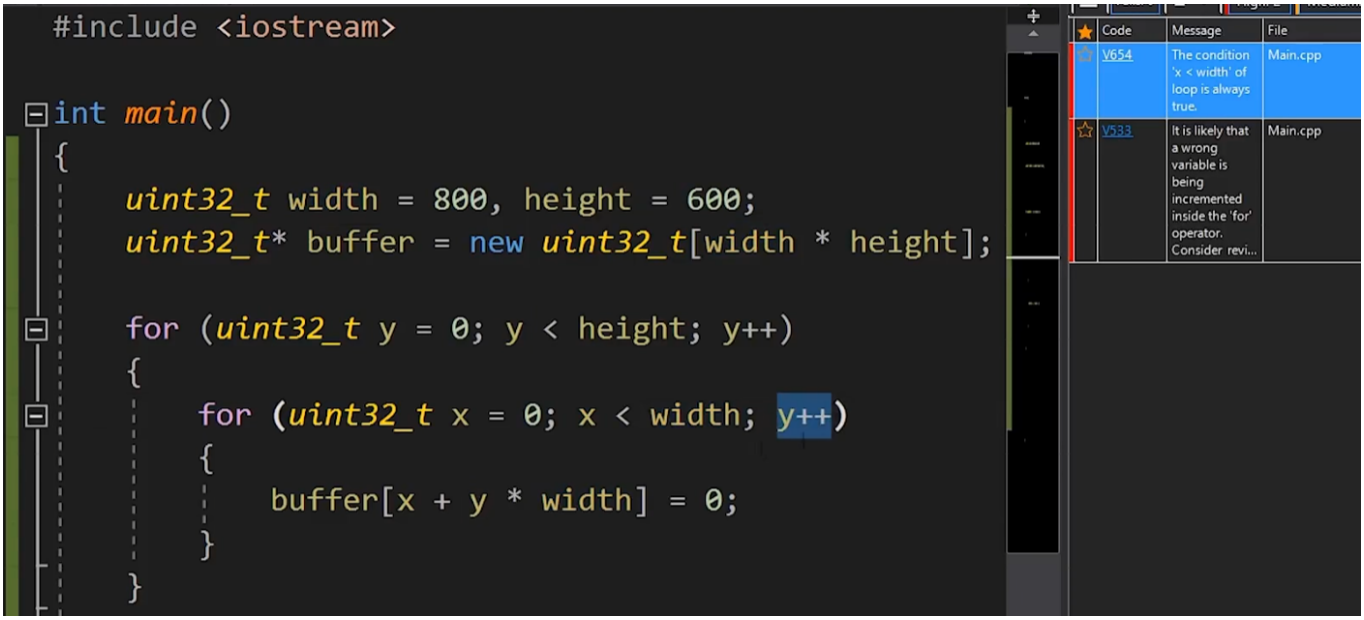
87 Static Analysis in C++

C++ 静态分析是在不实际执行代码的情况下对源代码或二进制代码进行分析的过程。这是通过检查代码中的模式、语法、结构和其他属性来完成的。静态分析的目的是在代码运行之前找出错误、漏洞或不符合编码标准的地方。



> Cherno 所用的是PVS Studio，感兴趣的话可以自行了解。

以下是C++静态分析的几个关键点：

1. **错误检测**: 静态分析工具可以检测出许多常见的编程错误，例如空指针解引用、未初始化的变量、内存泄漏等。
2. **代码质量提升**: 静态分析可以帮助识别出可能导致未来错误的代码片段、冗余的代码或不符合编码规范的代码。
3. **安全性**: 许多安全漏洞可以通过静态分析工具在代码进入生产阶段之前被检测出来。这对于防止潜在的安全威胁至关重要。
4. **性能**: 某些静态分析工具可以提供有关可能的性能瓶颈的反馈。
5. **代码审查的辅助**: 使用静态分析工具可以使代码审查过程更加高效，因为它可以自动标识出需要关注的问题区域。
6. **集成**: 许多静态分析工具都可以集成到持续集成/持续部署(CI/CD)流程中，使得代码的检查和验证成为自动化构建的一部分。

常见的C++静态分析工具包括：

- **Clang Static Analyzer**: 这是 Clang 编译器的一部分，它可以检测 C、C++ 和 Objective-C 代码中的各种问题。
- **Cppcheck**: 这是一个开源工具，用于检测 C 和 C++ 代码中的错误。
- **Coverity**: 是一个商业工具，提供了广泛的静态分析功能，特别是针对安全性问题。
- **Visual Studio Code Analysis**: 如果你使用 Visual Studio，它自带的代码分析工具也非常强大。

使用静态分析工具的一个常见误区是它们可以检测所有类型的错误或问题。实际上，任何工具都无法找到代码中的所有问题，但它们可以作为提高代码质量和安全性的有力工具。为了获得最佳结果，建议结合使用静态分析、动态分析和人工代码审查。