МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

Институт №3 – «Системы управления, информатика и электроэнергетика»

Кафедра 307 – «Цифровые технологии и информационные системы»

Отчёт по учебной практике

«Программирование на уровне ядра Linux»

«Приложение контроля осанки и выполнения упражнений»

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы М3О-214Б-21 | «\_\_\_\_\_\_\_\_\_» 2023 г. | Подпись: \_\_\_\_\_\_\_\_\_\_\_\_ *Ситникова Виктория Андреевна* |
| Проверил руководитель, ст. преподаватель каф. 307 | «\_\_\_\_\_\_\_\_\_» 2023 г. | Подпись: \_\_\_\_\_\_\_\_\_\_\_\_ *Максимов Алексей Николаевич* |

## СПИСОК ИСПОЛНИТЕЛЕЙ

|  |  |
| --- | --- |
| Руководитель учебной практики, ст. преподаватель кафедры 307 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А. Н. Максимов |
| Исполнитель: |  |
| Студент | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В. А. Ситникова |

**ЗАДАНИЕ НА УЧЕБНУЮ ПРАКТИКУ**

студенту группы М3О-214Б-21 Ситниковой Виктории Андреевне

Дата выдачи задания «29» июня 2023 года.

Дата представления отчета «26» июля 2023 года.

Содержание задания:

1. Создать модуль ядра Linux, выдающий «Hello» при вставке в ядро, и «Bye» при удалении из него;

2. Разработать символьный драйвер с циклическим буфером, ведущий подсчет числа запусков модуля. Добавить в символьный драйвер возможность записи фразы и ее вывода. Выводить число символов в фразе;

3. К уже существующему символьному драйверу добавить блокирующие и неблокирующие операции;

4. Разработать символьный драйвер для PCI устройства, выводящий сообщение при по помощи ioctl. Создать структуру file\_operations для считывания MAC-адреса;

5. Разработать программы клиента и сервера для отправки пакетов с помощью UDP-сокетов. Провести оценку времени передачи данных. Создать deb-пакеты для приложения клиента и сервера;

6. Создать приложение для стимуляции правильной осанки. Предусмотреть возможность отслеживания выполнения физических упражнений.

Руководитель практики: Максимов А.Н.

Студент: Ситникова В. А.

ОГЛАВЛЕНИЕ

[СПИСОК ИСПОЛНИТЕЛЕЙ 2](#_Toc141173499)

[ВВЕДЕНИЕ 5](#_Toc141173500)

[ТЕХНИЧЕСКОЕ ЗАДАНИЕ 6](#_Toc141173501)

[ЦЕЛЬ 10](#_Toc141173502)

[МЕТОДЫ РЕШЕНИЯ 11](#_Toc141173503)

[ЗАКЛЮЧЕНИЕ 13](#_Toc141173504)

[СПИСОК ЛИТЕРАТУРЫ 14](#_Toc141173505)

## ВВЕДЕНИЕ

Плохая осанка приводит к раннему износу и стиранию позвоночника, дегенерации и грыже дисков. Небольшая разминка позволит запустить двигательный центр и настроиться на рабочий лад. Приложение "Nice\_Posture.арр" поможет пользователю поддерживать в тонусе мышцы и суставы, правильно распределять нагрузку по всему организму.

## ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Задание 1. Разработать модуль ядра Linux, который будет выдавать «Hello» при вставке в ядро и «By» при удалении из ядра. Вставка модуля ядра осуществляется с помощью команды sudo insmod name\_module.ko, удаление — sudo rmmod name\_module.ko. Сборка программы происходит с помощью makefile. Файл сервера отвечает за запись и чтение данных, пришедших от пользователя. Пользовательское приложение посылает серверу требуемую строку информации при входе и выходе из приложения.

Входные данные: запуск модуля ядра через командную строку. Выходные данные: строка, информирующая о запуске или удаления модуля ядра.

Задание 2. Создать символьный драйвер, который будет содержать поток ядра. В потоке ядре инкрементировать переменную счетчик. В сервере реализовать команды чтения и записи вводимой пользователем информации.

С помощью циклического буфера осуществить более удобный подход хранения потоковых данных, которые создаются в реальном времени и вскоре после этого потребляются. Новые данные могут постоянно перезанимать пространство, ранее занимаемое уже использованными данными, за счет двух индексов: чтения и записи. В случае, когда они достигают конца буфера, то должны быть преобразованы в 0. Структура file\_operations содержит указатели на определенные драйвером функции, которые выполняют различные действия с устройством. В функциях dev\_init и dev\_exit должны происходить соответственно инициализация с регистрацией и закрытие драйвера; функция dev\_open открывает драйвер; dev\_read и dev\_write соответственно производят чтение и запись информации.

