

ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА АНАЛИЗА КОДА ПРОГРАММ

1. Исследование программы с использованием статического анализатора Cppcheck

Используя статический анализатор, удалось выяснить, что в программе присутствует одна ошибка производительности. Результаты работы анализатора представлены на рисунке 1.

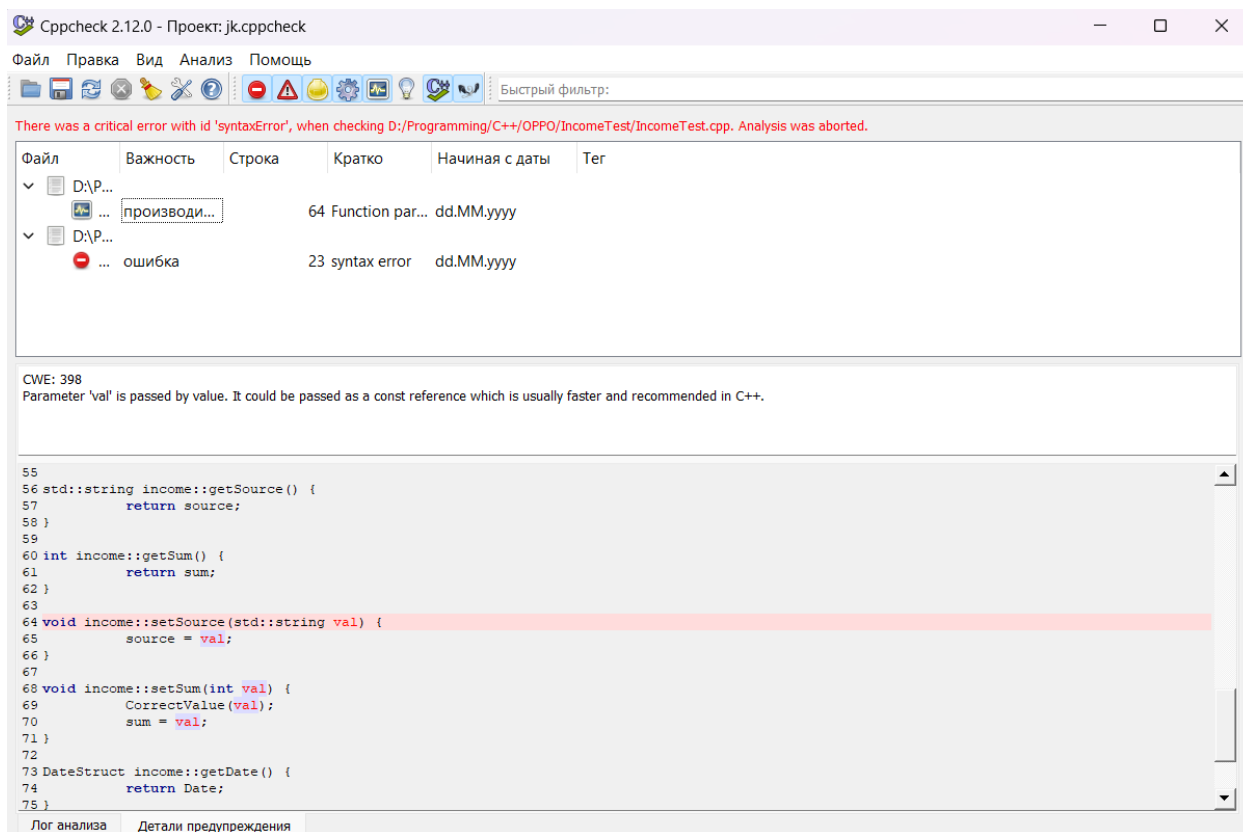


Рисунок 1 – результат работы анализатора Cppcheck.

Для исправления ошибки типа “CWE398” было принято решение об использовании ссылок при передаче аргументов в функцию. Исправление ошибки представлено на рисунке 2.

```
67  
68 void income::setSource(std::string &val) {  
69     source = val;  
70 }
```

Рисунок 2 – Исправление ошибки “CWE398”

2. 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.

