Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики
и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Операционные системы и системное программирование

Лабораторная работа №1
на тему:
«Знакомство с Linux/Unix и средой программирования. POSIX-совместимая файловая система»

Выполнил:
Студент группы 350501
Шахлан Е.С.

Проверил:
Поденок Л. П.

Минск 2025

1 УСЛОВИЕ ЛАБОРАТОРНОЙ РАБОТЫ

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. Программа должна быть переносимой (возможности linux не используются, только POSIX).

2 ОПИСАНИЕ АЛГОРИТМОВ И РЕШЕНИЙ

Программа `dir_walk` предназначена для сканирования файловой системы, начиная с указанного каталога, и вывода списка файлов, каталогов и символических ссылок в формате, аналогичном утилите `find`. Она поддерживает различные опции для фильтрации вывода и сортировки результатов. Целью разработки является создание переносимой утилиты, совместимой с POSIX.

2.1 Разбор аргументов командной строки

Разбор аргументов командной строки начинается с определения начального каталога. Если он не указан, по умолчанию используется `./`. Затем анализируются переданные пользователем опции (`-l`, `-d`, `-f`, `-s`), и фильтрующие флаги объединяются в единую маску. Для удобства обработки аргументов применяется функция `getopt()`, позволяющая корректно распознавать и комбинировать переданные параметры.

2.2 Рекурсивный обход файловой системы

Рекурсивный обход файловой системы осуществляется с использованием `opendir()` и `readdir()` для получения содержимого каталогов. Для каждого найденного элемента вызывается `lstat()` с целью определения его типа. В данной реализации задействованы структуры `dirent` и `stat`. Однако, поскольку поле `d_name` в `dirent` не всегда заполняется (например, на файловых системах, не поддерживающих `d_name`, таких как NFS), `lstat()` применяется для точного определения типа файла. После этого проверяется соответствие элемента заданным фильтрам. Если обнаружен подкаталог, и его обработка не запрещена опциями, выполняется рекурсивный вызов для его обхода.

2.3 Формирование списка результатов

После обхода формируется список результатов, в который добавляются все подходящие файлы, каталоги и ссылки. Если была указана опция сортировки (`-s`), список упорядочивается согласно `LC_COLLATE` с помощью функции `strcoll()`.

2.4 Вывод результатов

На завершающем этапе программа выводит отфильтрованный и, при необходимости, отсортированный список файлов в `print_result`, соблюдая установленный порядок и применённые фильтры.

3 ФУНКЦИОНАЛЬНАЯ СТРУКТУРА ПРОЕКТА

Описание файла dirwalk.c

Файл dirwalk.c реализует программу для рекурсивного обхода директорий с возможностью фильтрации и сортировки результатов.

Локальный динамический массив options для хранения флагов определяет режим работы программы.

По умолчанию путь для сканирования – текущий каталог ("."), локаль устанавливается с помощью strcoll((*all_files_list)[j], (*all_files_list)[j + 1]) > 0) для корректной обработки символов, если включена сортировка (-s). Аргументы командной строки обрабатываются с помощью getopt(), поддерживаются опции: -l – обработка только символических ссылок, -d – обработка только директорий, -f – обработка только файлов, -s – сортировка результатов. Опции -l, -d и -f взаимоисключающие. Для рекурсивного обхода файловой системы используются функции dir_walk() и sort_list().

Dir_walk выполняет обход без сортировки с использованием opendir(), readdir() и lstat() для обработки каждого элемента в каталоге, учитывая типы файлов и заданные флаги (-l, -d, -f), и рекурсивно вызывает себя для обработки подкаталогов.

Sort_list выполняет обход с сортировкой содержимого каталога: пути всех элементов собираются в массив, сортируются с помощью самописной сортировки пузырьком и функции сравнения strcoll((*all_files_list)[j], (*all_files_list)[j + 1]) > 0), после чего обрабатываются аналогично dir_walk(). Вспомогательная функция strcoll() используется для сортировки строк с учетом текущей локали. Функция print_result выводит путь к файлу или каталогу. Макрос #define MAX_LENGHT 255 обеспечивает совместимость с POSIX-функциями.