Пользовательское приложение должно запрашивать у пользователя данные, что будут записаны и прочитаны символьным драйвером. Сборка сервера и пользовательского приложения осуществляется с помощью Makefile для каждой программы.

Входные данные: данные, вводимые пользователем при помощи клавиатуры. Выходные данные: вывод прочитанных и записанных в буфер сервером информации.

Задание 3. К уже существующему символьному драйверу добавить функции IOCTL для задания периода изменения переменной счетчика и блокировку пользовательской программы при последовательных чтениях, если переменная была считана, но еще не изменилась в следующий раз.

Команды IOCTL, являющиеся аппаратно-зависимыми структурами битовых полей, используются для контроля работы модуля ядра и передачи параметров устройству. Подключаются функции с помощью заголовочного файла, в котором определены макросы \_IO — указатель для безаргументной команды, и \_IOR — указатель на данные, передающиеся из ядра. Блокировка осуществляется с помощью очереди ожидания wait\_queue\_head.

Входные данные: данные, вводимые пользователем при помощи клавиатуры. Выходные данные: вывод прочитанных и записанных в буфер сервером информации.

Задание 4. Разработать символьный драйвер для PCI устройства, который возвращает через IOCTL MAC-адрес сетевой карты. Узнать при помощи ip link MAC-адрес, получить из BAR адрес области данных PCI для сетевой карты и отобразить его при помощи ioremap\_nocash в адресное пространство ядра, в функции probe драйвера вывести на печать в HEX виде содержимое области данных PCI и узнать смещение MAC адреса.

PCI драйвер должен выполнять следующие задачи в init\_module: определить структуру pci\_driver, в которой инициализируются name уникальное имя драйвера, id\_table — таблица пар Vendor ID и Product ID, probe — функция вызываемая ядром после загрузки драйвера, которая служит для инициализации оборудования, remove — функция вызываемая ядром при выгрузке драйвера, служит для освобождения каких-либо ранее занятых ресурсов. Выходные данные: вывод MAC адреса с помощью IOCTL команд.

Задание 5. Разработать программы клиента и сервера для отправки пакетов с помощью UDP-сокетов, провести оценку времени передачи данных, собрать программы в deb-пакеты.

Программа клиента засекает время отправление запроса и получения ответа. Программа сервера должна получать запрос от клиента, читать значение из ранее созданного символьного драйвера и отправлять в ответ клиенту сообщение. Для программы сервера необходимо создать сервис, а также, для обоих программ, требуется разработать CMakeFilies для сборки. В программе сервера находится адрес сокета, используемый для хранения адреса клиента, считывается содержимое буфера от входящего клиента и отправляется тот же контент обратно пользователю. Программа клиента должна открывать и закрывать сокеты для передачи информации, записывать данные в буфер, засекать время передачи и передавать информацию серверу, получать ответ от сервера и отмечать время окончания соединения.

Сборка deb-пакетов для клиентского приложения и сервера осуществляется с помощью команды cpack.. .

Входные данные: данные, вводимые пользователем с клавиатуры. Выходные данные: количество времени, затраченное на отправление информации от клиента к серверу и обратно.

Задание 6. Написать программу для мониторинга осанки пользователя. Предусмотреть возможность отслеживания количества упражнений, выполненных пользователем в качестве разминки.

Программа должна проверять правильность осанки клиента и в случае выявления отклонений от нормы посылать звуковое уведомление.

При запуске программы на экран пользователя выводится окно с выбором: отслеживания осанки или выполнения упражнения. При нажатии на кнопку «Отслеживание осанки», пользователь переходит в следующие окно, в котором строится скелетная модель человека и проверяется правильность расположения спины, рук, ног и шеи в позиции сидя. В случае отклонения, транслируется звуковой файл с предупреждением. Пользователь может вернуться к начальному окну выбора через нажатие клавиши «esc».

При выборе «Выполнение упражнения» на экране клиента появляется окно с выбором физической активности: «Приседания», «Махи руками», «Наклоны». Клиент может вернуться на начальное окно с помощью нажатия на кнопку «Назад». После выбора упражнения пользователю предоставляется возможность провести разминку без или с записью видеофайла. На экране пользователя появляется окно с трансляцией видеопотока камеры со скелетной моделью человека, а также выводится счет выполненных упражнений с одним из двух возможных состояний критерия расположения рабочих частей тела индивида «Down» или «Up». Клиент может вернутся к выбору упражнения, нажав на клавиатуре «esc». Видеоклип разминки хранится в каталоге с проектным файлом.

