Министерство образования Новосибирской области

ГБПОУ НСО «Новосибирский авиационный технический колледж имени Б.С. Галущака»

Отчёт по производственной практике

по ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

НАТКиГ.721800.010.000

Разработал:

студент группы ПР-21.101

Портнов М.А.

**Содержание**

[Введение 3](#_Toc170807383)

[1 Описание предметной области 5](#_Toc170807384)

[2 Проектирование информационной системы 6](#_Toc170807385)

[2.1 Постановка задачи 6](#_Toc170807386)

[2.2 Диаграмма прецедентов и спецификация 6](#_Toc170807387)

[2.3 Макет приложения 8](#_Toc170807388)

[3 Разработка приложения 10](#_Toc170807389)

[3.1 Описание используемых технологий 10](#_Toc170807390)

[3.2 Описание разработанных процедур и функций 10](#_Toc170807391)

[4 Тестирование модулей для приложения 12](#_Toc170807392)

[4.1 Выбор стратегии тестирования 12](#_Toc170807393)

[4.2 Протоколы тестирования 12](#_Toc170807394)

[Заключение 13](#_Toc170807395)

[Библиография 14](#_Toc170807396)

Введение

Школа 21 − это международная школа программирования, основанная в Париже, Франция, в 2013 году французским предпринимателем и инноватором Ксавье Нилом. Основная концепция Школы 21 заключается в том, что она предлагает образование по принципам пирамиды без преподавателей и занятий, полностью ориентированное на самостоятельное обучение и коллаборацию между студентами.

В Школе 21 вместо традиционного обучения лекциями и семинарами, студенты самостоятельно изучают материалы и решают задачи в своем темпе. Обучение осуществляется через проектную работу и коллаборацию между студентами. Один студент может помогать другому, что способствует обмену знаниями и совместному решению проблем. Обучение в Школе 21 построено на выполнении реальных проектов, которые оцениваются на основе их успешности и качества реализации. А также программа обучения включает в себя не только программирование, но и другие аспекты, включая дизайн, разработку интерфейсов и управление проектами.

В рамках прохождения практики в Школе 21 была поставлена цель − познакомиться с базовыми утилитами Bash по работе с текстами на языке программирования С и разработать утилиты Cat и Grep, используя системные вызовы для работы с файлами.

Задачами производственной практики являются:

* чтение файлов, обработка данных в соответствии с переданными флагами;
* обработка специальных символов и флагов, в «Cat» − замена специальных символов, в «Grep» − вывод строк, соответствующих заданному регулярному выражению;
* компиляция и сборка через Makefile. Создание и настройка Makefile для компиляции программ.
* тестирование и отладка, проверка корректности работы утилиты через тестовые файлы и различные сценарии использования.

Необходимые требования для выполнения задания:

* в процессе разработки утилиты необходимо учитывать эффективное использование памяти и оптимизацию производительности, особенно при работе с большими файлами или потоками данных;
* навыки обработки и интерпретации аргументов, передаваемых через командную строку, таких как флаги и имена файлов;
* в случае утилиты «Grep» необходимо иметь знания о регулярных выражениях для поиска и фильтрации строк в файле в соответствии с заданным шаблоном;
* для работы с утилитой Cat нужно знать кодировка **ASCII.**

# Описание предметной области

«Школа 21» – школа цифровых технологий от Сбера и 2Гис. Основная концепция Школы 21 заключается в том, что она предлагает образование по принципам пирамиды без преподавателей и занятий, полностью ориентированное на самостоятельное обучение и коллаборацию между студентами.

Практическое задание направлено на развитие навыков системного программирования и работы с командной строкой в среде Linux, используя язык программирования C. Основная цель заключается в создании двух утилит: «Cat» и «Grep», каждая из которых должна реализовать определённый набор функций и поддерживать соответствующие флаги, согласно стандартам и расширениям GNU.

