**Лабораторная работа № 1**

1. Скачаем библиотеку boost с помощью утилиты wget. Адрес для скачивания.

Команда wget скачивает один файл и сохраняет его в текущей директории. Во время загрузки видны прогресс, размер файла, дату его последнего изменения, а также скорость загрузки.

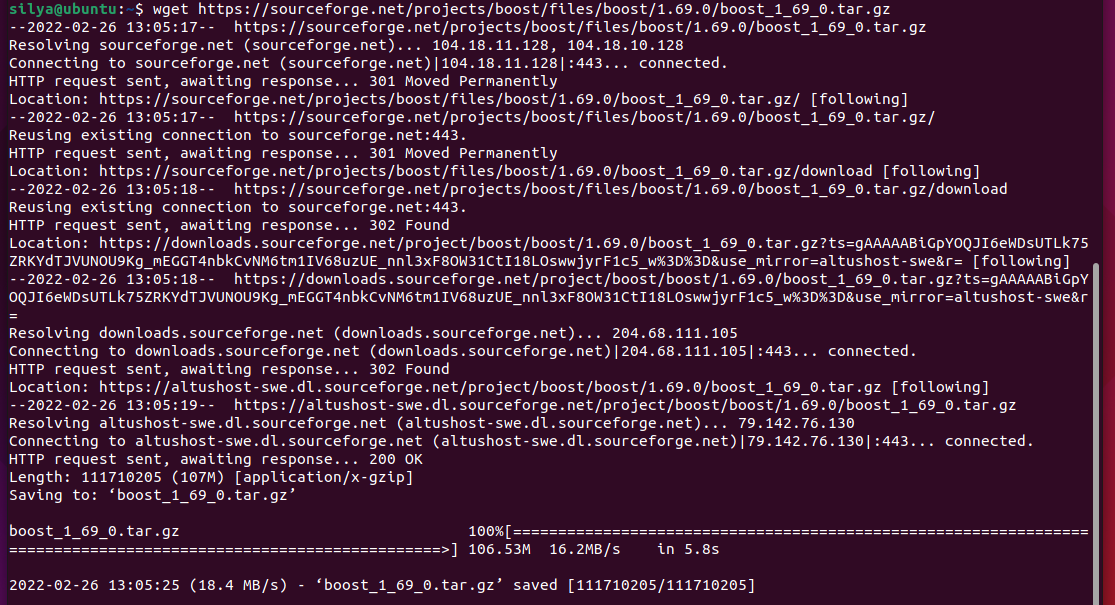


Рисунок. 1. Скачивание библиотеки boost

Тильда (~) является сокращением для домашнего каталога. Это означает, что не нужно вводить полный путь к домашней директории в командах.

1. Разархивация скаченного файла в директорию

tar используется для объединения или сжатия нескольких файлов в один файл. В утилите tar есть 4 основных режима работы.

c — Создать архив из файлов или каталогов

x — Извлечь архив

r — Добавить файлы в конец архива

t — список содержимого архива

Опция -x – разворачивание архива.

Опция -z перенаправляет вывод в команду gzip

Имя архива, который создаётся или разворачивается передаётся после параметра -f.



Рисунок. 2. Разархивация

Убедимся, что содержимое распакованного архива совпадает с оригинальной директорией с помощью команды ls -l -F:

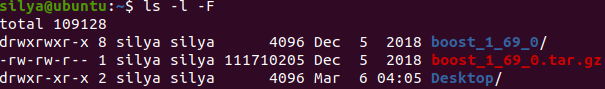
****

Рисунок. 3. Проверка разархивации

1. Подсчет количества файлов в директории не включая вложенные файлы

Для начала, с помощью команды pwd проверим текущую директорию



Рисунок. 4. Проверка текущей директории

Как видим, мы находимся в домашней директории. Для того, чтобы подсчитать количество файлов в директории boost\_1\_69\_0, мы изменим текущую директорию с помощью команды cd



Рисунок. 5. Переход в другую директорию

Далее делаем подсчет количества файлов в директории, не включая вложенные файлов. Используем команды:

ls **–** команда, отображающая список файлов и подкаталогов, находящихся в конкретной директиве.

wc -l – вывод только количества строк в объекте.



Рисунок. 6. Подсчет файлов в директории (без вложенных файлов)

1. Подсчитайте количество файлов в директории, включая вложенные директории.

