**ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА АНАЛИЗА КОДА ПРОГРАММ**

# **Исследование программы с использованием статического анализатора Cppcheck**

Используя статический анализатор, удалось выяснить, что в программе есть три критические ошибки. Результаты работы анализатора представлены на рисунке 1.

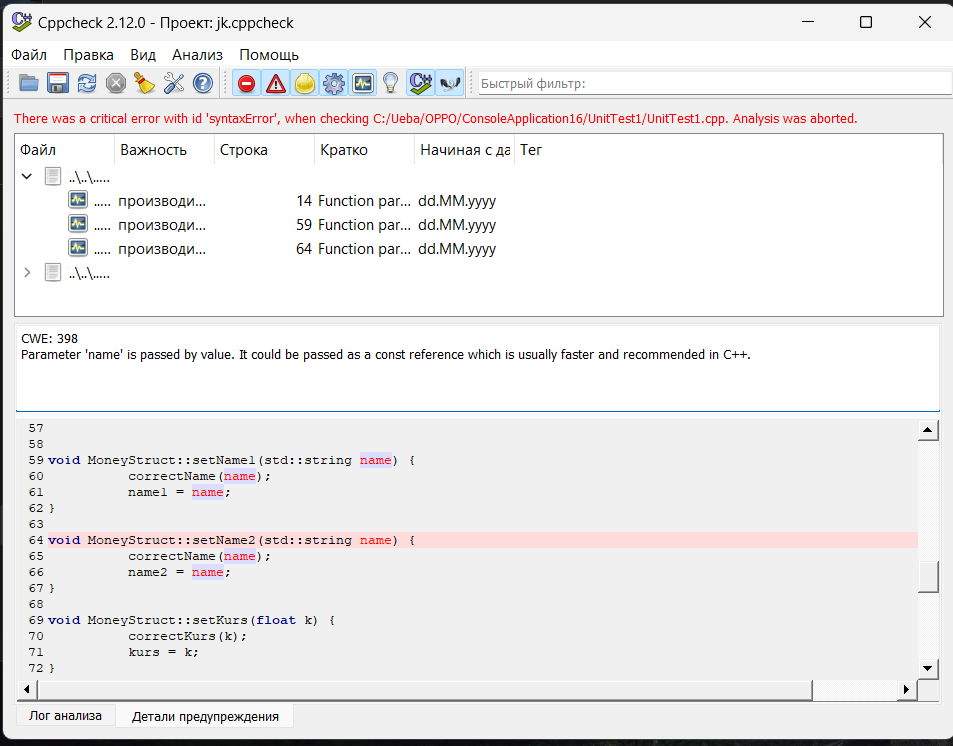


Рисунок 1 – результат работы анализатора Cppcheck.

Как видно из рисунка, ошибки находятся в файле “moneyMethods.cpp”, текст ошибки “*Parameter 'name' is passed by value. It could be passed as a const reference which is usually faster and recommended in C++.*”. Так же есть синтаксические ошибки.

Для исправления этих ошибок необходимо было передавать в функцию ссылку const. Аналогично для двух других ошибок.

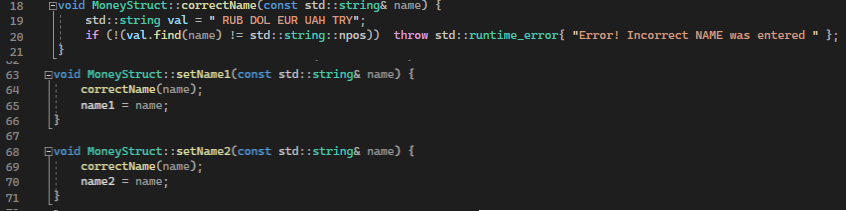


Рисунок 2 – Исправление ошибок, найденных в Cppcheck.

# **Исследование программы с использованием статического анализатора PVS-Studio**

Используя анализатор PVS-Studio удалось выявить несколько ошибок, разделённых различными категориями. Ошибки вида “High”, найденные в программе, представлены на рисунке 3.

