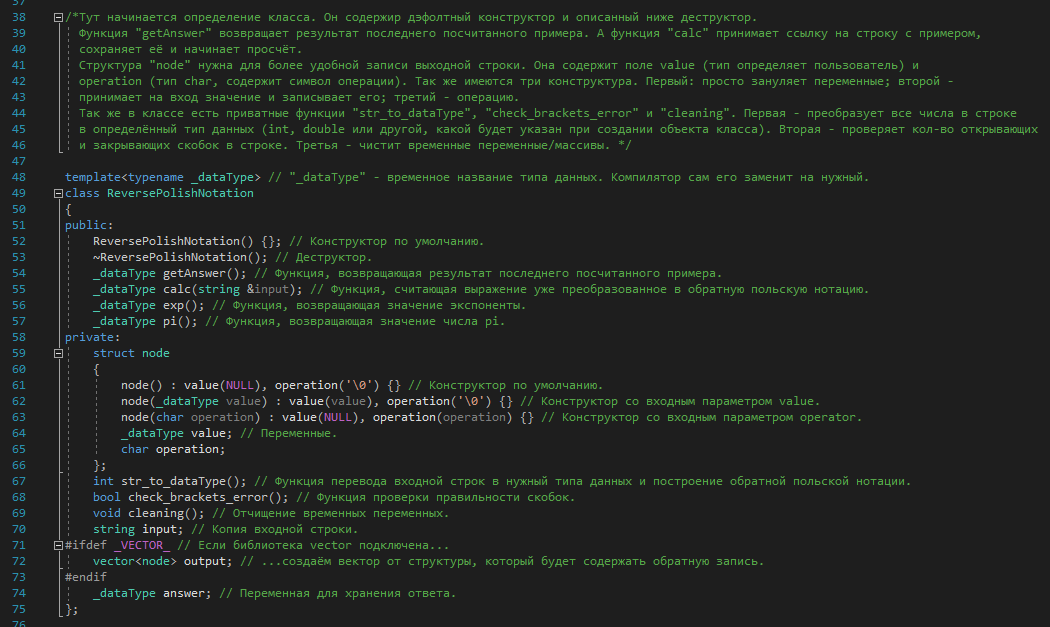
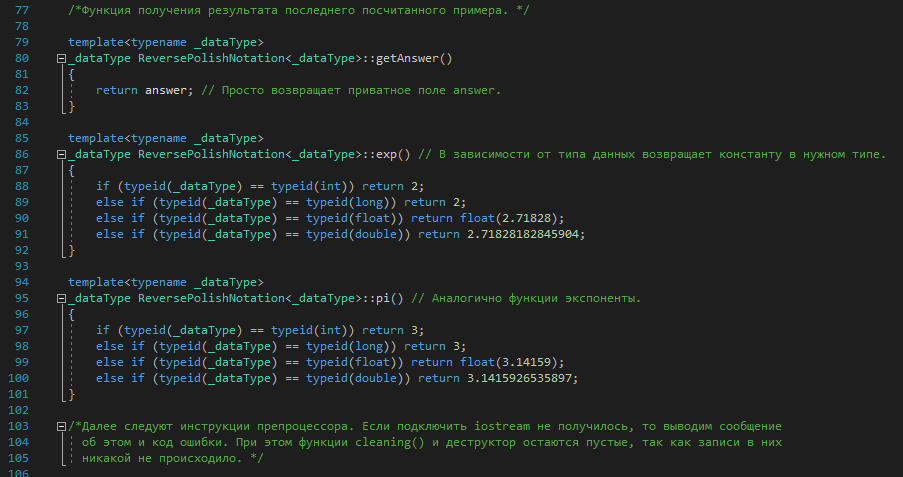
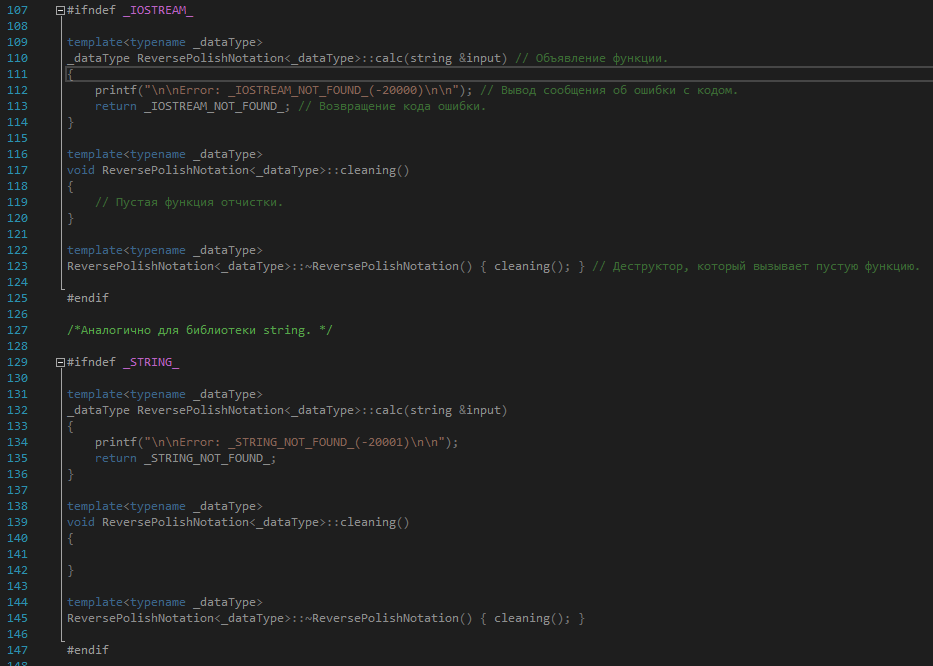
**6. Дополнительные материалы.**

