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 №3 на тему

ОСНОВЫ ПРОГРАММИРОВАНИЯ НА *С* ПОД *UNIX*. ИНСТРУМЕНТАРИЙ ПРОГРАММИСТА В *UNIX*

Выполнил: студент гр. 253503 Тимошевич К.С.

Проверил: ассистент кафедры информатики Гриценко Н.Ю.

СОДЕРЖАНИЕ

1 Постановка задачи	. 3
2 Описание работы программы	
2.1 Обработка входных данных	
2.2 Инверсия строк	
2.3 Makefile и управление сборкой	
3 Ход выполнения программы	
3.1 Примеры выполнения задания	
Вывод	
Список использованных источников.	
Приложение А (справочное) Исходный код	

1 ПОСТАНОВКА ЗАДАЧИ

Целью данной лабораторной работы является изучение инструментов программирования в среде *Unix/Linux*, включая компилятор *GCC*, систему управления сборкой *Make*, работу с библиотеками и системными вызовами [1]. В ходе работы была разработана программа-фильтр, выполняющая инверсию порядка символов в каждой строке потока данных без изменения порядка строк. Программа обрабатывает входные данные, поступающие либо из стандартного ввода, либо из файла, и выводит результат в стандартный вывод или в указанный пользователем файл. Длина строк ограничена некоторой достаточно большой константой.

Программа реализована с использованием многомодульной архитектуры, где головной модуль отвечает за обработку аргументов командной строки, чтение и запись данных, а рабочий модуль содержит функцию инверсии строки. Заголовочный файл определяет используемые функции. Для удобства сборки и управления проектом был создан *Makefile*, включающий цели для компиляции программы, очистки временных файлов и тестирования работоспособности программы на заранее подготовленных входных данных [2].

2 ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ

Данный раздел описывает работу программы, выполняющей инверсию символов в строках. Рассмотрены этапы обработки входных данных, алгоритм инверсии строк и запись результата в выходной поток или файл.

2.1 Обработка входных данных

Программа принимает на вход один или два аргумента командной строки. Первый аргумент (необязательный) указывает путь к входному файлу, второй аргумент (также необязательный) задает путь к выходному файлу. Если аргументы отсутствуют, программа считывает данные из стандартного ввода и выводит результат в стандартный вывод.

Перед обработкой входных данных проверяется корректность переданных аргументов. Если указан входной файл, но он не существует или не может быть открыт, программа выводит сообщение об ошибке и завершает выполнение. Аналогично, если указан выходной файл, но он недоступен для записи, программа также сообщает об ошибке и завершает работу.

2.2 Инверсия строк

После успешного открытия входного потока начинается построчное чтение данных. Каждая строка считывается в буфер, длина которого ограничена константой MAX_LINE_LENGTH . Затем вызывается функция reverse_line, которая выполняет инверсию символов в строке, не изменяя порядок самих строк.

Функция *reverse_line* реализует алгоритм инверсии с использованием двух указателей: один указывает на начало строки, второй – на конец. Последовательно меняя местами символы, функция переворачивает строку до тех пор, пока указатели не встретятся.

2.3 Makefile и управление сборкой

Сборка программы автоматизирована с помощью Makefile. Компиляция выполняется поэтапно: исходные файлы (.c) компилируются в объектные (.o), затем они объединяются в исполняемый файл. Реализованы цели make (сборка), $make\ clean$ (очистка временных файлов) и $make\ test$ (проверка работы). Это упрощает управление проектом, ускоряет пересборку и автоматизирует тестирование.

3 ХОД ВЫПОЛНЕНИЯ ПРОГРАММЫ

3.1 Примеры выполнения задания

Ha рисунке 3.1 продемонстрирована работа программы и использование *Makefile* (*make*, *clean*, *test*).

```
karina@Karina-Aspire-A515-45:~/Labs_OSISP/invert_filter$ make
gcc -c src/main.c -o obj/main.o
gcc -c src/invert.c -o obj/invert.o
gcc -o invert_filter ./obj/main.o ./obj/invert.o
karina@Karina-Aspire-A515-45:~/Labs_OSISP/invert_filter$ ./invert_filter input.txt output.txt
karina@Karina-Aspire-A515-45:~/Labs_OSISP/invert_filter$ ./invert_filter
hello, Nikita Yurievich!
!hciveiruY atikiN ,olleh
^c
karina@Karina-Aspire-A515-45:~/Labs_OSISP/invert_filter$ make test
echo "hello" | ./invert_filter
olleh
echo "reverse" | ./invert_filter > output.txt
cat output.txt
earina@Karina-Aspire-A515-45:~/Labs_OSISP/invert_filter$ make clean
rm -f invert filter ./obj/*.o
```

Рисунок 3.1 – Результаты работы

На рисунке 3.2 показано состояние файла input.txt, который использовался в качестве обрабатываемого при вызове ./invert_filter input.txt output.txt.