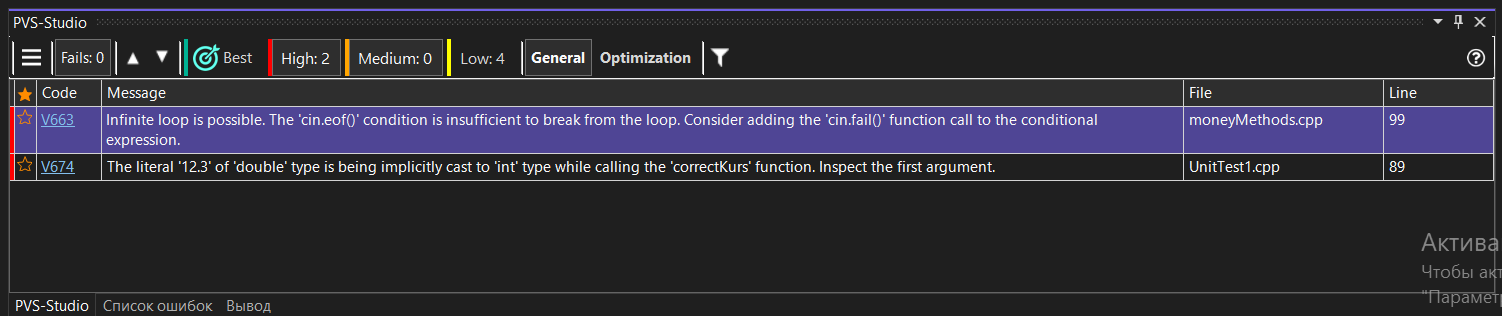


Рисунок 3 – Ошибки вида “High”, найденные в программе.

Как видно из рисунка 3, было найдено две ошибки в файлах: moneyMethods.cpp и UnitTest.cpp. Для исправления первой ошибки “ *- V663 Infinite loop is possible. The 'cin.eof()' condition is insufficient to break from the loop. Consider adding the 'cin.fail()' function call to the conditional expression. moneyMethods.cpp 99*”было принято решение добавить в условие цикла while дополнительный параметр. Исправление первой ошибки представлено на рисунке 4.

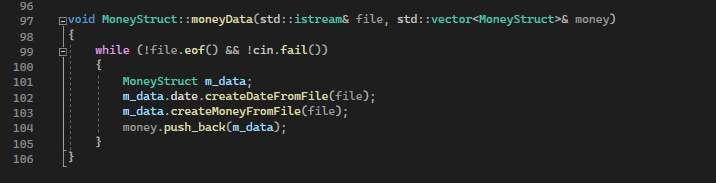


Рисунок 4 – Исправление ошибки V663.

Как видно из рисунка 4, в условие цикла было добавлено выражение “!std::cin.fail()”, что позволило корректно завершать цикл в случае, если чтение данных из файла будет неудачным.

Для исправления второй ошибки “*V674 The literal '12.2' of 'double' type is being implicitly cast to 'int' type while calling the 'correctKurs' function. Inspect the first argument. UnitTest1.cpp 89*”было принято решение изменить первый параметр в файле moneyMethods.cpp в функции correctKurs, (строка 13), вызываемая в TEST\_METHOD(validCorrectKurs) в UnitTest1.cpp, с типа int на тип float. Исправление второй ошибки представлено на рисунке 5.

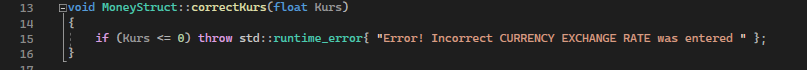
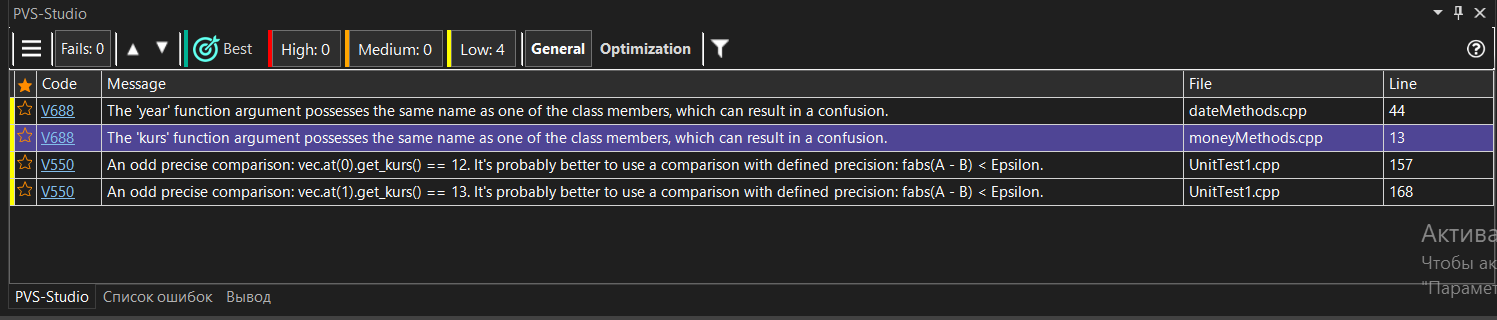


