Министерство образования Республики Беларусь
Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ
Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Операционные системы и системное программирование

ОТЧЕТ
по лабораторной работе №1
на тему
“Знакомство с Linux/Unix и средой программирования. POSIX-совместимая файловая система.”

Выполнил:
студ. гр. 350501 Везенков М. Ю.

Проверил:
ст.пр. Поденок Л. П.

МИНСК 2025

СОДЕРЖАНИЕ

1 ЦЕЛЬ 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.

Для обработки опций рекомендуется использовать getopt(3).

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

2 ВЫПОЛНЕНИЕ РАБОТЫ

2.1 Описание алгоритма выполнения работы

Программа предназначена для обхода каталогов с использованием функции nftw(), выводя список файлов, каталогов и символических ссылок в зависимости от указанных параметров. Она начинается с анализа переданных аргументов командной строки для определения желаемых типов вывода (символические ссылки, каталоги, файлы) и того, следует ли сортировать результаты. Если сортировка включена, программа инициализирует динамический массив для хранения путей к файлам для последующей сортировки.

Затем программа начинает обход каталога в указанном каталоге (по умолчанию в текущем каталоге, если ничего не указано). Во время обхода она обрабатывает каждую запись файловой системы с помощью функции обратного вызова process_entry, которая проверяет тип каждого элемента по указанным флагам. Если элемент соответствует критериям, его путь либо немедленно выводится, либо добавляется в массив для последующей сортировки в зависимости от пользовательских параметров.

После завершения обхода, если была указана сортировка, программа сортирует собранные пути с использованием сопоставления с учетом локали и выводит их последовательно. Наконец, вся динамически выделенная память освобождается для обеспечения эффективности использования ресурсов.

2.2 Описание основных функций

Программа включает несколько ключевых подпрограмм, каждая из которых выполняет определённую задачу. Они взаимодействуют между собой, чтобы обеспечить удобный рекурсивный обход файловой системы и гибкость в настройке параметров обработки и вывода данных.

Функция main() представляет собой основной вход в программу. Она отвечает за обработку аргументов командной строки, которые определяют стартовый каталог, типы обрабатываемых файлов и необходимость сортировки. В случае, если пользователь указывает путь или опции в произвольном порядке, программа самостоятельно различает их, назначая каталог и параметры фильтрации. После анализа входных данных функция передаёт управление функции dirwalk(), которая реализует обход файловой системы. main() также управляет потоком выполнения программы и завершает её с соответствующим кодом выхода в случае ошибок.

Функция dirwalk() отвечает за рекурсивный обход каталогов и обработку их содержимого в соответствии с указанными флагами. Она использует библиотечную функцию nftw() для последовательного анализа файлов и директорий, вызывая для каждого элемента функцию обратного вызова

process_entry(). В зависимости от переданных опций (FLAG_SORT) эта функция может сразу выводить результаты или сначала накапливать их в массиве для последующей сортировки. После завершения обхода, если была включена сортировка, результаты сортируются с помощью qsort() и выводятся в консоль. В случае ошибок, таких как невозможность открыть каталог, dirwalk() сообщает пользователю о проблеме и завершает выполнение программы.

Функция process_entry() выполняет фильтрацию файлов и каталогов в зависимости от их типа. Получив данные о текущем объекте, она проверяет, соответствует ли он критериям, указанным пользователем. Например, она может включать в обработку символические ссылки, каталоги или обычные файлы, основываясь на соответствующих флагах. Если объект удовлетворяет условиям, вызов назначенной функции, такой как puts() или add_to_list(), обеспечивает либо немедленный вывод результата, либо добавление его в массив для сортировки. Эта функция является основным механизмом фильтрации и передачи данных на дальнейшую обработку.

Функция compare_func() используется для сортировки строк, представляющих пути к файлам. Она реализует сравнение с помощью функции strcoll(), что позволяет учитывать локальные настройки системы. Это обеспечивает корректный порядок сортировки с учётом специфики языка. compare_func() интегрируется с функцией qsort(), которая упорядочивает массив путей, подготовленный к выводу после завершения обхода.

2.3 Описание порядка сборки и использования

Для сборки проекта используется утилита make, которая автоматизирует процесс компиляции исходного кода и создания исполняемого файла. Процесс сборки организован так, чтобы исполняемый файл и объектные файлы размещались в соответствующих поддиректориях каталога build. При отсутствии этих директорий они создаются автоматически в процессе сборки, что упрощает организацию проекта и поддержание порядка.

Используется два режима сборки: режим отладки (debug) и релизный режим (release). В режиме отладки (make без указания переменной MODE) компилятор включает флаги для упрощения отладки кода, такие как -g2 и -ggdb. В релизном режиме (make MODE=release) включаются строгие флаги компиляции для обеспечения качества финальной сборки, такие как -Werror, запрещающий игнорирование предупреждений компилятора.

Для удаления всех скомпилированных файлов, включая объектные файлы и исполняемый файл, а также для очистки директории build предусмотрена команда make clean. Это гарантирует, что последующие сборки будут выполнены с нуля и не будут зависеть от остаточных артефактов предыдущих процессов.

Таким образом, структура проекта поддерживает четкое разделение этапов

сборки, что упрощает переключение между режимами отладки и релиза, а также организацию рабочих файлов. Если потребуется более детальная информация о процессах сборки или использовании Makefile,

3 РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ

```
➜  lab01 git:(main) ./build/release/prog test_dir
test_dir
test_dir/link1
test_dir/subdir2
test_dir/file1.txt
test_dir/file2.txt
test_dir/subdir1
test_dir/subdir1/link2
```

Рисунок 1 – пример работы с параметрами по умолчанию

```
➜  lab01 git:(main) ./build/release/prog test_dir -ds
test_dir
test_dir/subdir1
test_dir/subdir2
```

Рисунок 2 – пример работы с двумя параметрами

ВЫВОД

При выполнении лабораторной работы была изучена работа с файлами в операционной системе Linux.

В ходе выполнения работы была разработана программа, которая способна сканировать файлы и выводить в поток вывода информацию в соответствии с опциями программы.