СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Linux - это операционная система, которая была разработана в начале 1990-х годов Линусом Торвальдсом. Она относится к семейству Unix-подобных операционных систем и является бесплатной и открытой.

Одной из главных особенностей Linux является то, что она поставляется в виде исходного кода, который может быть изменен и распространен пользователем. Это позволяет пользователям настраивать систему под свои нужды и создавать свои собственные версии Linux.

Linux имеет множество дистрибутивов, которые отличаются друг от друга по функциональности, набору программ и методам управления. Некоторые из наиболее популярных дистрибутивов Linux включают Ubuntu, Debian, Fedora, CentOS и Arch Linux.

Дистрибути́в (distribute — распространять) — форма распространения программного обеспечения. В данном случае, форма распространения операционной системы Linux. Дистрибутив Linux состоит из ядра операционной системы и набора программ, настроенных специальным образом. Для удобства речи операционную систему Linux установленную на компьютере тоже называют дистрибутивом, чтобы было понятно какая именно разновидность Linux используется.

Дистрибутив Linux — установочный пакет для развёртывания операционной системы, состоящей из ядра Linux, утилит GNU, дополнительного ПО и диспетчера пакетов. Он также может включать в себя пакет для установки дисплейного сервера и развёртывания среды рабочего стола.

Название "дистрибутив" происходит от английского слова "distribute" — "распространять". К примеру, компании Debian и Ubuntu занимаются именно таким распространением ядра Linux со всем необходимым программным обеспечением (таким, как сетевой менеджер, диспетчер

пакетов, среда рабочего стола и т.д.) в качестве полнофункциональной операционной системы.

Дистрибутив также осуществляет установку необходимых обновлений в процессе установки, а развёрнутая на базе определённого дистрибутива ОС — в дальнейшем.

Таким образом, Linux — ядро ОС, а дистрибутив Linux — установочный пакет какой-то из разновидностей этой операционной системы плюс дополнительные компоненты. Такие разновидности (моды) называют операционными системами на базе Linux.

Linux используется в различных областях, включая серверное оборудование, настольные компьютеры, мобильные устройства и встроенные системы. Она также широко используется в области научных и инженерных вычислений, а также в разработке программного обеспечения.

Linux имеет множество преимуществ по сравнению с другими операционными системами, включая высокую стабильность, безопасность, гибкость и открытость. Это делает ее очень привлекательной для пользователей, которые хотят получить больше контроля над своей системой и настроить ее под свои нужды.

1 ОБЗОР ЛИТЕРАТУРЫ

В качестве теории и материала использовалась книга «Основы программирования в Linux» от Нэйла Мэтью.
Так же использовалась книга Forouzan B.A., Gilberg R.F. - Computer Science. A Structured Approach Using C (2006).

В каждом источнике есть полезная информация, которая поможет в написании программы, пояснения различных аспектов программирования на Linux и примеры.

1.1 Постановка задачи

Задачей является разработать утилиту, собирающую информацию о системе. Программа должна выполнять определённые действия, а именно демонстрировать пользователю информацию о ядре, памяти и процессах, дисковом пространстве, батарее, процессоре, ЧРВ, материнской плате и BIOS, дисплее, IP адресах и пользователе.

Утилита (англ. utility) - это программное обеспечение, которое выполняет специфические задачи, такие как управление файлами, диагностика и оптимизация системы, удаление вирусов и т.д. Утилиты часто используются для поддержки и оптимизации работы операционных систем и других программных приложений. Они могут быть предназначены как для профессиональных пользователей, так и для обычных пользователей компьютера.

1.2 Обзор методов и алгоритмов решения поставленной задачи

Необходимо написать программу, которая будет собирать всю необходимую информацию, а именно: информацию о ядре, памяти и процессах, дисковом пространстве, батарее, процессоре, ЧРВ, материнской

плате и BIOS, дисплее, IP адресах и пользователе. Программу можно логически поделить на несколько разделов. Каждый раздел будет написан в отдельном файле.

Каждое из перечисленных устройств имеет свои характеристики, которые и будут показаны утилитой.