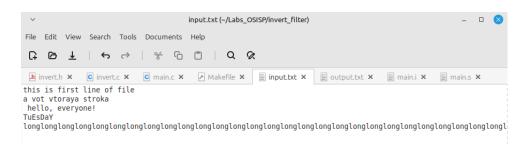


Рисунок 3.2 – Состояние файла *input.txt*

Состояние файла *output.txt* в результате обработки *input.txt* продемонстрировано на рисунке 3.3.

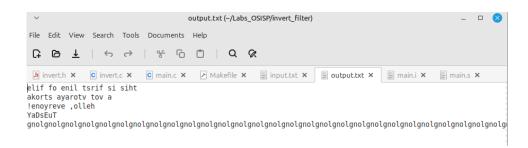


Рисунок 3.3 – Состояние файла *output.txt*

ВЫВОД

В ходе выполнения лабораторной работы была разработана и реализована программа на языке C, выполняющая инверсию порядка символов в каждой строке входного потока без изменения порядка строк. В процессе работы были изучены и применены основные инструменты разработки в среде Unix/Linux, включая компилятор GCC, систему управления сборкой Make и работу с файловыми потоками [3].

Программа поддерживает обработку данных как из стандартного ввода, так и из файлов, что делает ее удобной для использования. Реализована проверка входных данных, обеспечивающая корректную обработку аргументов командной строки и контроль возможных ошибок при открытии файлов. В основе алгоритма инверсии строк лежит использование двух указателей, позволяющих эффективно менять местами символы.

автоматизации процесса сборки, очистки и тестирования программы был разработан Makefile, который позволяет компилировать проект, управлять временными файлами и проводить тестирование работы Проведенные тесты программы заранее подготовленных данных. работоспособность подтвердили программы, корректность входных данных и соответствие результата заданным требованиям. В результате работы был получен инструмент для обработки текстовых данных, демонстрирующий применение работы с файлами и автоматизации сборки в Unix/Linux.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Makefile Tutorial By Example [Электронный ресурс]. Режим доступа: https://makefiletutorial.com/. Дата доступа: 14.02.2024.
- [2] gcc Command in Linux [Электронный ресурс]. Режим доступа: https://www.tutorialspoint.com/unix_commands/gcc.htm. Дата доступа: 14.02.2024.
- [3] Работа с файлами через язык Си [Электронный ресурс]. Режим доступа: https://itproger.com/course/c-programming/9. Дата доступа: 14.02.2024.

ПРИЛОЖЕНИЕ А (справочное) Исходный код

```
# main.c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "invert.h"
int main(int argc, char *argv[]) {
    FILE *input = stdin;
    FILE *output = stdout;
    if (argc > 1) {
        input = fopen(argv[1], "r");
        if (!input) {
            perror("Ошибка открытия входного файла");
            return 1;
        }
    }
    if (argc > 2) {
        output = fopen(argv[2], "w");
        if (!output) {
            perror("Ошибка открытия выходного файла");
            fclose(input);
            return 1;
        }
    char line[MAX LINE LENGTH];
    while (fgets(line, MAX LINE LENGTH, input)) {
        size t len = strlen(line);
        if (line[len - 1] == '\n') {
            line[len - 1] = ' \setminus 0';
            len--;
        reverse line(line, len);
        fprintf(output, "%s\n", line);
    if (input != stdin) fclose(input);
    if (output != stdout) fclose(output);
    return 0;
# invert.c
#include "invert.h"
#include <string.h>
void reverse line(char *line, int length) {
    int i = \overline{0}, j = length - 1;
    while (i < j) {
        char temp = line[i];
        line[i] = line[j];
        line[j] = temp;
```

```
i++;
        j--;
    }
}
#invert.h
#ifndef INVERT_H
#define INVERT_H
#define MAX LINE LENGTH 1024
void reverse line(char *line, int length);
#endif
# Makefile
CC = gcc
TARGET = invert filter
PREF SRC = ./src/
PREF OBJ = ./obj/
SRC = $(PREF SRC)main.c $(PREF SRC)invert.c
OBJ = $(patsubst $(PREF SRC)%.c, $(PREF OBJ)%.o, $(SRC))
all: $(TARGET)
$(TARGET): $(OBJ)
      $(CC) -o $(TARGET) $(OBJ)
$(PREF OBJ)%.o: $(PREF SRC)%.c
      @mkdir -p $(PREF OBJ)
      (CC) -c < -0 
clean:
      rm -f $(TARGET) $(PREF_OBJ)*.o
test: $(TARGET)
      echo "hello" | ./$(TARGET)
      echo "reverse" | ./$(TARGET) > output.txt
      cat output.txt
```