1. **Разберите программу, представленную в файле**[**task\_for\_lecture5.cpp**](https://canvas.instructure.com/courses/1844563/files/87817234/download?wrap=1)**. В программе создается 2 потока, каждый из которых вычисляет средние значения матрицы, один по строкам исходной матрицы***matrix***, а другой - по столбцам. Запустите программу и убедитесь в ее работоспособности.**

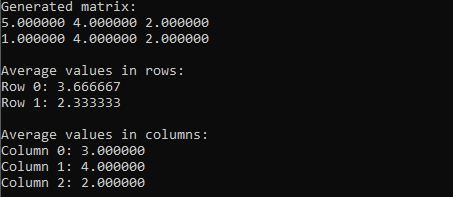


Рис.1 – Определение средних значений строк и столбцов

Вычислим эти значения:

Вычисления совпали => Программа работает верно.

1. **Проанализируйте программу и введите в нее изменения, которые, по Вашему мнению, повысят ее производительность.**

Одним из пунктов по оптимизации кода является правильный подбор типов данных. Тип unsigned занимает 4 байта, хотя status может принимать только два значения. Поэтому я решила заменить тип unsigned на тип bool, который занимает только 1 байт.

bool main()

{

//const unsigned ERROR\_STATUS = -1;

//const unsigned OK\_STATUS = 0;

const bool ERROR\_STATUS = false;

const bool OK\_STATUS = true;

bool status = OK\_STATUS;

try

{

…

1. **Определите с помощью**Intel Parallel Inspector**наличие в программе таких ошибок как: *взаимная блокировка*, *гонка данных*, *утечка памяти*. Сделайте скрины результатов анализа**Parallel Inspector **(вкладки***Summary***,***Bottom-up***) для всех упомянутых ошибок*,*где отображаются обнаруженные ошибки, либо отражается их отсутствие. Запускайте анализы на разных уровнях (***Narrowest***,***Medium***,***Widest***).**

Проведем проверку программы на проблемы с памятью с помощью инструмента Inspector.

Как видно на скриншоте 2, в 4 местах в коде есть Memory leak, или утечка памяти. Это значит, что память выделили, но не освободили.