В итоге необходимо реализовать две части кода: основную, с которой будет взаимодействовать пользователь и эта часть будет являться интерфейсом утилиты, и ту часть которая будет собирать информацию и передавать ее первой части.

2 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

Функциональное проектирование - это процесс разработки функциональной модели системы, которая определяет, какие функции должны выполняться системой и как они должны быть связаны между собой. Этот процесс включает в себя анализ требований, определение функций, их взаимодействие и связи, а также создание диаграмм и других моделей для представления этой информации. Функциональное проектирование является ключевым этапом в разработке любой системы, поскольку оно определяет ее основные функции и возможности.

2.1 Структура программы

Сбор информации в разрабатываемой утилите будет производиться по одному принципу для каждого устройства. Рассмотрим пример.

Код является программой, которая выводит информацию о системе, используя различные системные вызовы и библиотеки. Она начинает с проверки аргументов командной строки и вызывает функцию menu_sys_info(), если нет аргументов. В этом меню пользователь может вывести информацию, которая ему интересна.

Если при вызове программы переданы аргументы комнандной строкии, программа вызывает функцию processArguments(argc, argv), где происходит обработка аргументов. При правильном аргументе аргументе выводится соответсвующая информация. Если аргумент не правильный — выводится сообшение об неправильном аргументе. Примеры:

Если первый аргумент равен "-kernel", программа вызывает функцию switch_info('1') для получения информации о системе, такой как имя системы, узла, версии и т.д., и выводит эту информацию на экран.

Если первый аргумент равен ""-user"", программа использует функцию switch_info('2') для получения информации о пользователе, таких как имя,

индентификатор, домашнйи католог, оболочка позьзователя и выводит эту информацию на экран.

Общая структура программы состоит из проверки аргументов командной строки, вызова соответствующих функций и вывода результатов на экран. Код также включает заголовочные файлы и определения констант и переменных для работы с системными вызовами и библиотеками.

В целом, код представляет собой не сложную программу для получения информации о системе с использованием различных системных вызовов и библиотек.

2.2 Описание библиотек

Каждая из библиотек подключается для использования соответствующих функций и системных вызовов. Например, библиотека <sys/utsname.h> используется для вызова функции uname(), которая возвращает информацию о системе, включая имя системы, версию и т.д.

Библиотека <sys/utsname.h> в Linux предоставляет функции для получения информации о системе, такой как имя хоста, тип процессора, версия операционной системы и т.д. Она содержит структуру "struct utsname", которая используется для хранения этой информации.

Функция "int uname(struct utsname *buf)" является основной функцией библиотеки и используется для заполнения структуры "struct utsname". Она принимает указатель на структуру "struct utsname", которая будет заполнена информацией о системе. Функция возвращает 0 в случае успеха и -1 в случае ошибки.

Структура "struct utsname" содержит следующие поля:

- char sysname[] - имя операционной системы
- char nodename[] - имя хоста
- char release[] - версия операционной системы

- char version[] - дополнительная информация о версии операционной системы

- char machine[] - тип процессора

Библиотека <sys/sysinfo.h> используется для вызова функции sysinfo(), которая возвращает информацию о системных ресурсах.

Библиотека <sys/sysinfo.h> в Linux предоставляет функции для получения информации о системе, такой как общее количество свободной и используемой памяти, количество процессоров, загрузку системы и т.д. Она содержит структуру "struct sysinfo", которая используется для хранения этой информации.

Функция "int sysinfo(struct sysinfo *info)" является основной функцией библиотеки и используется для заполнения структуры "struct sysinfo". Она принимает указатель на структуру "struct sysinfo", которая будет заполнена информацией о системе. Функция возвращает 0 в случае успеха и -1 в случае ошибки.

Структура "struct sysinfo" содержит следующие поля:

- long uptime - время работы системы в секундах

- unsigned long loads[3] - средняя загрузка системы за последние 1, 5 и 15 минут

- unsigned long totalram - общее количество памяти в системе в байтах

- unsigned long freeram - количество свободной памяти в системе в байтах

- unsigned long sharedram - количество разделяемой памяти в системе в байтах

- unsigned long bufferram - количество буферной памяти в системе в байтах

- unsigned long totalswap - общее количество swap-памяти в системе в байтах

- unsigned long freeswap - количество свободной swap-памяти в системе в байтах

- unsigned short procs - общее количество процессов в системе

Библиотека <stdio.h> используется для вывода результатов на экран с помощью функций printf() и puts().