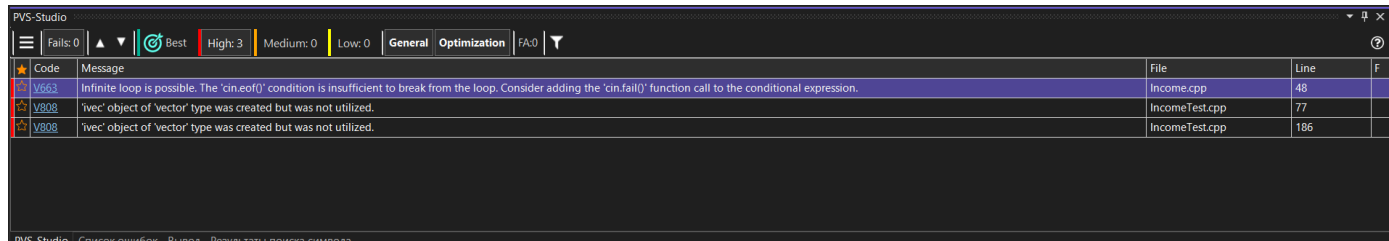


Рисунок 3 – Ошибки вида “High”, найденные в программе.

Как видно из рисунка 3, было найдено две ошибки в файлах: Income.cpp и incomeTest.cpp. Для исправления первой ошибки “-V663” было принято решение добавить в условие цикла while дополнительный параметр. Исправление первой ошибки представлено на рисунке 4.

```
46
47 void income::readIncome(std::istream& file, std::vector<income>& pvec) {
48     while (!file.eof() && !std::cin.fail()) {
49         income incomeData = income::createFromStream(file);
50         pvec.push_back(incomeData);
51     }
52 }
```

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 “V808” было принято решение об удалении лишних неиспользуемых векторов в файле incomeTest.cpp в функциях validIncomePrintTest() и invalidIncomePrintTest(). Исправление второй ошибки представлено на рисунке 5.

```

73
74     TEST_METHOD(validIncomePrintTest) {
75         income inc;
76         DateStruct date;
77         std::string str = "Source1";
78
79         date.setDD(12);
80         date.setMM(12);
81         date.setYY(21);
82
83         inc.setDate(date);
84
85         inc.setSum(3200);
86         inc.setSource(str);
87
88         std::string res = "Дата: 12.12.21 Источник дохода: Source1 Сумма: 3200\n";
89
90         std::stringstream x;
91         inc.print(x);
92         Assert::AreEqual(x.str(), res);
93     }

```

Рисунок 5 – Исправление ошибки V808

Аналогичные действия были произведены и со второй функцией.

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

Code	Message	File	Line	F
V688	The 'dd' function argument possesses the same name as one of the class members, which can result in a confusion.	Date.cpp	59	
V688	The 'mm' function argument possesses the same name as one of the class members, which can result in a confusion.	Date.cpp	59	
V688	The 'yy' function argument possesses the same name as one of the class members, which can result in a confusion.	Date.cpp	59	
V688	The 'sum' function argument possesses the same name as one of the class members, which can result in a confusion.	Income.cpp	36	

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

Как видно из рисунка, в трёх файлах (Date.cpp, Income.cpp), были найдены ошибки одного вида – V688. Для решения ошибки V688, было принято решение о простом переименовании параметров функции Correct(). Исправление ошибки представлено на рисунке 7.

```

58
59 void DateStruct::Correct(int day, int month, int year) {
60     int maxDays = 31;
61     switch (month) {
62     case 2:
63         maxDays = isLeapYear(year) ? 29 : 28;
64         break;
65     case 4:
66     case 6:
67     case 9:
68     case 11:
69         maxDays = 30;
70         break;
71     }
72
73     if (day < 1 || !(day <= maxDays)) throw std::runtime_error("День введён неверно!");
74     if (month < 1 || month > 12) throw std::runtime_error("Месяц введён неверно!");
75     if (year < 0 || year > 99) throw std::runtime_error("Год введён неверно!");
76 }

```

Рисунок 7 – Исправление ошибки V688.

Как видно из рисунка, старые параметры функции `Correct()` – `dd`, `mm`, `yy`, были заменены на более осмысленные `day`, `month`, `year`, что помогло решить ошибку.

Для исправления второй ошибки V688 было принято аналогичное решение о переименовании параметра функции `CorrectValue()`. Исправление ошибки V688 представлено на рисунке 8.

```
35
36 void income::CorrectValue(int Sum) {
37     if (Sum < 0) throw std::runtime_error("Неверно введена сумма!");
38 }
```

Рисунок 8 – Исправление ошибки V688.

Помимо всех вышеперечисленных ошибок, были найдены также ошибки вида “Medium”. На рисунке 9 представлены найденные ошибки.