Задание не только предоставляет возможность практической работы с системным программированием на языке C, но и развивает ключевые навыки работы с файлами, аргументами командной строки и регулярными выражениями. Оно направлено на достижение глубокого понимания основ системных утилит и их реализации, что является важным аспектом для разработчиков, работающих в области операционных систем и прикладного программирования под Linux.

# Проектирование информационной системы

## Постановка задачи

Цель производственной практики – разработать утилиты «Cat» и «Grep» на языке С, используя системные вызовы для работы с файлами. Для достижения поставленной цели необходимо решить следующие задачи:

* для разработки утилиты «Cat» необходимо реализовать функционал для вывода содержимого файлов в стандартный вывод. Должна присутствовать поддержка всех стандартных флагов, таких как -b, -n, -E, -s и других;
* для разработки утилиты «Grep» выполнить реализацию функционала поиска строк в файле с использованием регулярных выражений с помощью библиотеки PCRE или regex. Должна присутствовать поддержка флагов -e, -i, -v, -c, -l, -n для выполнения различных операций с найденными строками.

## Диаграмма прецедентов и спецификация

На рисунке 1 представлена диаграмма прецедентов.

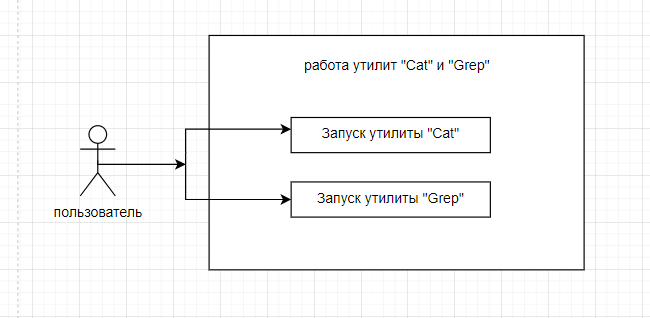


Рисунок 1 − Диаграмма прецедентов

Пользователь может запустить утилиту «Cat», в таблице 1 предоставлена спецификация для прецедента «Запуск утилиты «Cat»».

Таблица 1 – Спецификация для прецедента «Запуск утилиты «Cat»».

|  |  |
| --- | --- |
| Раздел | Описание |
| Краткое описание | Пользователь запускает утилиту «Cat» |
| Действующие лица | Пользователь |
| Предусловия | Пользователь имеет доступ к файлам, которые планирует использовать в утилите Cat |
| Основной поток | 1. Пользователь запускает утилиту Cat с флагами и файлами в качестве аргументов   А1. Неверно введены аргументы   1. Утилита открывает каждый указанный файл с помощью системного вызова 2. Утилита последовательно читает содержимое файлов с помощью системного вызова и выводит в зависимости от флагов в стандартный вывод с помощью системного вызова 3. По завершении чтения файлов, утилита Cat закрывает файловый дескриптор с помощью системного вызова 4. Утилита Cat завершает работу и прецедент завершается |
| Альтернативный поток | А1. Неверно введены аргументы   1. Система выводит сообщение «Некорректные данные», т.е. был неправильно определен путь к файлу, или введен несуществующий флаг 2. Программа завершается |
| Постусловие | Результатом будет вывод содержимого указанных файлов в стандартный вывод |

Для запуска утилиты «Grep» используются те же действия. Отличие заключается в том, что в «Сat» обработка файла идет посимвольно, а в «Grep» построчно. В таблице 2 предоставлена спецификация для прецедента «Запуск утилиты «Grep»».