Произведем подсчет файлов в директории, включая вложенные директории, с использованием следующей команды:

****

Рисунок 7. Подсчет файлов в директории (включая вложенные директории)

Команда find предназначена для поиска файлов и каталогов на основе специальных условий.

Команда find имеет такой синтаксис:

find [папка] [параметры] критерий шаблон [действие]

Используемые в этом пункте параметры:

-type f – искать только файлы.

wc -l – вывод только количества строк в объекте.

Итоговый результат: 61191 файл

1. Подсчитаем количество заголовочных файлов, файлов с расширением .cpp, сколько остальных файлов (не заголовочных и не .cpp).
   1. Количество заголовочных файлов

Для подсчета заголовочных файлов используем команду:



Рисунок 8. Подсчет заголовочных файлов

В программировании заголовочный файл (англ. header file) или подключаемый файл — файл, содержимое которого автоматически добавляется препроцессором в исходный текст в том месте, где располагается некоторая директива ({$I file.inc} в Паскале, #include <file.h> в Си). Иногда для заголовочных файлов языка C++ используют расширение .hpp.

-o – логическое ИЛИ. Позволяет команде find выполнить поиск на основе одного из критериев поиска.

-type – тип объекта поиска, в нашем случае: f – файл.

-name – поиск по имени. В данном случае нужны те, имена которых оканчиваются на .h, .hpp, .inc.

5.2. Количество файлов с расщирением .cpp

Команда для подсчета количества файлов с расширением .cpp:



Рисунок 9. Подсчет количества файлов с расширением .cpp

5.3. Количество остальных файлов (не заголовочных и не .cpp)

Команда для подсчета количества файлов:



Рисунок 10. Подсчет количества файлов

-a – логическое И. Объединяем несколько критериев поиска.

-not или ! – логическое НЕ. Инвертирует критерий поиска.

6. Нахождение полного пути до файла any.hpp внутри библиотеки boost

Так как по заданию необходимо осуществить поиск в библитеке boost, то перейдем в нее, сначала перейдя в директорию boost\_1\_69\_0, а затем в директорию boost с помощью команды cd. Поиск выполним аналогично пункту 5.

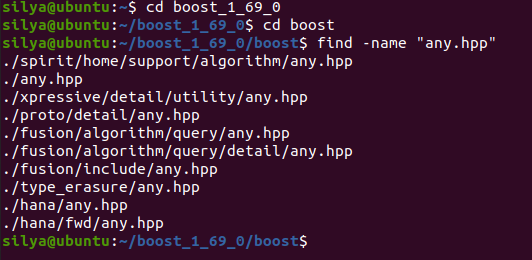


Рисунок 11. полного пути до файла any.hpp

7. Вывеcти в консоль все файлы, где упоминается последовательность boost::asio

Произведем вывод с использованием команды grep.

grep ищет в одном или нескольких входных файлах строки, соответствующие заданному шаблону, и записывает каждую соответствующую строку в стандартный вывод. Если файлы не указаны, grep читает из стандартного ввода, который обычно является выводом другой команды.

Синтаксис команды выглядит следующим образом:

$ grep [опции] шаблон [имя файла...]

Чтобы подавить grep вывод по умолчанию и печатать только имена файлов, содержащих сопоставленный шаблон, используем параметр -l.

Предусмотрим рекурсивный поиск, чтобы при поиске просмотреть все вложенные директории.

Для рекурсивного поиска шаблона в grep используем -r опцию (или --recursive ). Когда эта опция используется, grep будет выполняться поиск по всем файлам в указанном каталоге, пропуская символические ссылки, которые встречаются рекурсивно.

Чтобы перейти по всем символическим ссылкам , вместо этого -r используйте -R опцию (или --dereference-recursive ).

Команда выглядит следующим образом: grep -rl "boost::asio".

В связи с тем, что при выполнении команды на экран выводится большое количество информации, ниже (рис. 12) будет приведен скриншот только с частью выведенных данных.