Рисунок 5 – Исправление ошибки V674

Помимо ошибок вида “High”, были найдены также ошибки вида “Low”, которые являются незначительными. На рисунке 6 представлены найденные ошибки.

Рисунок 6 – Ошибки вида “Low”, найденные в программе.

Как видно из рисунка, в трёх файлах (dateMethods.cpp, moneyMethods.cpp, UnitTest1.cpp), были найдены ошибки всего двух видов – V688 и V550. Для решения ошибки V688, которая имеет вид: “*V688 The 'year' function argument possesses the same name as one of the class members, which can result in a confusion. dateMethods.cpp 44*”, для решения было принято решение о переименовании параметров функции isLeapYear(), (с ‘year’ на ‘Year’). Исправление ошибки представлено на рисунке 7.

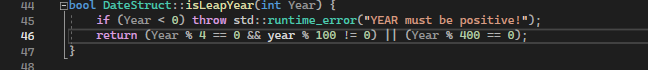


Рисунок 7 – Исправление ошибки V688.

Аналогичное решение, представленное на рисунке 8, было представлено в файле moneyMethods.cpp, (с ‘kurs’ на ‘Kurs’).

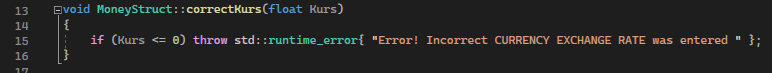


Рисунок 8 – Исправление ошибки V688.

Для исправления ошибки V550, которая имеет вид – “ *V550. An odd precise comparisom: vec.at(0).get\_kurs()==12. It’s probably better to use a comparison with defined precision: fabs(A-B) < Epsilion UnitTest1.cpp in line 157, 168”,* прочитав информацию на оффициальном сайте, где говорилось: “Странно ожидать что значение будет равно именно 12, было принято решение об изменении условия проверки в функции TEST\_METHOD(validMoneyData), (строка 157, 169), (с ‘==’ на ‘>=’). Исправление ошибок V550 представлено на рисунке 9.

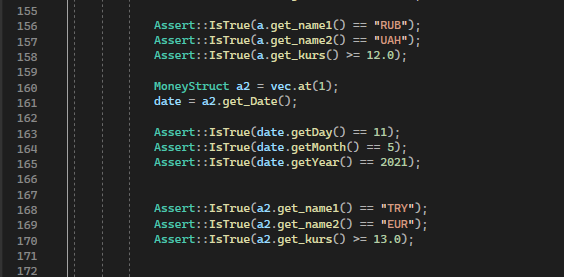


Рисунок 9 – Исправление ошибок V550.

Так же были найдены ошибки вида “Medium”. На рисунке 10 представлены найденные ошибки.

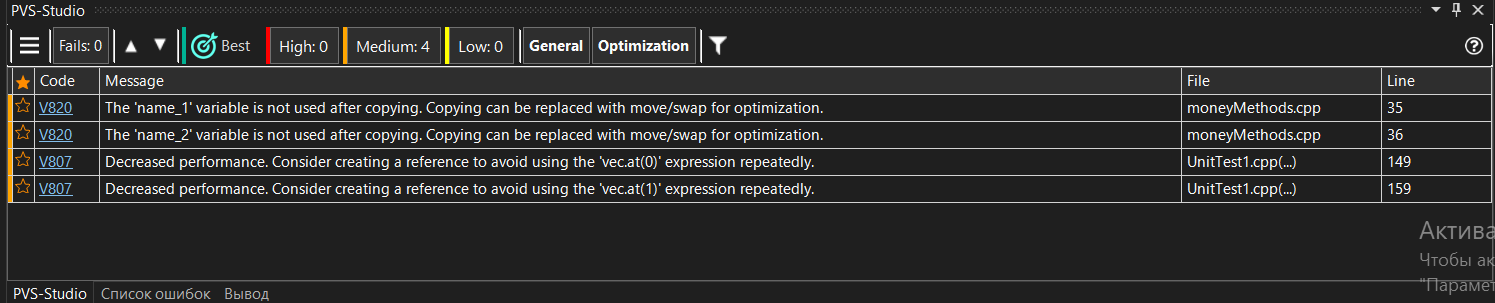


Рисунок 10 – Ошибки вида “Medium”, найденные в программе.