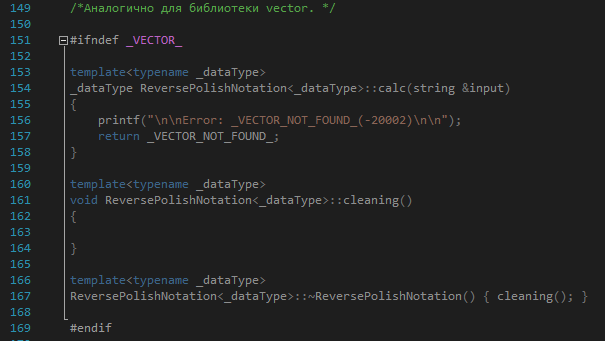
Прикрепляю скриншоты кода для более удобной визуальной проверки.

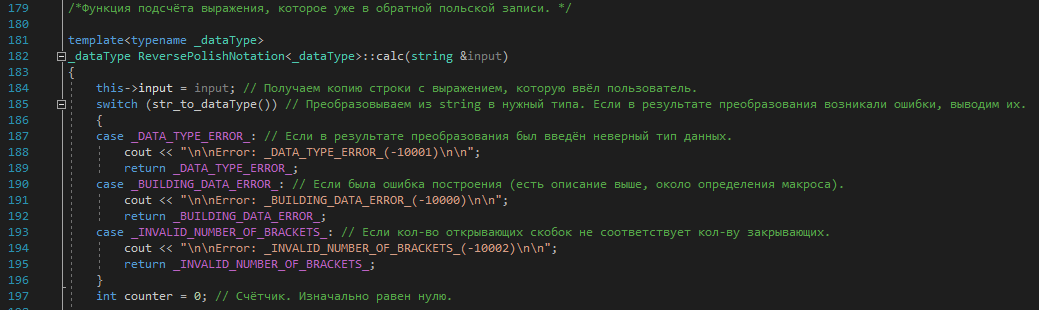


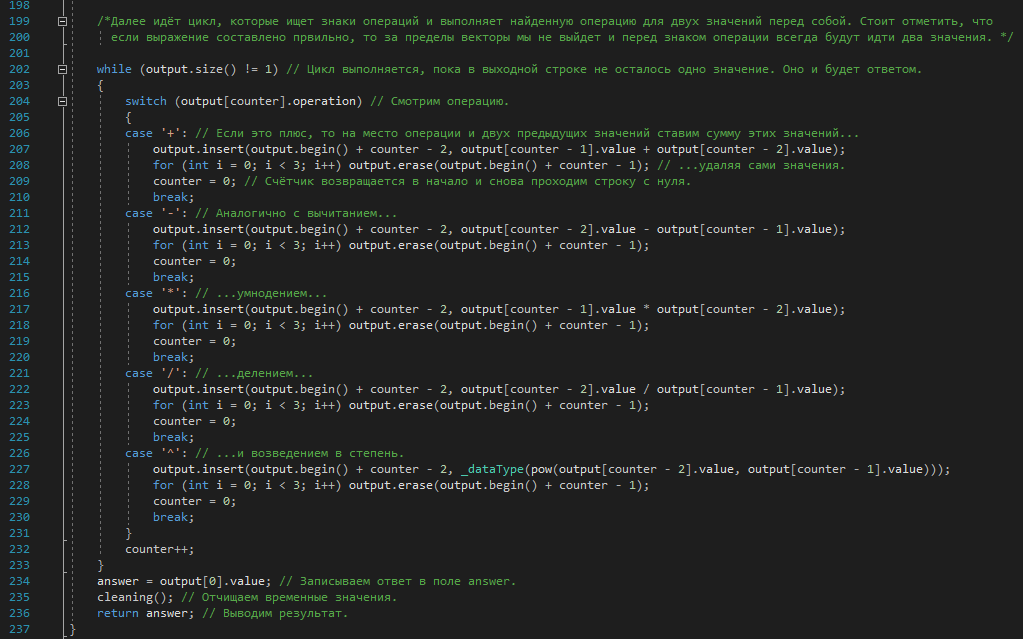


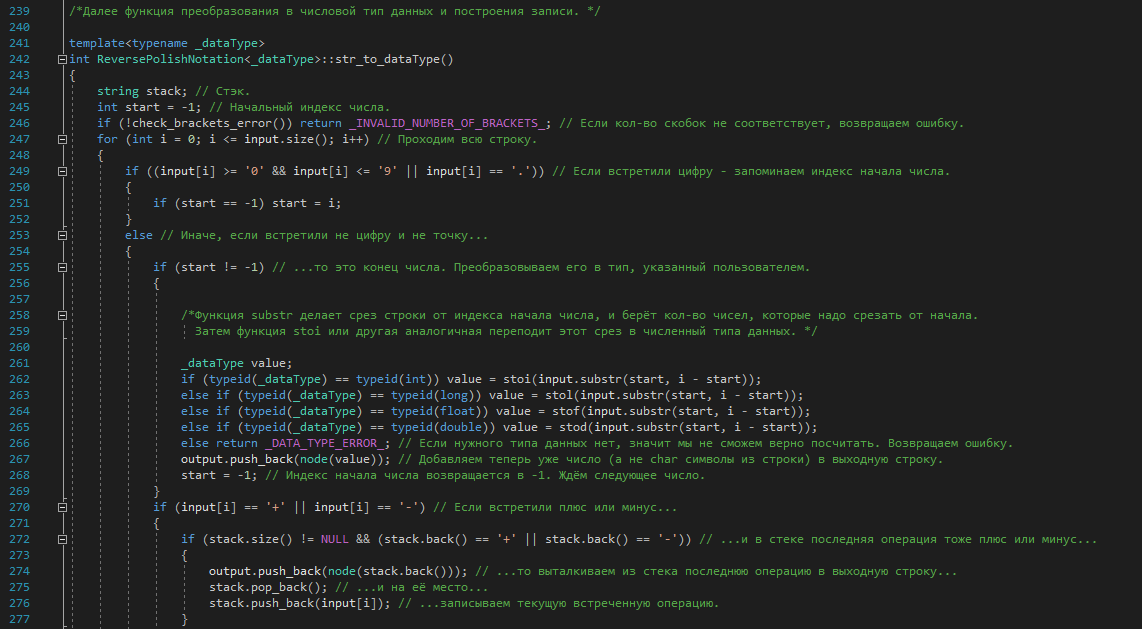


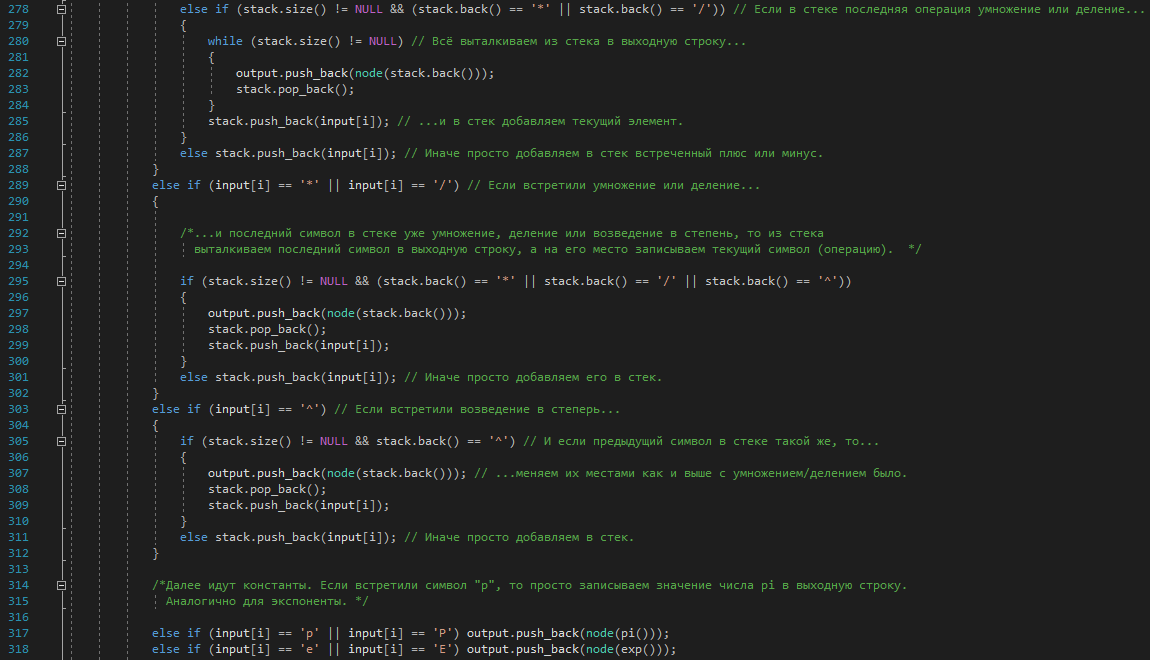


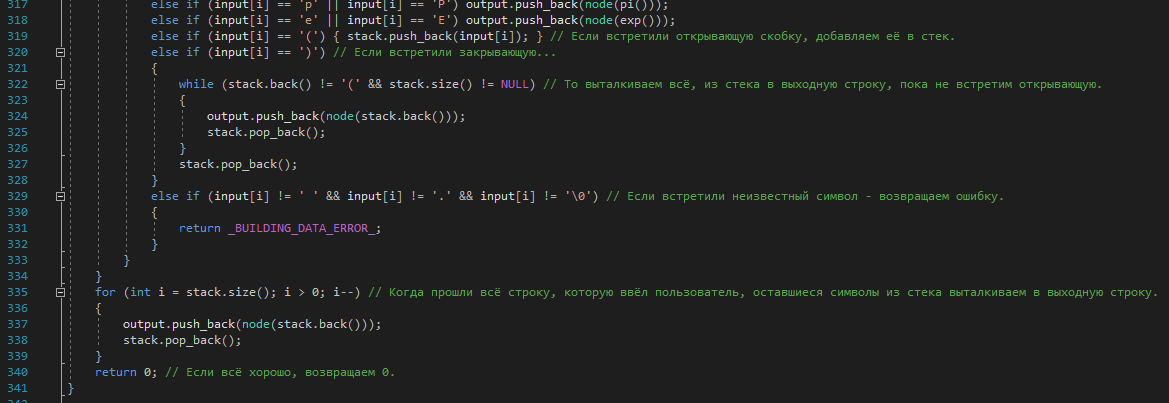


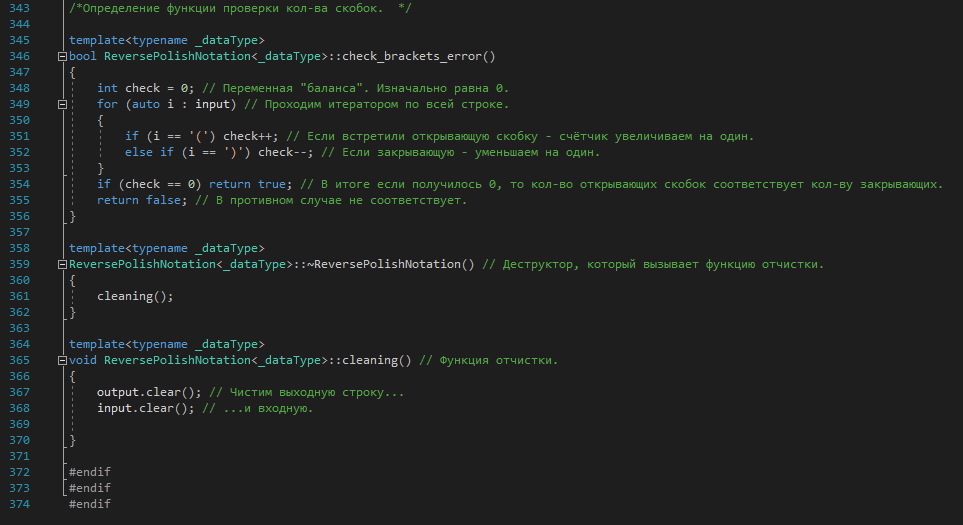








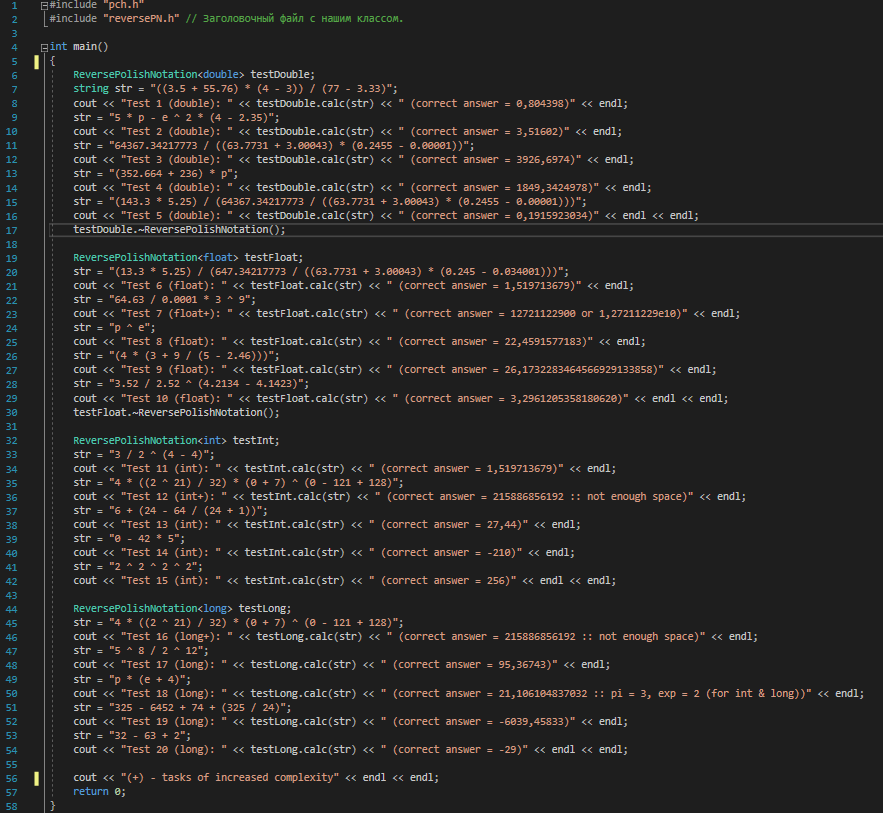


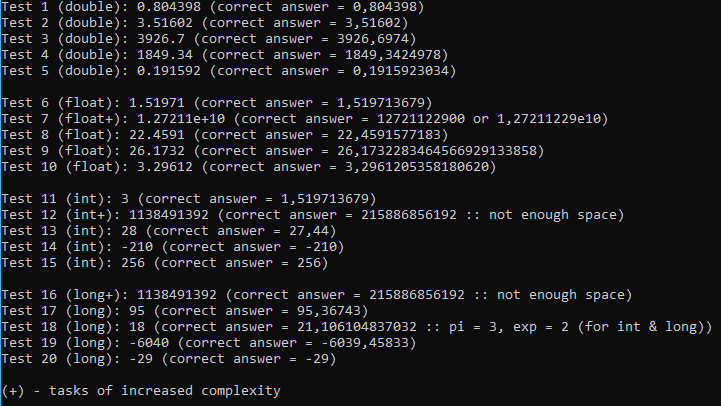


Тесты:



Листинг 12: тесты