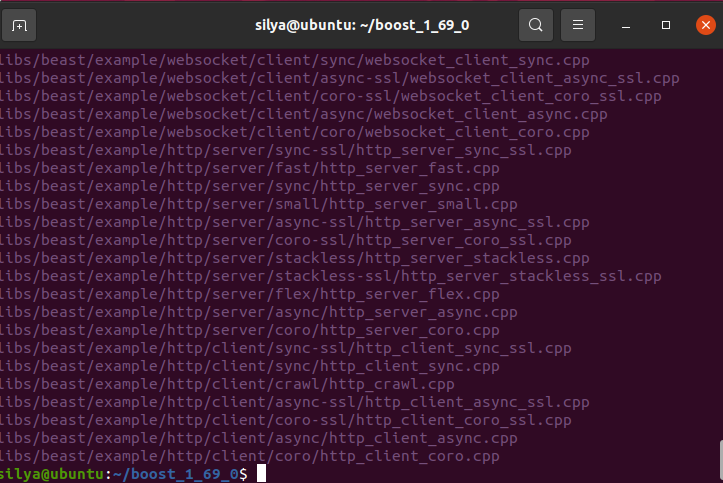


Рисунок 12. Вывод в консоль всех файлов, где упоминается последовательность boost::asio

8. Компиляция boost

Boost предоставляет бесплатные рецензируемые портативные исходные библиотеки C++. Библиотеки Boost предназначены для широкого использования и использования в широком спектре приложений.

Установим программу, используемую для генерации конфигурации сборки и сценария компиляции bootstrap.sh.

Для компиляции воспользуемся командой ./bootstrap.sh ./b2

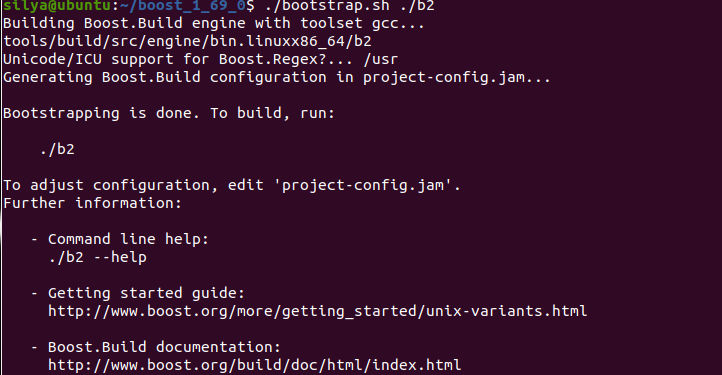


Рисунок 13. Компиляция boost

Чтобы выполнить фактическую сборку, мы вызываем b2, используя следующую команду: ./b2 и получаем результат, представленный на рисунке 14.



Рисунок 14. Результат выполнения команды

9. Перенос всех скомпилированных на предыдущем шаге статических библиотек в директорию ~/boost-libs

Перенос всех скомпилированных на предыдущем шаге статических библиотек в директорию ~/boost-libs осуществим с помощью команды: mv stage/lib ~/boost-libs.

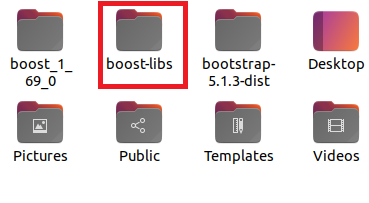


Рисунок 15. Результат переноса

10. Подсчет занимаемого дискового пространства для каждого файла в директории boost-libs

ls — утилита Unix, которая печатает в стандартный вывод содержимое каталогов. Параметр -s позволяет выводить размер файла в блоках, -h - выводит размеры в удобном для чтения формате.

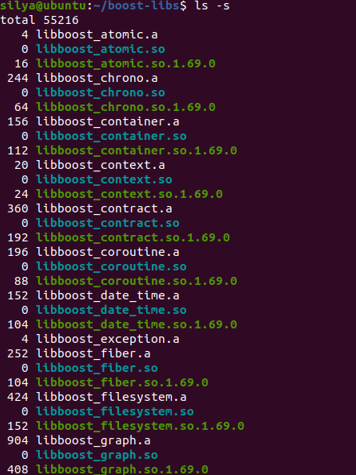


Рисунок 16. Вывод занимаемого дискового пространства для части файлов в директории boost-libs

11. Найдите топ10 самых «тяжёлых» файлов

Используем ту же команду, что и в предыдущем пункте, добавив параметр -S, позволяющий сортировать по размеру файла.

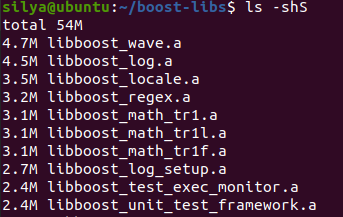


Рисунок 17. Топ 10 самых «тяжелых» файлов