Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра ЭВМ

Дисциплина: Операционные системы и системное программирование

ОТЧЁТ
к лабораторной работе №1
на тему
Знакомство с Linux/Unix и средой программирования. POSIX-совместимая файловая система.

Выполнил студент гр.230501 Лазовский И.А.

Проверил старший преподаватель кафедры ЭВМ
Поденок Л.П.

Минск 2024

1 УСЛОВИЕ ЛАБОРАТОРНОЙ РАБОТЫ.

Освоить эффективную работу с файлами в оболочке и mc.

Разработать программу dirwalk, сканирующую файловую систему и выводящую в stdout информацию в соответствии с опциями программы.

Формат вывода аналогичен формату вывода утилиты find.

dirwalk [dir] [options]

dir − начальный каталог. Если опущен, текущий (./).
options − опции.
-l – только символические ссылки (-type l)
-d – только каталоги (-type d)
-f -- только файлы (-type f)
-s — сортировать выход в соответствии с LC_COLLATE
Опции могут быть указаны как перед каталогом, так и после.
Опции могут быть указаны как раздельно, так и вместе (-l -d, -ld).
Если опции ldf опущены, выводятся каталоги, файлы и ссылки.

Для обработки опций рекомендуется использовать getopt(3) или gengetopt(1).

2 ОПИСАНИЕ АЛГОРИТМОВ И РЕШЕНИЙ.

Данная программа предназначена для рекурсивного обхода файлов и директорий в указанной директории. В данной программе используются две основные функции processing_options(), dirwalk().

2.1. Функция processing_options()

Функция processing_options() используется для обработки параметров командной строки и последуещего вызова функции dirwalk с необходимыми параметрами.

2.2. Функция dirwalk()

Функция dirwalk осуществляет рекурсивный обход директории и выводит информацию о каждом файле. Она принимает параметры, определяющие, какие типы файлов включать в вывод, а также начальный путь, который используется для поиска в нем определенных файлов.

Открывается директория с помощью opendir.

Происходит итерация по всем элементам в директории с помощью readdir.

Для каждого элемента проверяется его тип.

Если элемент - директория, рекурсивно вызывается dirwalk для обработки этой директории.

При помощи макросов проверяет тип считанного элемента.

После обработки всех элементов в директории, она закрывается с помощью closedir.

3. ФУНКЦИОНАЛЬНАЯ СТРУКТУРА ПРОЕКТА.

Проект собирается с помощью makefile. Для запуска проекта нам требуется в терминале заупустить программу dirwalk, далее указать путь к файлу и основные опции -l(символические ссылки), -d(каталоги), -f(файлы). Опции предназначенны для того чтобы определить что именно мы хотим видеть в данной директории. Так же можно опустить опции и программа выведет всю информацию о сведениях в каталоге. Dirwalk должен соответсвовать утилите $ find.

В проекте имеется каталог для сборки debug и release. Каталог git для системы контроля версий моего проекта. Директория src с исходным кодом. И makefile для компиляции и сборки моего проекта.

4. ПОРЯДОК СБОРКИ И ИСПОЛЬЗОВАНИЯ.

Для компиляции и сборки проекта используется makefile.

Порядок сборки:

1) Задание переменных:

- DEBUG и RELEASE - пути к каталогам для отладочной и релизной сборки соответственно.

- OUT_DIR - текущий каталог для выходных файлов (по умолчанию используется отладочная сборка).

- FLAGS_DEBUG и CFLAGS_RELEASE - флаги компиляции для отладочной и релизной сборки соответственно.

objects - объектные файлы для компиляции.

prog - имя выходного исполняемого файла.

2. Определение компилятора и флагов компиляции:

CC - компилятор (gcc).

CFLAGS - флаги компиляции, выбираются в зависимости от переменной MODE (отладочная или релизная сборка).

3. Определение зависимостей:

vpath - указание директорий для поиска файлов с исходным кодом и заголовочных файлов.

ifeq ($(MODE), release) - установка флагов и каталогов в случае релизной сборки.

4. Определение целей:

all - основная цель, компиляция всех объектных файлов и создание исполняемого файла.

`$(prog)` - правило для создания исполняемого файла.

`$(OUT_DIR)/%.o: %.c` - правило для компиляции каждого исходного файла в объектный.

Порядок использования.

1. Компиляция:

Для отладочной сборки: make или make MODE=debug

Для релизной сборки: make MODE=release

2. Очистка:

make clean - удаляет все объектные файлы и исполняемый файл.

3. Запуск:

После успешной компиляции запустите исполняемый файл, например: ./build/debug/dirwalk.

5. МЕТОД ТЕСТИРОВАНИЯ И РЕЗУЛЬТАТ ТЕСТИРОВАНИЯ.

Программа dirwalk должна выводить такие же данные как и утилита find в оболочке shell. Для сравнения строк мы используем | wc . Пример запуска программы:

1. Символические ссылки.

```
ilua@fedora:~/tar_working_dir/Лазовский И.А/lab01$ build/debug/test dirwalk /etc -l | wc
      858    2574    5534
```

Сравнение с $find:

```
ivan@fedora:~/tar_working_dir/Лазовский И.А/lab01$ find /etc -type l | wc
      858    858   42477
```

2. Директории.

```
ilua@fedora:~/tar_working_dir/Лазовский И.А/lab01$ build/debug/test dirwalk /etc -d | wc
      436    873   14662
```

Сравнение с $find:

```
ilua@fedora:~/tar_working_dir/Лазовский И.А/lab01$ find /etc -type l | wc
      436    436   9858
```

3. Файлы

```
ilua@fedora:~/tar_working_dir/Лазовский И.А/lab01$ build/debug/test dirwalk /etc -f | wc
     1637    4912   83407
```

Сравнение с $find:

```
ilua@fedora:~/tar_working_dir/Лазовский И.А/lab01$ find /etc -type f | wc
     1637    1638   60489
```

4. Без опций

```
ilua@fedora:~/tar_working_dir/Лазовский И.А/lab01$ build/debug/test dirwalk /etc | wc
     2931    8359  153416
```

Сравнение с $find:

```
ilua@fedora:~/tar_working_dir/Лазовский И.А/lab01$ find /etc | wc
 2931    2932  112824
```