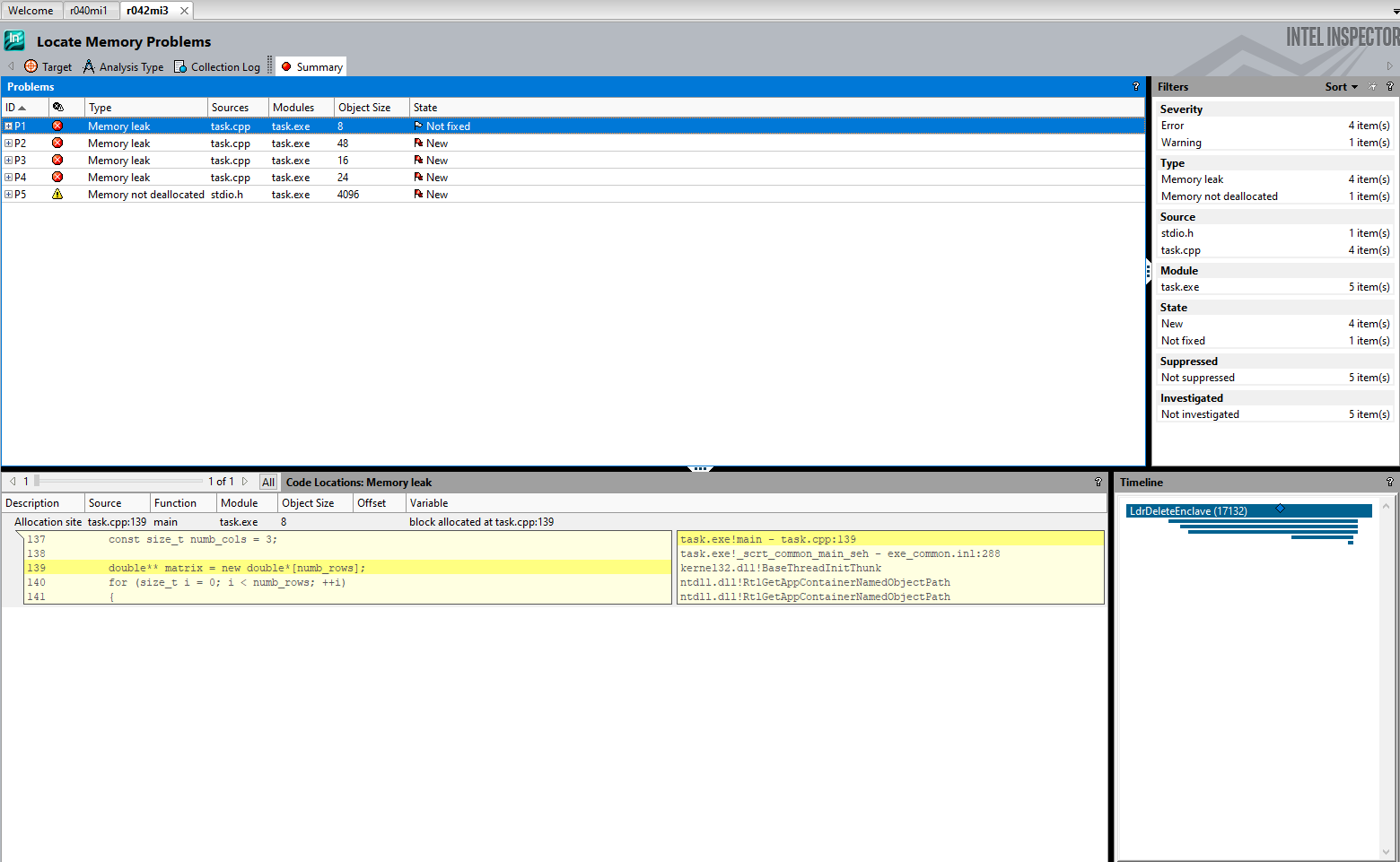


Рис.2 – Результат анализа Inspector. Обнаружены утечки памяти

После была проведена проверка на ошибки взаимодействия потоков (гонки данных и взаимные блокировки), которые были не найдены (рисунок 3).

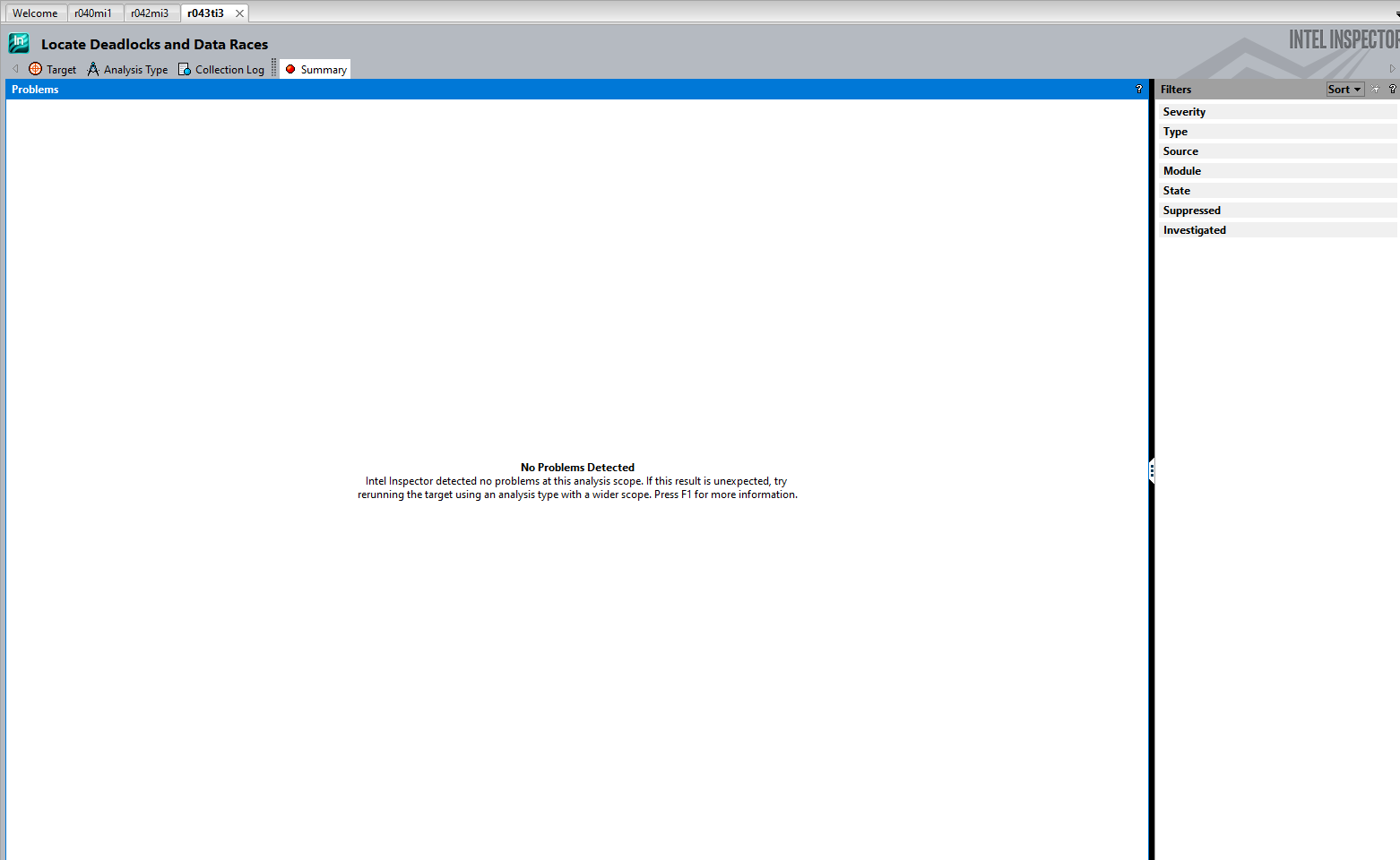


Рис.3 – Результат анализа программы на ошибки взаимодействия потоков

1. **Измените код программы таким образом, чтобы***Inspector***при проверке не находил в программе ошибок, перечисленных в**п. 3. **Сделайте скрины результатов запуска**Parallel Inspector**.**

