Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Операционные среды и системное программирование

ОТЧЁТ

к лабораторной работе №2

на тему

**ОБРАБОТКА ТЕКСТОВОЙ ИНФОРМАЦИИ. РЕГУЛЯРНЫЕ ВЫРАЖЕНИЯ.**

Выполнил: студент гр.253504 Сапроненко В.В.

Проверил: ассистент кафедры информатики Гриценко Н.Ю.

Минск 2025

**СОДЕРЖАНИЕ**

[1 Формулировка задачи 3](#_Toc183307053)

[2 Краткие теоритические сведения 4](#_Toc183307054)

[3 Описание функций программы 5](#_Toc183307055)

[Заключение 6](#_Toc183307056)

[Список использованных источников 7](#_Toc183307057)

[Приложение А (обязательное) исходный код программы 8](#_Toc183307058)

# **1 ФОРМУЛИРОВКА ЗАДАЧИ**

В данной лабораторной работе необходимо изучить методов и средств обработки текстовой информации, включая регулярные выражения, а также освоение утилит sed, awk и других инструментов Unix, используемых в shell-скриптах. Целью работы является не только создание конкретного скрипта, но и понимание того, как различные команды и утилиты могут быть объединены в единое решение для автоматизации обработки текста.

В рамках лабораторной работы изучаются основные элементы и конструкции shell-скриптов на языке bash, включая работу с переменными, аргументами командной строки, выполнение текстовых преобразований с использованием регулярных выражений и применение утилит обработки текста, таких как sed. Особое внимание уделяется принципам работы с многострочными данными, обработке исключительных ситуаций и корректной интерпретации входных данных.

Необходимо написать скрипт на языке zsh с использованием утилиты sed для обработки текстовых файлов в соответствии с заданным вариантом. Скрипт должен выполнять функцию «автокорректора», заменяя строчные буквы на заглавные в начале предложений. Предложение определяется как текст, начинающийся:

При этом начало предложения может находиться на новой строке, а предыдущий знак препинания — на предыдущей строке. Скрипт принимает входной файл как аргумент командной строки и выводит обработанный текст в консоль с возможностью перенаправления результата в другой файл.

Требования к реализации:

1 Использовать утилиту sed для выполнения замен с применением регулярных выражений.

2 Обеспечить обработку многострочных данных, включая переносы строк между знаками препинания и началом следующего предложения.

3 Предусмотреть корректное поведение скрипта при ошибочных или "неочищенных" входных данных (например, пустой файл или отсутствие аргумента).

4 Реализация должна демонстрировать понимание работы с потоками ввода-вывода, конвейерами и базовыми конструкциями zsh, такими как чтение файла в переменную и передача данных в утилиты обработки текста.

Разработанный скрипт позволяет автоматизировать задачу форматирования текста, закрепляя навыки работы с регулярными выражениями, утилитой sed и основами программирования в zsh. В процессе выполнения работы углубляется понимание принципов создания гибких и эффективных скриптов для обработки текстовой информации в среде Unix.

# **2 КРАТКИЕ ТЕОРИТИЧЕСКИЕ СВЕДЕНИЯ**

*Shell*-*скрипты* представляют собой мощный инструмент для автоматизации задач в *Unix/Linux*, позволяющий объединять команды оболочки и утилиты в последовательности для обработки данных и управления системой. Они дают возможность пользователю выполнять сложные операции с текстом, файлами и потоками данных, минимизируя ручной ввод команд. Основу скриптов составляют переменные, аргументы командной строки, управляющие конструкции, вызовы утилит и перенаправление ввода-вывода [1].

Переменные в shell служат для хранения информации и могут быть определены пользователем или предоставлены системой. Например, переменная $*PWD* содержит путь к текущей директории, а $1 предоставляет доступ к первому аргументу, переданному скрипту. Управление логикой выполнения осуществляется через конструкции ветвления, такие как *if-else*, которые позволяют обрабатывать условия, например, проверять наличие входного файла. Для многократного выполнения операций применяются циклы, такие как *for* или *while*, которые полезны при обработке нескольких файлов или строк.

Ключевую роль в обработке текста играют утилиты, такие как *sed, awk* и grep, которые вызываются из скрипта для выполнения специализированных задач. Например, *sed* используется для потокового редактирования текста с применением регулярных выражений, позволяя заменять, удалять или форматировать данные. Встроенные команды оболочки, такие как echo или cat, обеспечивают базовую работу с вводом-выводом, а арифметические операции выполняются через expr или $(( )), хотя в задачах обработки текста они применяются реже.