Таблица 2 – Спецификация для прецедента «Запуск утилиты «Grep»»

|  |  |
| --- | --- |
| Раздел | Описание |
| Краткое описание | Пользователь запускает утилиту «Grep» |
| Действующие лица | Пользователь |
| Предусловия | Пользователь имеет доступ к файлам, которые планирует использовать в утилите Grep |

Разрыв таблицы 2

|  |  |
| --- | --- |
| Основной поток | 1. Пользователь запускает утилиту Grep с флагами и файлами в качестве аргументов   А1. Неверно введены аргументы   1. Утилита открывает каждый указанный файл с помощью системного вызова 2. Утилита последовательно читает содержимое файлов с помощью системного вызова и выводит в зависимости от флагов в стандартный вывод с помощью системного вызова 3. каждая строка проверяется на наличие регулярного выражения, если строка его содержит, то она выводит ее иначе нет 4. По завершении чтения файлов, утилита Grep закрывает файловый дескриптор с помощью системного вызова 5. Утилита завершает работу и прецедент завершается |
| Альтернативный поток | А1. Неверно введены аргументы   1. Система выводит сообщение «Некорректные данные», т.е. был неправильно определен путь к файлу, или введен несуществующий флаг 2. Программа завершается |
| Постусловие | Результатом будет вывод содержимого файлов в стандартный вывод и вывод регулярных выражений, если такие есть |

## Макет приложения

В представлении общей структуры и основных компонентов разрабатываемого консольного приложения "Cat" и "Grep" представлены ключевые аспекты функциональности и взаимодействия компонентов, необходимые для полного понимания архитектуры и работы приложения. Утилита «Cat» представлена на рисунке 2. Утилита «Grep» представлена на рисунке 3.