Для устранения утечки памяти необходимо освободить выделенную память:

//Освобождение памяти--------------------

for (size\_t i = 0; i < numb\_rows; ++i)

{

delete[] matrix[i];

}

delete[]matrix;

delete[]average\_vals\_in\_cols;

delete[]average\_vals\_in\_rows;

//---------------------------------------

После добавления данных строк, отвечающих за освобождение памяти, ошибок с памятью было не найдено. Результат анализа приведен на рисунке 4.

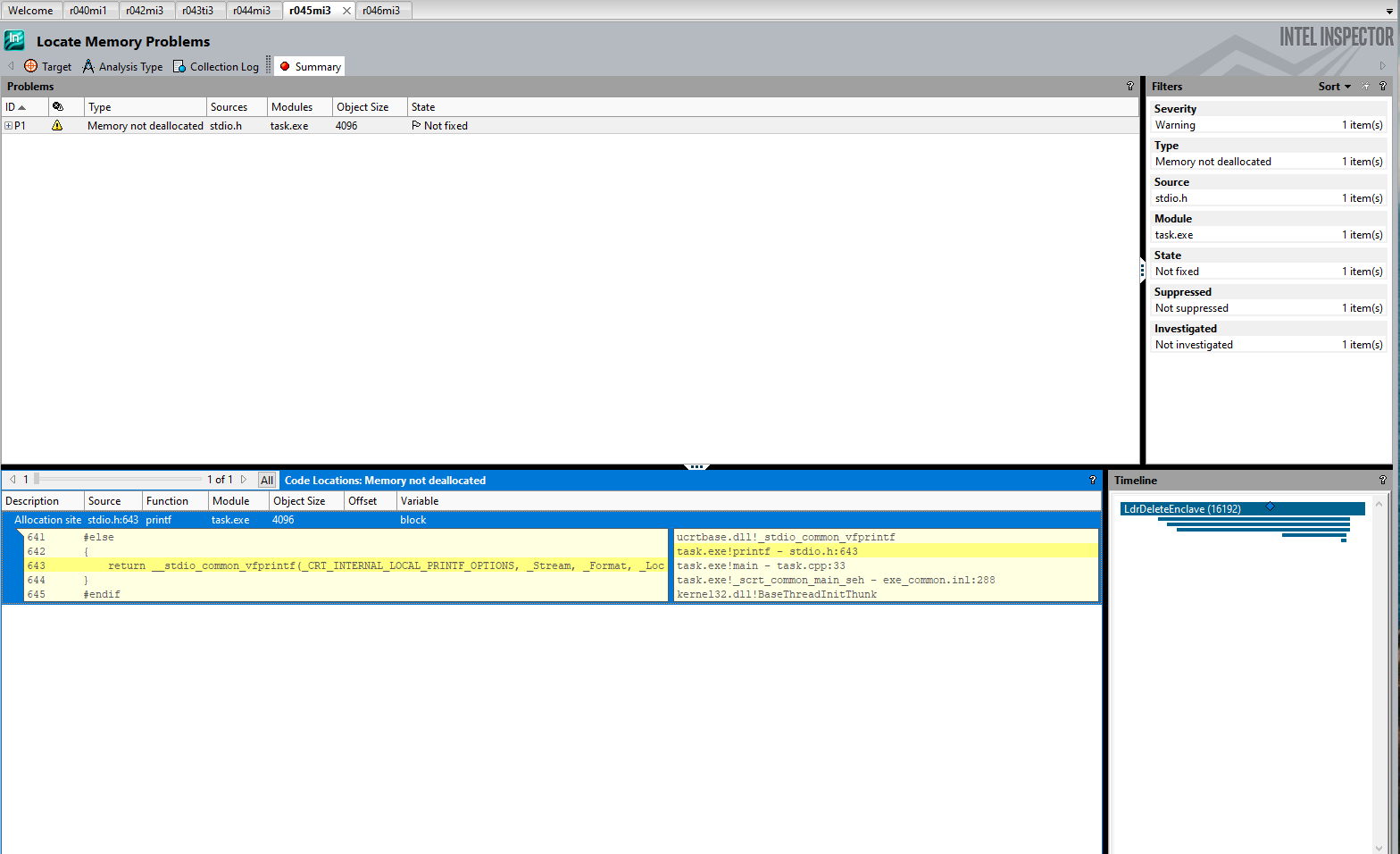


Рис.4 – Утечки памяти не обнаружены

**Выводы:** В рамках данной лабораторной работы мы продолжили знакомится с инструментом для поиска ошибок Inspector. Провели анализ до и после устранения данной ошибки.

При выполнении работы использовалась Visual Studio 2017 и IPS 2019.

Операционная система: Windows 10

Процессор: 2.80ГГц, 4 физических ядра, 8 логических ядер