Перенаправление ввода-вывода и конвейеры являются важными механизмами для управления потоками данных. Символ > перенаправляет вывод в файл, перезаписывая его, а >> добавляет данные в конец файла. Конвейер | позволяет передавать результат одной команды в другую, например, из cat в sed, что упрощает обработку текста в реальном времени. Для создания и редактирования скриптов используются текстовые редакторы, такие как vim, nano или micro, которые обеспечивают удобную работу в терминале. В данной лабораторной работе требуется разработать zsh-скрипт с использованием sed для преобразования текста, заменяя строчные буквы на заглавные в начале предложений, определяемых по заданным правилам. Результаты обработки должны быть выведены в консоль с возможностью сохранения в файл.

Таким образом, изучение shell-скриптов, включая переменные, аргументы, управляющие конструкции, утилиты обработки текста и механизмы перенаправления, позволяет создавать эффективные решения для автоматизации задач в Unix-подобных системах, таких как форматирование текста и работа с файлами.

# **3 ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ**

Программа представляет собой zsh-скрипт, предназначенный для обработки текстовых файлов с функцией «автокорректора», заменяющего строчные буквы на заглавные в начале предложений.

Основная функциональность:

1 Скрипт реализует единый режим обработки текста

2 Замена строчных букв на заглавные выполняется в начале файла и после знаков препинания (., !, ?), за которыми следуют пробельные символы, включая переносы строк. Для этого используется sed с двумя выражениями: одно для начала текста, другое — для текста после знаков препинания. Содержимое файла считывается с помощью cat и передаётся в sed через конвейер. Результат выводится в терминал с возможностью перенаправления в файл.

Скрипт предусматривает базовую обработку ошибок: при отсутствии входного файла или некорректных данных обработка завершается без вывода. Отладочные возможности ограничены стандартным поведением sed, но могут быть расширены добавлением сообщений о входном файле через echo. Работа программы продемонстрирована на рисунке 3.1.

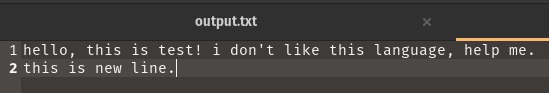


Рисунок 3.1 – Пример текста

Аргументы обрабатываются последовательно, и скрипт ожидает единственный параметр — имя файла. Для обработки текста используется sed, поддерживающий регулярные выражения и многострочные данные.

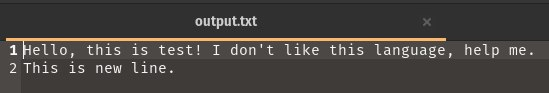


Рисунок 3.2 – Результат обработки многострочного текста

Таким образом, программа демонстрирует использование базовых элементов bash-программирования, а также иллюстрирует интеграцию этих элементов для создания простого инструмента форматирования текста.

# **ЗАКЛЮЧЕНИЕ**

Программа представляет собой shell-скрипт, предназначенный для поиска файлов в дереве каталогов с выполнением заданных действий над найденными файлами. Скрипт реализует функциональность, аналогичную утилите find, но с дополнительной обработкой содержимого файлов на основе их заголовков. Для реализации функциональности используются несколько вспомогательных функций и механизмов обработки аргументов командной строки.

Программа представляет собой bash-скрипт, разработанный для автоматизации обработки текстовых файлов с функцией «автокорректора», заменяющего строчные буквы на заглавные в начале предложений. Скрипт использует утилиту sed для выполнения преобразований текста на основе регулярных выражений, обрабатывая входной файл, переданный через аргумент командной строки.

Основная логика работы программы заключается в чтении содержимого файла с помощью команды cat, передаче текста в sed через конвейер и применении двух выражений замены: для начала файла и после знаков препинания (., !, ?). Обработанный текст выводится в терминал, предоставляя пользователю возможность перенаправить результат в файл.

Таким образом, данный скрипт демонстрирует использование базовых элементов программирования в zsh, таких как переменные, конвейеры и вызовы внешних утилит. Программа иллюстрирует, как эти элементы могут быть интегрированы для создания простого и эффективного инструмента форматирования текста в консольной среде Unix.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] What is "the Shell"? [Электронный ресурс]. – Режим доступа: https://linuxcommand.org/lc3\_lts0010.php

[2] Introduction to Shell Scripting Using Zsh [Электронный ресурс]. – Режим доступа: https://hcsonline.com/images/PDFs/Scripting\_Intro\_Zsh.pdf

# **ПРИЛОЖЕНИЕ А (обязательное) Исходный код программного продукта**

#!/bin/sed -Ez

s/^( \*\n\*|[[:space:]]\*)([a-z])/\1\u\2/

s/([^0-9])([.!?])([[:space:]]+)([a-z])/\1\2\3\u\4/g