Для выхода из программы можно нажать либо на «esc», либо на закрывающую кнопку окна — крестик.

## ЦЕЛЬ

Первая часть практики задания ориентирована на ознакомление с системным программированием в Linux, то есть на написание системного ПО, низкоуровневый код которого взаимодействует напрямую с ядром и основными системными библиотеками. Основная цель 1-5 задний — введение в программирование ядра, подразумевающее изменение существующих модулей ядра или добавление новых модулей в ядро Linux.

Вторая часть практики посвящена разработке приложения, главной задачей которого является отслеживание правильной осанки, и предупреждение пользователя, в случае отклонения его положения от нормы. В качестве расширения функциональных возможностей программы необходимо предусмотреть наличие алгоритма распознавания физических упражнений.

## МЕТОДЫ РЕШЕНИЯ

Решение 1-й части задания представлено по ссылке – URL: <https://github.com/VASitnikova03/practice/tree/main/ядро%20linnux>

Программы написаны на языки Си и реализованы на основе операционной системы Ubuntu дистрибутива Linux. Являются модулями ядра, то есть элементами кода, который можно загружать в ядро и выгружать для расширения его функциональности без необходимости перезагрузки системы.

Решение 2-й части задания представлено по ссылке – URL: <https://github.com/VASitnikova03/practice/tree/main/osanka>

Программа «контроля осанки и выполнения физических упражнений» написана в среде PyCharm Community Edition 2022.2.3 с использованием языка Python. Операционная система Windows.

Программа использует библиотеки: mediapipe, cv2, os, sys, PySide6, numpy, keyboard. Для удобства реализации поставленных задач функции распознавания правильности осанки и подсчета различных физических упражнений реализуются в отдельных файлах, что упрощает читабельность программы и ее отладку.

Файл main.py отвечает за запуск пользовательского приложения. Он содержит класс App(QMainWindow), отвечающий за создание экземпляра пользовательского интерфейса, который запускается методами:

app = QApplication(sys.argv)  
window = App()  
window.show()

Файлы ui\_main, choice.py, camera(1,2).py представляют собой реализацию пользовательского интерфейса, созданного с помощью программы Qt Designer, основной функцией которого является упрощение доступа пользователей к возможностям разработанной программы.

Алгоритмы оценки правильности осанки, а также подсчет выполняемых упражнений реализуются в файлах pose.py, prised.py, mahi.py, naklon.py, основой которых являются библиотеки mediapipe — фреймворк для создания кросс-платформенных мультимодальных конвейеров машинного обучения, и cv2 — библиотека компьютерного зрения, которая предназначена для анализа, классификации и обработки изображений.

Основные методы, используемые в этих файлах:

cap = cv2.VideoCapture(1) — отвечает за импорт видеопотока;

landmarks = results.pose\_landmarks.landmark — получение опорных точек из библиотеки mediapipe;

mp\_drawing.draw\_landmarks() — отрисовка линий, соединяющих опорные точки позы человека;

cv2.imshow(**'Mediapipe Feed'**, image) — вывод всплывающего окна с кадром.

## ЗАКЛЮЧЕНИЕ

По итогам прохождения учебной практики проведено ознакомление с модулями ядра Linux — на языке С написаны собственные объектные файлы, расширяющие функциональность базового ядра ОС. Также, на основе библиотек mediapipe и cv2, на языке Python разработан алгоритм определения правильной осанки человека, лежащий в основе созданной программы “Nice\_Posture.арр”, предоставляющей также возможность выполнения упражнений с контролем числа их выполнения. Отчет по учебной практике составлен в соответствии с ГОСТ 7.32-2017 в программе Microsoft Word.

## СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 7.32-2017 [Электронный ресурс]. ГОСТ 7.32-2017 СИБИД. Отчет о научно-исследовательской работе. ГОСТ от 24 октября 2017 года №7.32-2017. — URL: <http://docs.cntd.ru/document/1200157208> Дата обращения: 25.07.2023
2. Пособие по программированию модулей ядра Linux [Электронный ресурс] — URL: <https://habr.com/ru/companies/ruvds/articles/681880/> Дата обращения: 25.07.2023
3. Building a Poor Body Posture Detection & Alert System Using MediaPipe Body Tracking [Электронный ресурс] — URL: <https://learnopencv.com/building-a-body-posture-analysis-system-using-mediapipe/> Дата обращения: 25.07.2023