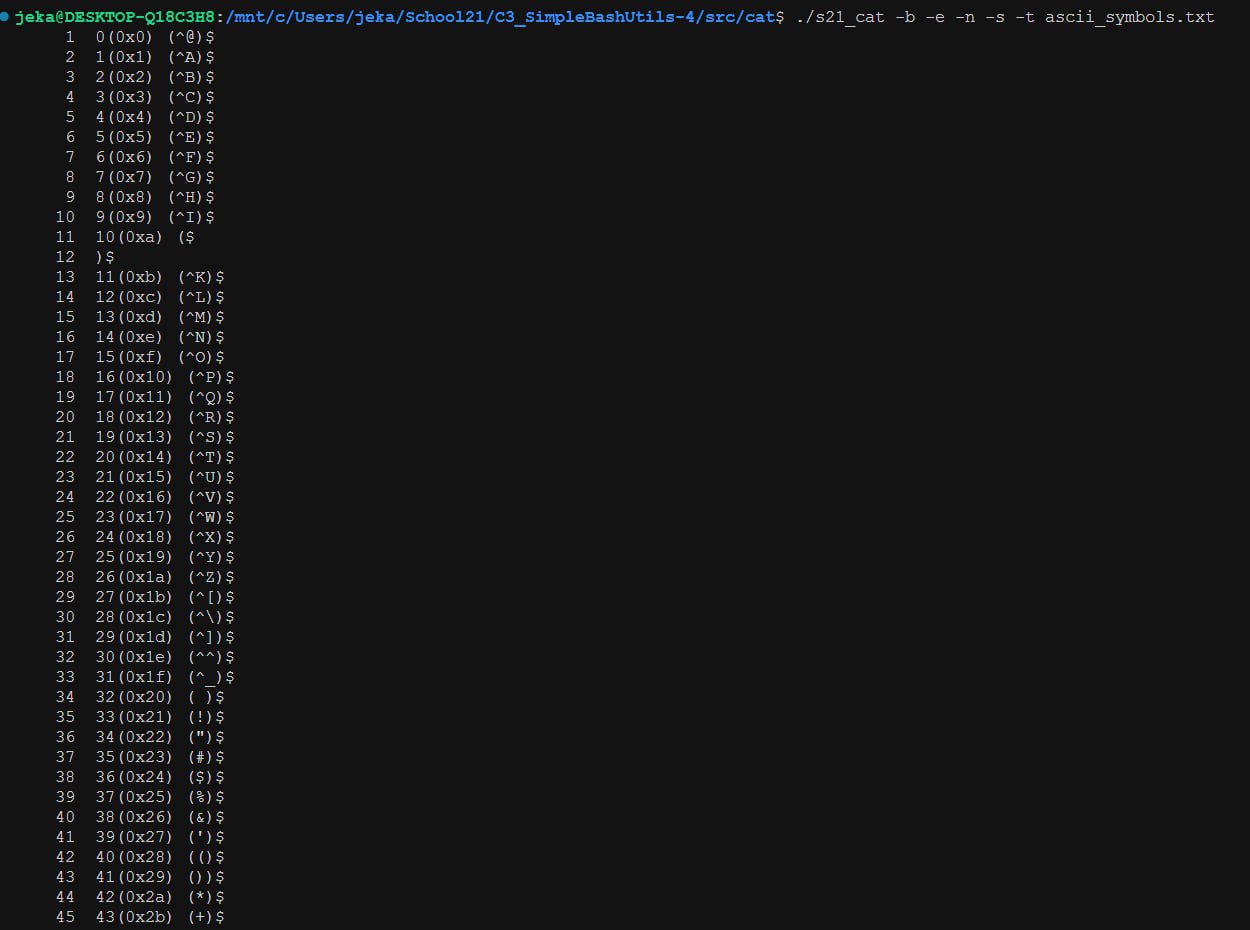


Рисунок 2 – Утилита «Cat»

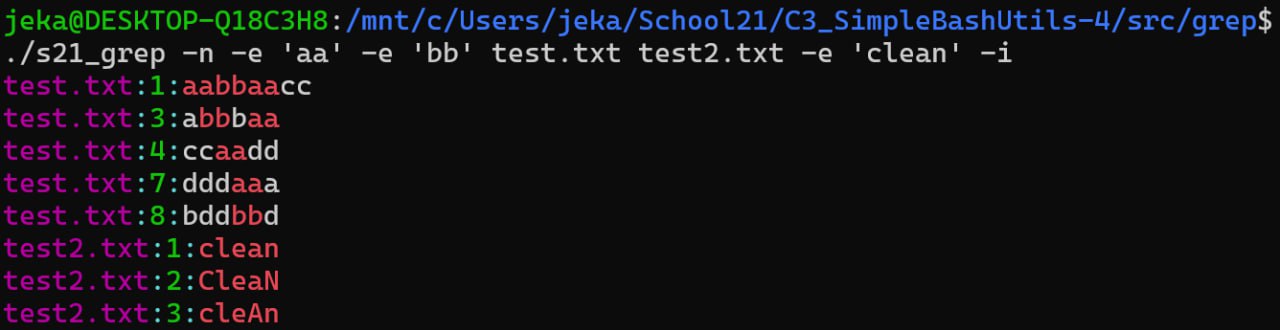


Рисунок 3 – Утилита «Grep»

# Разработка приложения

## Описание используемых технологий

Для разработки утилит «Cat» и «Grep» на языке C были использованы различные технологии и средства, позволяющие эффективно работать с файлами и реализовывать необходимую функциональность. В таблице 3 представлены используемые в проекте библиотеки.

Таблица 3 – Библиотеки

|  |  |
| --- | --- |
| Название библиотеки | Описание |
| stdbool.h | Определяет макросы для использования логических значений true и false в языке С |
| stdio.h | Предоставляет функции для стандартного ввода и вывода, такие как чтение и запись файлов, ввод с клавиатуры и вывод на экран |
| stdlib.h | Содержит функции для управления памятью, преобразования чисел и других стандартных функций языка С |
| getopt.h | Обеспечивает функции для разбора аргументов командной строки, включая обработку флагов и их аргументов |
| ctype.h | Содержит функции для работы с символами, такие как проверка на числовой или буквенный символ, преобразование регистра символов и другие операции |
| regex.h | Предоставляет функции для работы с регулярными выражениями |
| string.h | Содержит функции для работы со строками |