Таким образом, эта часть кода, используя библиотеки собирает нужную информацию о системе и предоставляет ее пользователю.

В файле Myprin.h находятся функции для вывода информации об устройстве.

В файле tools.h определены константы для работы с цветом и для рамки при выводе системной информации, а также дополнительные функции необходимые для получений какой-либо системной информации.

3 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ

Разработка программных модулей - это процесс создания отдельных компонентов программного обеспечения, которые выполняют конкретные функции в рамках более крупной системы. Каждый модуль может быть написан на разных языках программирования и может использоваться для решения различных задач, таких как обработка данных, взаимодействие с пользователем или выполнение определенных вычислений. Разработка программных модулей включает в себя проектирование архитектуры, написание кода, тестирование и интеграцию с другими компонентами системы.

В нашем случае необходимо разработать алгоритм, позволяющий собрать нужную нам информацию с устройств компьютера.

3.1 Разработка алгоритма сбора информации

Библиотеки содержат функции, которые помогут достать всю необходимую информацию. Ознакомившись с описанием библиотек составим алгоритм сбора информации, приведенный в приложении А.

Библиотека имеет функцию, которая используется для вывода определённой информации. Функция выдаёт ошибки в случае, когда информация не может быть получена.

Функции содержат множество полей, в которых и храниться интересующая нас информация. Остаётся передать эту информацию в утилиту и продемонстрировать пользователю в понятном и читаемом виде.

4 РЕЗУЛЬТАТЫ РАБОТЫ

В результате проделанной работы была создана программа, позволяющая собирать всю необходимую информацию о системе.

С помощью Sysinfo вы можете получить информацию о ядре, памяти и процессах, дисковом пространстве, батарее, процессоре, ЧРВ, материнской плате и BIOS, дисплее, IP адресах и пользователе.

Утилита имеет простой и интуитивно понятный интерфейс, который позволяет быстро получить необходимую информацию.

Данная программа подойдёт для использования рядовыми пользователями для получения информации о системе, различным сервисам, направленным на ремонт компьютерной техники или индивидуальным предпринимателям, занимающихся диагностикой и ремонтом компьютерной техники.

Так же эта программа может быть полезна для разработчиков Linux. Благодаря тому, что Linux имеет множество дистрибутивов, данную ОС можно настроить под себя максимально удобно. Помочь при разработке дистрибутива так же может помочь данная программа

Далее приведен пример работы программы.

После запуска программы без аргументов командной строки появляется главное меню (рис 5.1). В нем пользователь получает возможность использовать 10 функций утилиты.

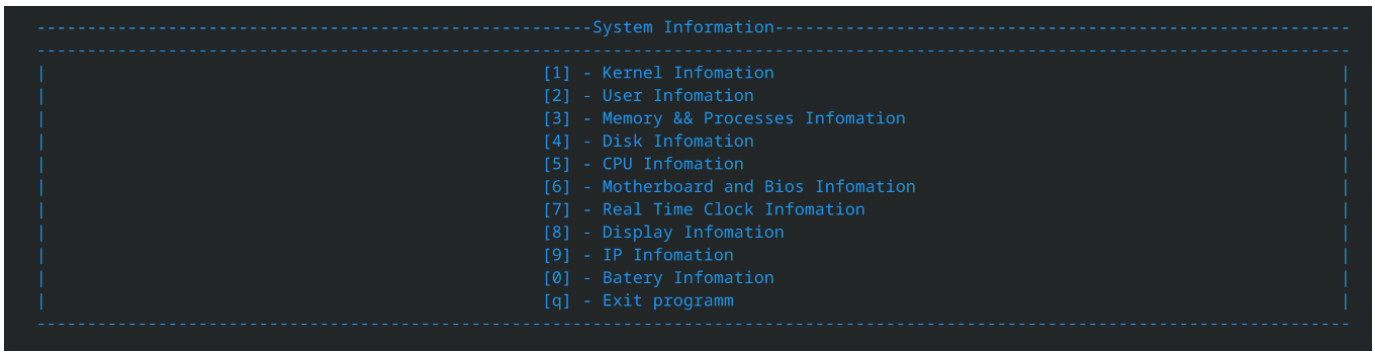

Рисунок 5.1 – Меню программы

Как показано на рисунке 5.1 пользователю предоставляется меню для выбора желаемой функции. Для выполнения функций, при запуске программы через терминал необходимо добавить в конце соответсвующий аргумент командной строки.

