# **Конспект №4: этапы компиляции в C++.**

## *Препроцессинг*

Препроцессор - это макропроцессор, который преобразовывает программу для дальнейшнего компилирования. На данной стадии происходит работа с директивами препроцессора:

1. Комментарии заменяются на пустые строки - //
2. Хэдеры рекурсивно проходят стадии препроцессинга и включаются в файл - #include
3. Макроподстановки заменяются их значениями - #define
4. Обрабатываются директивы условной компиляции - #if, #ifdef, #ifndef

Препроцессинг возможно запустить вручную в G++ с помощью флага -E, который сообщает компилятору, что компилировать файл не нужно, а только провести его препроцессинг.

g++ -E driver.cpp -o driver.ii

## *Компиляция*

Компиляция - процесс преобразования полученного на прошлом шагу код без директив в ассемблерный код. Это промежуточный шаг между высокоуровневым языком и машинным кодом.

1. *Лексический анализ* - последовательность символов преобразуется в последовательность лексем.
2. *Синтаксический анализ* - последовательность лексем преобразуется в дерево разбора.
3. *Семантический анализ* - обработка дерева разбора: привязка идентификаторов к декларациям и типам, проверка совместимости.
4. *Оптимизация* - удаление излишних конструкций и упрощение кода с сохранением его смысла.

Источники:

* [дерево разбора](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7)

## *Ассемблирование*

Ассемблер преобразует ассемблерный код в машинный, сохраняя его в объектном файле. Обычно этот процесс относится к компиляции, однако его можно контролировать.

Флаг -S сообщает компилятору остановиться после стадии компиляции, получив ассемблерный код:

g++ -S driver.ii -o driver.s

Флаг as сообщает компилятору произвести стадию ассемблирования, получив объектный файл с машинным кодом:

as driver.s -o driver.o

## *Компоновка*

Компоновщик связывает все объектные файлы и статические библиотеки в единый исполняемый файл, который можно будет запустить. Компоновщик или линкер способен строить связи между данными среди множества других файлов благодаря таблице символов, хранящих ссылки, имена переменных, функций, классов, объектов и т.д. В итоге получается исполняемый файл:

g++ driver.o -o driver

Источники:

* [таблица символов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_%D1%81%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB%D0%BE%D0%B2)

## *Загрузка*

Последний этап при компиляции программы - вызов загрузчика для загрузки программы в память. На данном этапе возможно подключение динамических .dll библиотек.

Источники:

* [динамические и статические библиотеки](https://qna.habr.com/q/489534)