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»

|  |  |
| --- | --- |
|  |  |

**ОТЧЕТ**

к лабораторной работе № 3

на тему «Основы программирования на С под Unix. Инструментарий программиста в Unix»

Выполнил             Я. Ю. Прескурел

Проверил                          Н. Ю. Гриценко

Минск 2024

**СОДЕРЖАНИЕ**

[1 Постановка задачи 3](#_Toc157722973)

[2 Краткие теоретические сведения 4](#_Toc157722974)

[3 Результаты выполнения лабораторной работы 5](#_Toc157722975)

[Выводы 6](#_Toc157722976)

[Список использованных источников 7](#_Toc157722977)

[Приложение А (обязательное) Листинг исходного кода 8](#_Toc157722978)

# **1 ПОСТАНОВКА ЗАДАЧИ**

Целью выполнения данной лабораторной работы является изучение среды программирования и основных инструментов, таких как компилятор/сборщик gcc, управление обработкой проекта make и языка makefile. Кроме того, на практике необходимо написать программу, на языке программирования С, реализующую шифрование символов по азбуке Морзе. Также необходимо создать makefile для управления обработкой проекта, собрать исполняемый файл.

# **2 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

Язык программирования С является языком общего назначения, который широко используется для разработки системного программного обеспечения, прикладных программ и встраиваемых систем.[1]

GCC – это коллекция компиляторов для различных языков программирования, включая С. Также в данную коллекцию входят компиляторы для языков С++, Go. В зависимости от расширения имени файла, передаваемых в качестве параметра, и дополнительных опций, GCC запускает необходимые препроцессоры, компиляторы, сборщики.[2]

Компиляция – это процесс преобразования исходного кода программы из языка высокого уровня в машинный код. Компиляция происходит с помощью компилятора, который анализирует исходный код. Процесс компиляции преобразует исходный код программы в объектные файлы. Сборщик же создает исполняемый файл из одного или нескольких объектных файлов, полученных в результате компиляции.

Make – это утилита для автоматизации процесса сборки программы из исходных файлов. Make использует файл makefile, который содержит правила для компиляции и сборки проекта.[3]

Для больших проектов использование makefile и утилиты make позволяет достаточно сократить время компиляции и сборки.

Для выполнения данной лабораторной работы были использованы следующие сведения и концепции:

1 Язык программирования С: код программы был полностью написан на языке программирования С.

2 Стандартные библиотеки языка С: для работы с функциями ввода и вывода, строками и символами были использованы стандартные библиотеки языка С.

3 Компилятор GCC: для компиляции и сборки программы был использован компилятор GCC, который включает в себя компилятор языка С.

4 Система сборки make: для автоматизации процесса компиляции и сборки программы был создан makefile, в котором описаны все правила для компиляции и сборки различных модулей программы.

5 Модульное программирование: программа разделена на несколько модулей, каждый из которых отвечает за определенный функционал.

# **3 РЕЗУЛЬТАТЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ**

В ходе выполнения лабораторной была разработана программа, которая позволяет шифровать текст по азбуке Морзе. Пользователю необходимо ввести данные в виде последовательного набора символ из латинского алфавита. Результат работы программы представлен на рисунке 3.1.

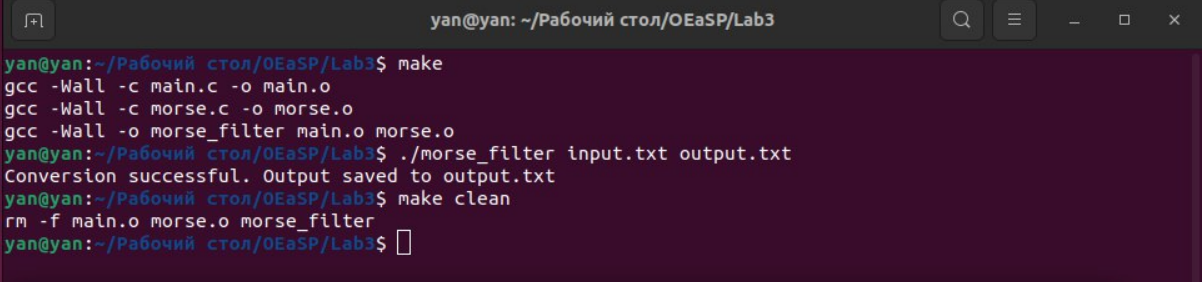


Рисунок 3.1 – Результат работы программы

При помощи утилиты make была организована возможность сборки и отчистки. Возможный шифрованный текст и фактически полученные результаты должны быть занесены в определенные файлы перед сборкой и после нее. Результат входных и выходных данных представлен на рисунке 3.2.