Подключаемые библиотеки: <stdio.h> для ввода/вывода, <stdlib.h> для динамического выделения памяти и сортировки, <string.h> для работы со строками, <dirent.h> для работы с каталогами, <sys/stat.h> для lstat() и определения типа файла. Программа поддерживает следующие режимы работы: рекурсивный обход без сортировки (dir_walk) и рекурсивный обход с сортировкой (sort_list).

Пример использования:

./dirwalk – рекурсивный обход текущего каталога по умолчанию;

./dirwalk -s ~/Projects – обработка только директорий с сортировкой;

./dirwalk -f /var/log – обработка только файлов без сортировки;

./dirwalk -l /usr/bin – обработка символических ссылок.

4 ПОРЯДОК СБОРКИ И ИСПОЛЬЗОВАНИЕ

4.1 Сборка и запуск debug версии

```
$tar -xvf lab1.tar
$cd lab01/dirwalk
$make
$cd build/debug
```

4.2 Сборка и запуск release версии

```
$tar -xvf lab1.tar
$cd lab01/dirwalk
$make MODE=release
$cd build/release
```

4.3 Сборка и проверка программы при помощи утилиты Valgrind

```
$tar -xvf lab1.tar
$cd lab01/dirwalk
$make
$make valgrind
```

4.4 Использование программы

`dirwalk [путь] [опции]` — порядок описания опций и пути не имеет значения, как и вид указания опций(слитное, раздельное).

5 ТЕСТИРОВАНИЕ

Тестированием является сравнение результатов работы программы и команды find в одном каталоге с использованием утилиты wc(word count). Тестирование сортировки приводится с выводом результата программы и команды find в одном каталоге

5.1 Тестирование без флагов в корневом каталоге

```
shakhlanchik@fedora:~/tar_working_dir/Шахлан Е.С./lab01$ make
gcc -Wall -Wextra -std=c11 src/lab01.c -o lab01
./build/debug/dirwalk.o ./build/debug/function.o -o build/debug/dirwalk
$shakhlanchik@fedora:~/dirwalk$ ./build/debug/dirwalk ~ | wc
  36403   37065 3619853
$shakhlanchik@fedora:~/dirwalk$ find ~ | wc
  36403   37065 3619853
```

5.2 Тестирование с поиском символических ссылок в корневом каталоге

```
$shakhlanchik@fedora:~/dirwalk$ ./build/debug/dirwalk ~ -l | wc
          5        5      314
$shakhlanchik@fedora:~/dirwalk$ find ~ -type l | wc
          5        5      314
```

5.3 Тестирование с поиском каталогов в корневом каталоге

```
 $shakhlanchik@fedora:~/dirwalk$ ./build/debug/dirwalk ~ -d | wc
   2114    2167  171073
 $shakhlanchik@fedora:~/dirwalk$ find ~ -type d | wc
   2114    2167  171073
```

5.4 Тестирование с поиском файлов в корневом каталоге

```
$shakhlanchik@fedora:~/dirwalk$ ./build/debug/dirwalk ~ -f | wc
  34327   34938 3453728
$shakhlanchik@fedora:~/dirwalk$ find ~ -type f | wc
  34327   34938 3453728
```

5.5 Тестирование с сортировкой в текущем каталоге

```
$shakhlanchik@fedora:~/dirwalk$ ./build/debug/dirwalk -s
.
./build
./build/debug
./build/debug/dirwalk
./build/debug/dirwalk.o
./build/release
./Makefile
./src
./src/dirwalk.c
shakhlanchik@fedora:~/dirwalk$ find | sort
.
./build
```

```
./build/debug
./build/debug/dirwalk
./build/debug/dirwalk.o
./build/release
./Makefile
./src
./src/dirwalk.c
```