Как видно из рисунка 10, была найдено две ошибки одного типа – V820, которые имеют вид: “ *V820. The ‘name\_1’ variable is not used after copying. Copying can be replaced with move/swap for optimization. MoneyMethods.cpp in line 35, 36.*”. Суть ошибки в том что в функции createMoneyFromFile(), создавались переменные, которые после копирования больше не использовались, поэтому целесообразнее использовать move(). На рисунке 11 представлено исправление этой ошибки.

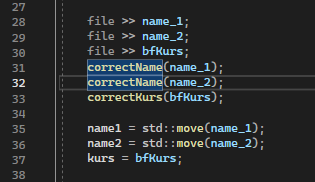


Рисунок 11 – Исправление ошибки V820.

Так же были найдены две ошибки одного типа – V807, которые имеют вид: *“Decreased perfomance. Comsider creating a reference to avoid using the ‘vec.at(0)’ expression repeatedly.”* Так как код содержит цепочки обращений к одному и тому же объекту более целесообразней заменить на обращение к временному указателю (было созданно два временный указателя ‘a’ и ‘a2’). На рисунке 12 представлено исправление этой ошибки.

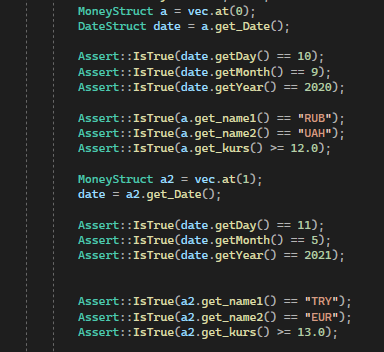


Рисунок 12 – Исправление ошибки V807.

# **Исследование программы на возможные утечки памяти с использованием Visual Leak Detector**

Для проверки программы на всевозможные утечки памяти была использована утилита Visual Leak Detector. Результат работы утилиты выводится в консоль. Для обеспечения её работоспособности необходимо подключить её, используя директиву #include.

После подключения достаточно лишь запустить программу для проверки её на возможные утечки. Уведомление о том, что утечки памяти в программе отсутствуют, изображено на рисунке 13.

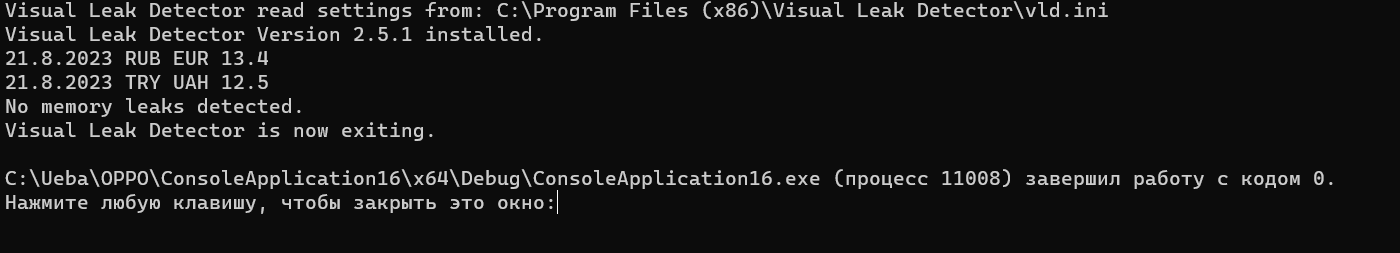


Рисунок 13 – Уведомление об отсутствии утечек памяти в программе.

Как видно из рисунка 13, в программе утечек памяти не наблюдается.

По итогу можно сказать, что программа была исследована на предмет всевозможных ошибок и утечек с использованием различных инструментов.