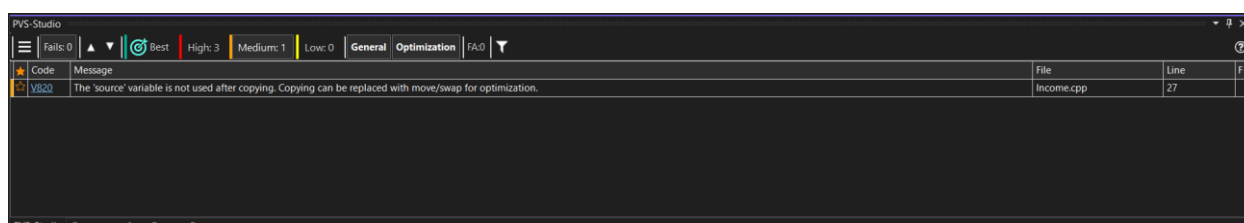


Рисунок 9 – Ошибки вида “Medium”, найденные в программе.

Как видно из рисунка 10, была найдена ошибка типа V820. На рисунке 10 представлено исправление этой ошибки.

```
25
26 incomeData.CorrectValue(sum);
27 incomeData.source = std::move(source);
28 incomeData.sum = sum;
```

Рисунок 10 – Исправление ошибки V808.

Как видно из рисунка 10, для устранения ошибки было принято решение об использовании `std::move` в качестве функции, позволяющей передать значение `source` в `incomeData.source`, что помогло исправить ошибку.

После выполнения всех вышеперечисленных ошибок, анализатор уведомил о том, что ошибок больше в программе нет. Уведомление представлено на рисунке 11.

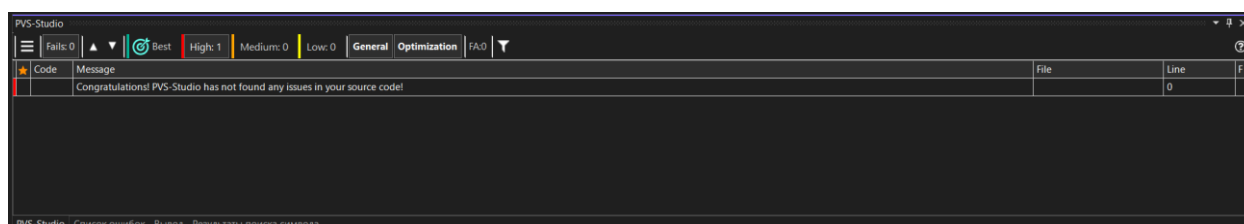
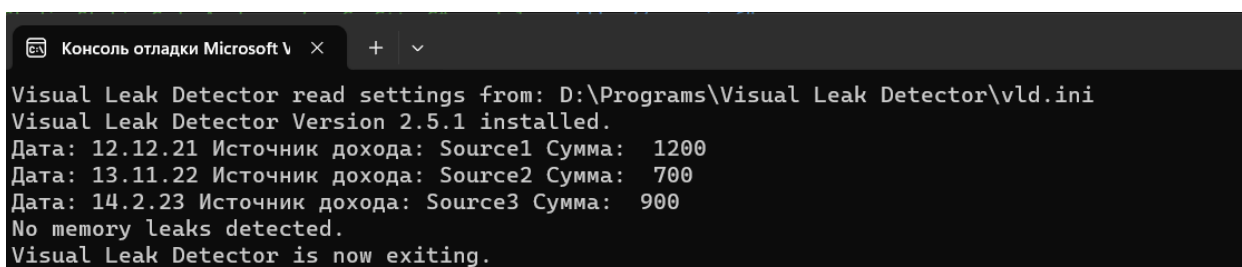


Рисунок 11 – Уведомление об отсутствии ошибок.

3. Исследование программы на возможные утечки памяти с использованием Visual Leak Detector

Для проверки программы на всевозможные утечки памяти было принято решение об использовании утилиты Visual Leak Detector. Результат работы утилиты по умолчанию 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 12.



```
Консоль отладки Microsoft V  x  +  v
Visual Leak Detector read settings from: D:\Programs\Visual Leak Detector\vld.ini
Visual Leak Detector Version 2.5.1 installed.
Дата: 12.12.21 Источник дохода: Source1 Сумма: 1200
Дата: 13.11.22 Источник дохода: Source2 Сумма: 700
Дата: 14.2.23 Источник дохода: Source3 Сумма: 900
No memory leaks detected.
Visual Leak Detector is now exiting.
```

Рисунок 12 – Уведомление об отсутствии утечек памяти в программе.

Как видно из рисунка 12, в программе утечек памяти не наблюдается.