**7. Пояснения.**

Для типов int и long константы π и exp равны 3 и 2 соответственно. Для типа float эти константы округлены до 5-ти знаков после запятой. Для типа double – около 10 знаков после запятой.

Для типов int и long любое число, возведённое в степень меньше нуля, считается равным нулю. Для типов float и double возведение в степень считается как в математике (23 = 8; 2-3 = 0.125).

Тесты 12 и 16 дают верный результат при увеличении кол-ва байтов, выделяемых под типы int и long. Преобразование в научную запись невозможно из-за огромных потерь, так как данные типы не поддерживают дробную часть. Однако если данные тесты просчитывать в типе float или double, то погрешность будет около 150-ти тысяч из-за научной записи.

Результаты выражения в типе float и double могут быть преобразованы в научную запись, что позволяет считать примеры с результатом верхней границы до 1 \* 10100 и нижней границей до 1 \* 10-100.

Выбрать тип, в котором будет считаться пример, можно при объявлении объекта класса в main файле. Объявление аналогично контейнерным классам vector, list, map и аналогичным.

Классу можно отправлять несколько примеров за раз и можно несколько раз объявлять от одного или разных типов данных (как в коде тестов).

В данной программе не реализован унарный минус. Выражения типа “5 \* (-4)” следует записывать в виде “5 \* (0 - 4)”.

В отличии от операций умножения и сложения, которые имеют равный приоритет и выполняются слева направо если повторяются, операция возведения в степень выполняется справа налево. Рассмотрим 15-й тест. В нём выражение “2 ^ 2 ^ 2 ^ 2” выполняется справа налево, так как данное выражение можно преобразовать в , что в свою очередь преобразовывается в и равно 256. Если выполнять 15-й тест слева направо, то должна быть возможность преобразовать данное выражение в формулу, которая получает 65536. Однако, вывести такую формулу я не смог и поэтому сделал счёт справа налево.

Из чисел типа float и double можно извлекать корень, помня следующее: . Чтобы посчитать выражение его нужно записать в виде . Для данной программы это будет выглядеть так: 2187 ^ (1 / 7). *Эта инструкция не применима к типам int и long.*