## Описание разработанных процедур и функций

Обработка символов «Cat». На вход подается символ и аргументы командной строки, на выходе идет вывод в консоль символа. В зависимости от флагов, программа распознает, нужно ли этот символ обрабатывать.

void processSymbol(char symbol, const Options\* const opt) {

if (symbol == '\n') {

fputc(symbol, stdout);

} else if (symbol == '\t') {

if (opt->t) {

printCtrlSymbol(symbol);

} else {

printf("%c", symbol);

}

}

else if (((unsigned char)symbol) > DEL\_ASCII) {

if (opt->v) {

const char meta\_symbol = (signed char)symbol + DEL\_ASCII + 1;

printf("M-");

if (meta\_symbol <= LAST\_SPECIAL\_SYMBOL || meta\_symbol == DEL\_ASCII) {

printCtrlSymbol(meta\_symbol);

} else {

fputc(meta\_symbol, stdout);

}

} else {

fputc(symbol, stdout);

}

} else if (opt->v && (symbol <= LAST\_SPECIAL\_SYMBOL || symbol == DEL\_ASCII)) {

printCtrlSymbol(symbol);

} else {

if (!(opt->e && symbol == '\r')) {

fputc(symbol, stdout);

}

}

}

# Тестирование модулей для приложения

## Выбор стратегии тестирования

При тестировании системы было использовано:

* тестирование функциональности.

## Протоколы тестирования

Необходимо проверить функционал приложения. Для этого, для функции был разработан TestCase.

Для тестирования приложения был написан скрипт на Python, который компилирует программу, перемещает в папку, где создает файлы со случайным набором символов, и запускает данный «Cat» с оригинальным со всеми возможными комбинациями флагов и сравнивает их вывод.

В таблице 4 предоставлено тестирование утилиты «Cat».

Таблица 4 – Тестирование утилиты «Cat»

|  |  |  |
| --- | --- | --- |
| Название | Test-1 | |
| Функция | Утилита «Cat». | |
| Предусловие | Запустить приложение | |
| Действие | Ожидаемый результат | Результат теста |
| Шаги теста(positive)  1. Запустить скрипт на python, который компилирует программу и сравнивает вывод | Окно с выводом информации открылось | пройден |
| Шаги теста(negative)  1. Запустить скрипт на python, который компилирует программу и сравнивает вывод | Окно с выводом информации не открылось | пройден |

Для тестирования утилиты «Grep» производятся те же действия.

Заключение

В ходе проделанной работы были созданы утилиты «Cat» и «Grep» на языке С, используя системные вызовы для работы с файлами.

В ходе производственной практики выполнены все поставленные задачи, а именно:

* для разработки утилиты «Cat» был реализован функционал для вывода содержимого файлов в стандартный вывод. Присутствует поддержка всех стандартных флагов, таких как -b, -n, -E, -s и других;
* для разработки утилиты «Grep» была выполнена реализация функционала поиска строк в файле с использованием регулярных выражений с помощью библиотеки PCRE или regex. Присутствует поддержка флагов -e, -i, -v, -c, -l, -n для выполнения различных операций с найденными строками.

Была проведена работа с указателями, управление памятью и операции ввода-вывода. Приобретены навыки обработки флагов и параметров, передаваемых через командную строку при запуске программы, а также работа с регулярными выражениями и их применение с помощью библиотек PCRE или regex.

В итоге проделанной работы приобретены новые умения, которые помогли приобрести больше опыта и знаний для дальнейшего совершенствования в области программирования.

Библиография

1. TimeWeb [электронный ресурс] – утилита «Cat» Linux [https://timeweb.com/ru/community/articles/komanda-Cat-v-linux](https://timeweb.com/ru/community/articles/komanda-cat-v-linux) (дата последнего обращения 11.06.24).
2. Losst [электронный ресурс] – утилита «Grep» – <https://losst.pro/gerp-poisk-vnutri-fajlov-v-linux> (дата последнего обращения 16.06.24).