Рассмотрим подробнее функционал программы, предоставленный пользователю:

1) При выборе «1», пользователю будет предоставлена информация о ядре. (рис. 5.2)

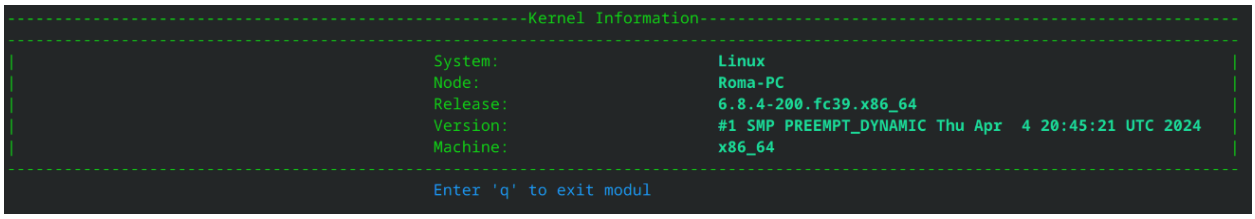

Рисунок 5.2 – Демонстрация информации о ядре

2) При выборе «2», пользователь сможет получить информацию о пользователе (рис. 5.3)

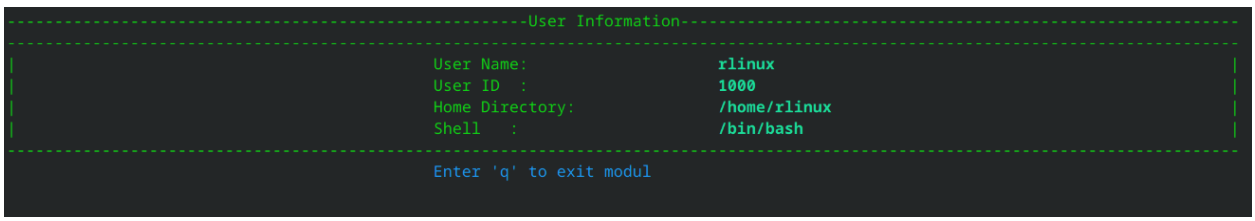

Рисунок 5.3 – Пример демонстрации информации о пользователе

3) При выборе «3» пользователь сможет получить информацию о оперативной памяти и количестве процессов. (рис.5.4)

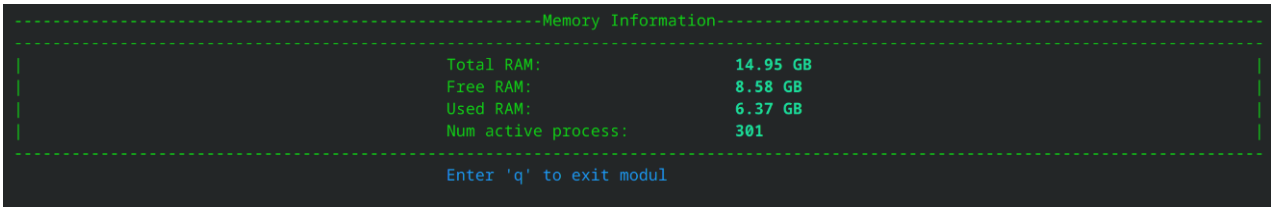

Рисунок 5.4 – пример демонстрации информации о дисковом пространстве

4) При выборе «4» пользователь сможет просмотреть информацию о жёстком диске. (рис.5.5)