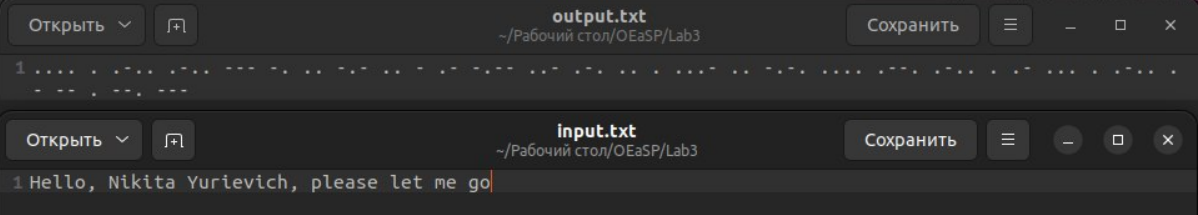


Рисунок 3.2 – Результат входных и выходных данных

Таким образом, в ходе данной лабораторной работы была реализована программа шифрующая символы по азбуке Морзе.

# **ВЫВОДЫ**

В ходе лабораторной работы были изучена среда программирования и основные инструменты, такие как компилятор/сборщик gcc, управление обработкой проекта make и языка makefile. Кроме того, была разработана программа на языке программирования С, реализующая шифрование символов по азбуке Морзе. Также был создан makefile для управления обработкой проекта, а также сборки исполняемого файла.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Язык программирования С [Электронный ресурс]. – Режим доступа: https://metanit.com/c/tutorial/1.2.php. – Дата доступа: 28.02.2024.

[2] Компилятор GCC [Электронный ресурс]. – Режим доступа: https://metanit.com/c/tutorial/1.3.php. – Дата доступа: 28.02.2024.

[3] Makefile [Электронный ресурс]. – Режим доступа:   
https://habr.com/ru/articles/155201/. – Дата доступа: 28.02.2024.

## **ПРИЛОЖЕНИЕ А**

## **(обязательное)**

## **Листинг исходного кода**

Листинг 1 – Программный код main.c

#include <stdio.h>

#include <stdlib.h>

#include "morse.h"

int main(int argc, char \*argv[]) {

if (argc != 3) {

fprintf(stderr, "Usage: %s input\_file output\_file\n", argv[0]);

exit(EXIT\_FAILURE);

}

const char \*input\_file = argv[1];

const char \*output\_file = argv[2];

if (convertToMorse(input\_file, output\_file)) {

printf("Conversion successful. Output saved to %s\n", output\_file);

} else {

fprintf(stderr, "Error during conversion.\n");

}

return 0;

}

Листинг 2 – Программный код morse.h

#ifndef MORSE\_H

#define MORSE\_H

int convertToMorse(const char \*input\_file, const char \*output\_file);

#endif

Листинг 3 – Программный код morse.c

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include "morse.h"

const char \*charToMorse(char ch) {

switch (toupper(ch)) {

case 'A': return ".- ";

case 'B': return "-... ";

case 'C': return "-.-. ";

case 'D': return "-.. ";

case 'E': return ". ";

case 'F': return "..-. ";

case 'G': return "--. ";

case 'H': return ".... ";

case 'I': return ".. ";

case 'J': return ".--- ";

case 'K': return "-.- ";

case 'L': return ".-.. ";

case 'M': return "-- ";

case 'N': return "-. ";

case 'O': return "--- ";

case 'P': return ".--. ";

case 'Q': return "--.- ";

case 'R': return ".-. ";

case 'S': return "... ";

case 'T': return "- ";

case 'U': return "..- ";

case 'V': return "...- ";

case 'W': return ".-- ";

case 'X': return "-..- ";

case 'Y': return "-.-- ";

case 'Z': return "--.. ";

case '0': return "----- ";

case '1': return ".---- ";

case '2': return "..--- ";

case '3': return "...-- ";

case '4': return "....- ";

case '5': return "..... ";

case '6': return "-.... ";

case '7': return "--... ";

case '8': return "---.. ";

case '9': return "----. ";

default: return "";

}

}

int convertToMorse(const char \*input\_file, const char \*output\_file) {

FILE \*input = fopen(input\_file, "r");

FILE \*output = fopen(output\_file, "w");

if (input == NULL || output == NULL) {

perror("Error opening files");

return 0;

}

char ch;

while ((ch = fgetc(input)) != EOF) {

if (isalnum(ch)) {

fprintf(output, "%s", charToMorse(ch));

}

}

fclose(input);

fclose(output);

return 1;

}

Листинг 4 – Программный код makefile

CC = gcc

CFLAGS = -Wall

SRC = main.c morse.c

OBJ = $(SRC:.c=.o)

TARGET = morse\_filter

all: $(TARGET)

$(TARGET): $(OBJ)

$(CC) $(CFLAGS) -o $(TARGET) $(OBJ)

%.o: %.c

$(CC) $(CFLAGS) -c $< -o $@

clean:

rm -f $(OBJ) $(TARGET)