Разработать плагин для CLion, который позволит открывать/сохранять PVS-Studio отчет , осуществлять навигацию по предупреждениям и запускать анализ текущего проекта.

## Вводная часть к плагинам

Разработать статический анализатор кода — это ещё не все. Анализатор, как правило, представляет собой инструмент командной строки, которым в таком виде не все захотят пользоваться. Этот факт наталкивает на мысль, что для более удобной и приятной работы с анализатором нужны различные плагины, которые будут позволять запускать анализ, просматривать отчеты анализатора, сохранять отчеты и так далее, не вдаваясь в подробности консольного запуска анализатора.

На сегодняшний день у нас реализованы плагины:

* для C++: Visual Studio
* для C#: Visual Studio, JetBrains Rider
* для Java: IntelliJ IDEA, gradle, maven.

Но много плагинов не бывает! На данный момент для С++ разработчиков Visual Studio считается наиболее предпочитаемой IDE. Но время не стоит на месте, конкуренты с каждым годом потихоньку отбирают позиции у лидера, среди которых CLion. А пока CLion набирает популярность, нужно задуматься о плагине. Поэтому наше задание связано c тем, что мы хотим пополнить ряды наших плагинов, а именно, плагином для CLion.

[CLion](https://www.jetbrains.com/clion/) — IDE от JetBrains для С/С++ разработчиков. Любой плагин для CLion является расширением IntelliJ Platform. IntelliJ Platform — это open-souce платформа от JetBrains, предназначенная для создания IDE, плагинов, а также различных инструментов разработчика.

[Статья](https://www.jetbrains.com/help/clion/develop-plugins-for-clion.html) раскрывает основы разработки плагина IntelliJ Platform, которые позволят вам сделать первые шаги по реализации вашего плагина CLion.

## О задании

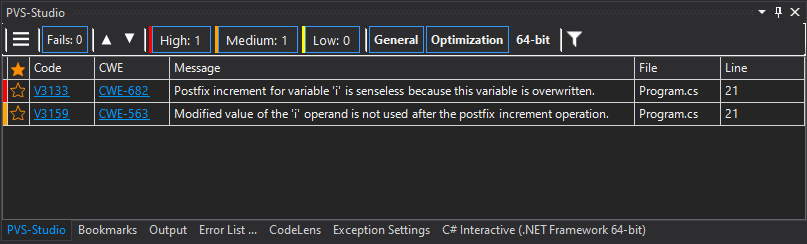
Нужно сделать плагин для CLion, который будет соответствовать минимальным требованиям:

* Меню выбора для открытия/сохранения отчета анализатора
* Окно с загруженными предупреждениями
* Двойным щелчком на предупреждение открывать файл на строке с ошибкой
* Запускать анализ решения / проекта

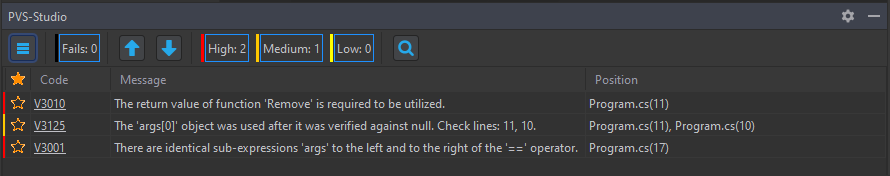
Чтобы иметь представление о том, что из себя представляет отчет анализатора, загляните [сюда](https://github.com/PVSStudioHacks/TulaHack2021/blob/main/sources/report.json)

В качестве визуального представления о плагине вы можете посмотреть на уже реализованные нами: Visual Studio/JetBrains Rider/InteliJ IDEA. Чтобы вживую пощелкать плагины, их можно скачать [здесь](https://www.viva64.com/ru/pvs-studio-download/).

Вот пример того, как это выглядит окно с предупреждениями в Visual Studio 2019:



А вот пример как это выглядит в Rider/IntelliJ IDEA:



Для реализации запуска анализа понадобится сам анализатор PVS-Studio. Его можно скачать вот [тут](https://www.viva64.com/ru/pvs-studio-download/) (Windows, exe). Для запуска анализа необходимы лицензионные данные. Их вы можете найти [здесь](https://github.com/PVSStudioHacks/TulaHack2021/blob/main/sources/PVS-Studio.lic).

Для того, чтобы ввести полученные данные, выполните:

PVS-Studio\_Cmd.exe credentials --username NAME --serialNumber XXXX-XXXX-XXXX-XXXX

Чтобы реализовать запуск анализа с помощью PVS-Studio нужно воспользоваться утилитой PVS-Studio\_Cmd.exe. После установки PVS-Studio ее можно будет найти по пути "C:\Program Files (x86)\PVS-Studio". Вот пример простого запуска анализа:

PVS-Studio\_Cmd.exe -t "D:\BestProject\BestProjectCpp.sln" \

-o "D:\BestProject\results.plog"

Где в '-t' нужно передать путь до решения или проекта. Во флаг '-o' передается путь до выходного файла, который и будет содержать в себе результаты работы. Подробнее о данной утилите Вы можете почитать [здесь](https://www.viva64.com/ru/m/0035/).