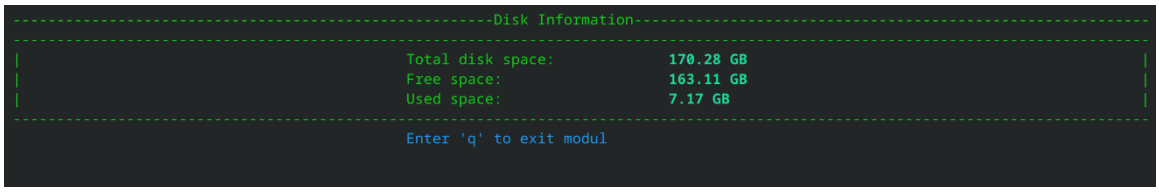

Рисунок 5.5 – пример списка установленных звуковых карт

5) При выборе «5» пользователь получит информацию о процессоре. (рис. 5.6)

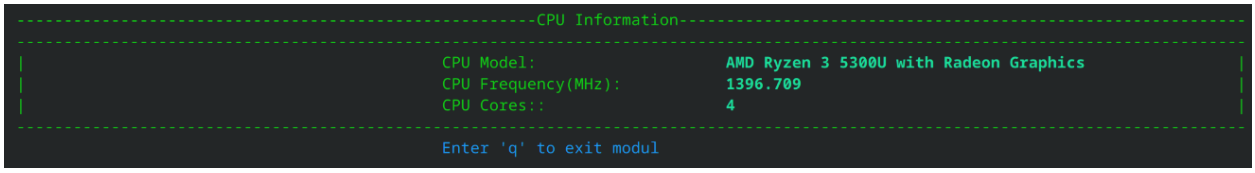

Рисунок 5.6 – пример предоставления информации о процессоре

6) При выборе «6» программа предоставит информацию о материнской плате и BIOS (рис. 5.7)

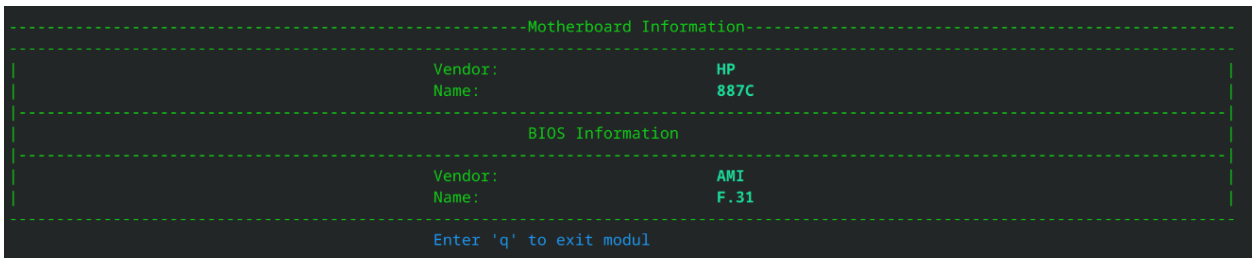

Рисунок 5.7– пример предоставленной информации о материнской плате и BIOS

7) При выборе «7» пользователь может получить информацию о часах реального времени(ЧСВ) (рис. 5.8).

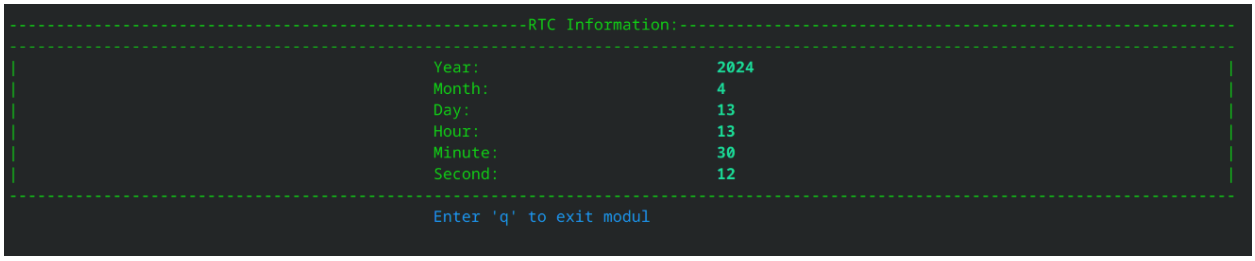

Рисунок 5.8 – пример предоставленной информации о часах реального времени(ЧСВ)

8) При выборе «8» пользователь может получить информацию о доступных ему мониторах (рис. 5.9). В случае, когда пользователь имеет несколько мониторов программа предоставит информацию о каждом из них.

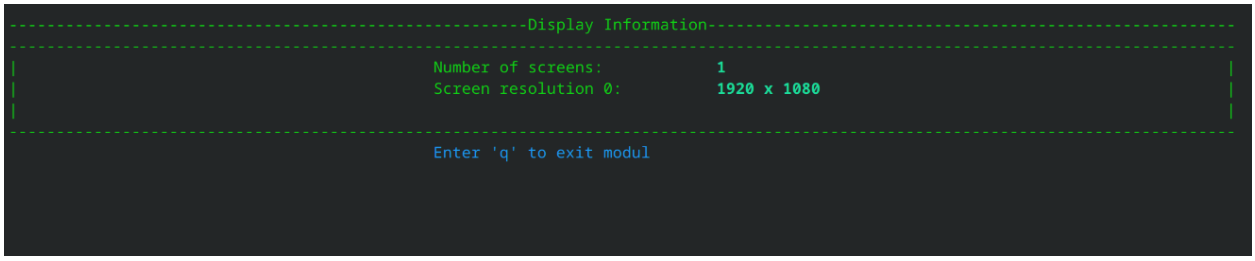

Рисунок 5.9 – пример получения информации о мониторе

9) При выборе «9» последнего пункта из меню можно получить информацию о IP машины.(рис. 5.10)

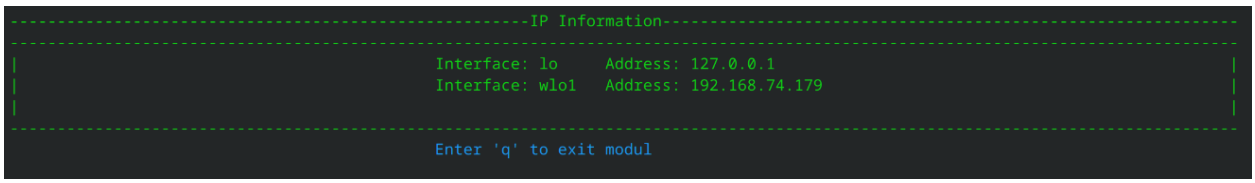

Рисунок 5.10 – пример полученной информации о IP машины

10) При выборе «0» из меню можно получить информацию о батарее. (рис. 5.11)

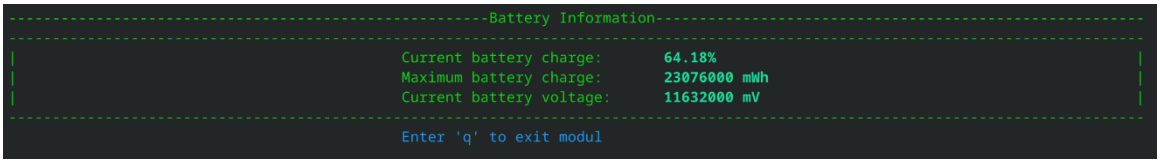

Рисунок 5.11 – пример полученной информации о батарее

ЗАКЛЮЧЕНИЕ

В заключение, Linux является мощной и гибкой операционной системой, которая широко используется в различных областях, включая программирование на языке C. Она предоставляет пользователям большую свободу и контроль над своей системой, что делает ее очень привлекательной для разработчиков. Более того, Linux имеет богатый набор инструментов и библиотек для программирования на языке C, что позволяет разработчикам создавать высококачественное программное обеспечение. В целом, использование Linux для программирования на языке C может значительно улучшить процесс разработки и повысить качество конечного продукта.

ЛИТЕРАТУРА

[1].Н. Мэтью Основы программирования в Linux. 2009г.

[2].Forouzan B.A., Gilberg R.F. - Computer Science. A Structured Approach Using C (2006).

ПРИЛОЖЕНИЕ А

Схема алгоритма

ПРИЛОЖЕНИЕ Б

Ведомость документов