МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Интеллектуальные информационные технологии»

ОТЧЕТ

По лабораторной работе №5

«ВВОД/ВВЫВОД»

**Выполнил:** Антонюк Н.А.

**Номер зачетной книжки 220203**

**Группа:** ПО-11

**Проверил:** Давидюк Ю.И.

Брест 2023

**Задание для выполнения**

**Часть I**

Ознакомиться с руководством по системным вызовам open, read, write, close. Вспомнить, что такое конвейер и перенаправление ввода-вывода (этот пункт задания может оказаться трудновыполнимым, если соответствующие знания не были приобретены в процессе работы над л/р №1-3).

**Часть II**

**Вариант 3.** Написать программу, которая получает со стандартного потока ввода любое

руководство, и выводит в стандартный поток вывода заголовки всех секций данного

руководства. Протестировать на различных командах с использованием конвейеров в различных

комбинациях вашей программы и команд man, sort, head, tail.

**Часть III**

**Вариант 3.** Создайте отдельные текстовые файлы для каждой секции руководства, и

поместите туда эти секции.

**Часть II**

**Код:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <stddef.h>

#include <sys/types.h>

#include <ctype.h>

#define MAX\_STRING\_SIZE 400

int main()

{

size\_t sizeRead, sizeWrite;

char data[MAX\_STRING\_SIZE] = {};

char string[MAX\_STRING\_SIZE] = {};

int indexStr = 0;

char header[30];

/\* Считываем данные из стандартного потока ввода \*/

while ((sizeRead = read(0, data, MAX\_STRING\_SIZE)) > 0)

{

if (sizeRead <= 0)

{

perror("read");

return -1;

}

// Проверяем каждый символ в строке

for (int i = 0; i < sizeRead; i++)

{

if (data[i] == '\n')

{

// Если найден символ новой строки, выводим сохраненную часть строки

string[indexStr] = data[i];

if(string[0] != ' ' && string[0] != '\n')

{

for (int l = 0; l < strlen(string) + 1; l++)

{

if ((!islower(string[l]) || string[l] == ' ') && !isdigit(string[l]))

{

header[l] = string[l];

}

else

{

memset(header, 0, strlen(header));

break;

}

}

write(1, header, strlen(header));

}

indexStr = 0;

memset(string, 0, strlen(string));

}

else

{

// Сохраняем символ в строке

string[indexStr] = data[i];

indexStr++;

}

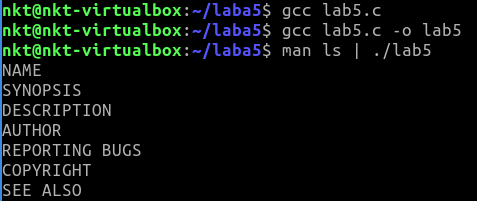
}

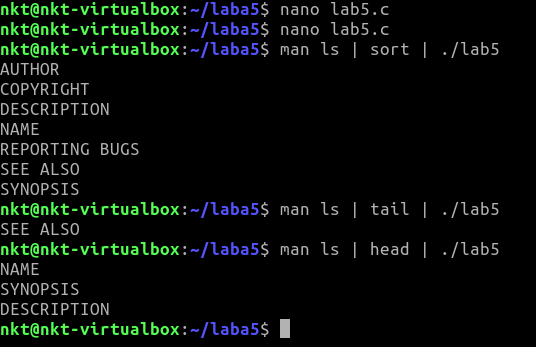
}

return 0;

}

**Вывод:**





**Часть III**

**Код:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/types.h>

#include <stddef.h>

#include <ctype.h>

#include <fcntl.h>

#include <stdbool.h>

#define MAX\_STRING\_SIZE 400

int main()

{

size\_t sizeRead, sizeWrite;

char data[MAX\_STRING\_SIZE] = {};

char string[MAX\_STRING\_SIZE] = {};

int fd;

bool flag = false;

int indexStr = 0;

char header[30];

// Считываем данные из стандартного потока ввода

while ((sizeRead = read(0, data, MAX\_STRING\_SIZE)) > 0)

{

if (sizeRead <= 0)

{

perror("read");

return -1;

}

// Проверяем каждый символ в строке

for (int i = 0; i < sizeRead; i++)

{

if (data[i] == '\n')

{

// Если найден символ новой строки, выводим сохраненную часть строки

string[indexStr] = data[i];

if (string[0] != ' ' && string[0] != '\n')

{

if (flag == true)

{

if (close(fd) < 0)

{

perror("close");

return -1;

}

}

for (int l = 0; l < strlen(string) + 1; l++)

{

if ((!islower(string[l]) || string[l] == ' ') && !isdigit(string[l]))

{

header[l] = string[l];

}

else

{

memset(header, 0, strlen(header));

break;

}

}

// Исключаем лишние символы новой строки и пробелы в начале и конце строки

int length = strlen(header);

while (length > 0 && (header[length - 1] == '\n' || header[length - 1] == ' '))

{

header[length - 1] = '\0';

length--;

}

if (length > 0)

{

(void)umask(0);

if ((fd = open(header, O\_RDWR | O\_CREAT | O\_APPEND | O\_TRUNC, 0666)) < 0)

{

perror("open");

return -1;

}

else

{

flag = true;

}

}

}

if (string[0] == ' ')

{

sizeWrite = write(fd, string, strlen(string));

if (sizeWrite != strlen(string))

{

// Если записалось меньшее количество байт, сообщаем об ошибке

perror("write");

return -1;

}

}

indexStr = 0;

memset(string, 0, strlen(string));

}

else

{

// Сохраняем символ в строке

string[indexStr] = data[i];

indexStr++;

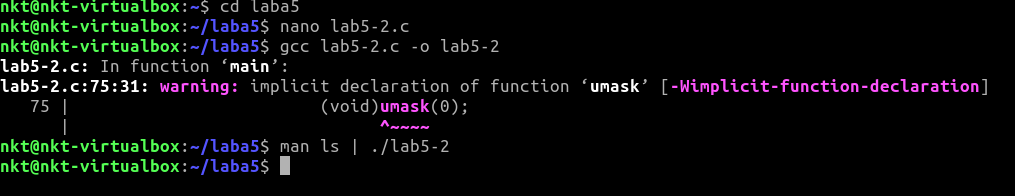
}

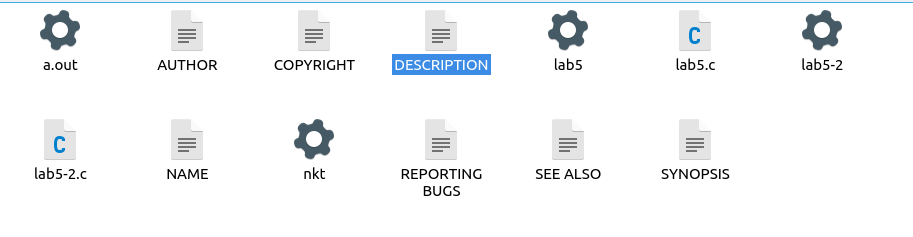
}

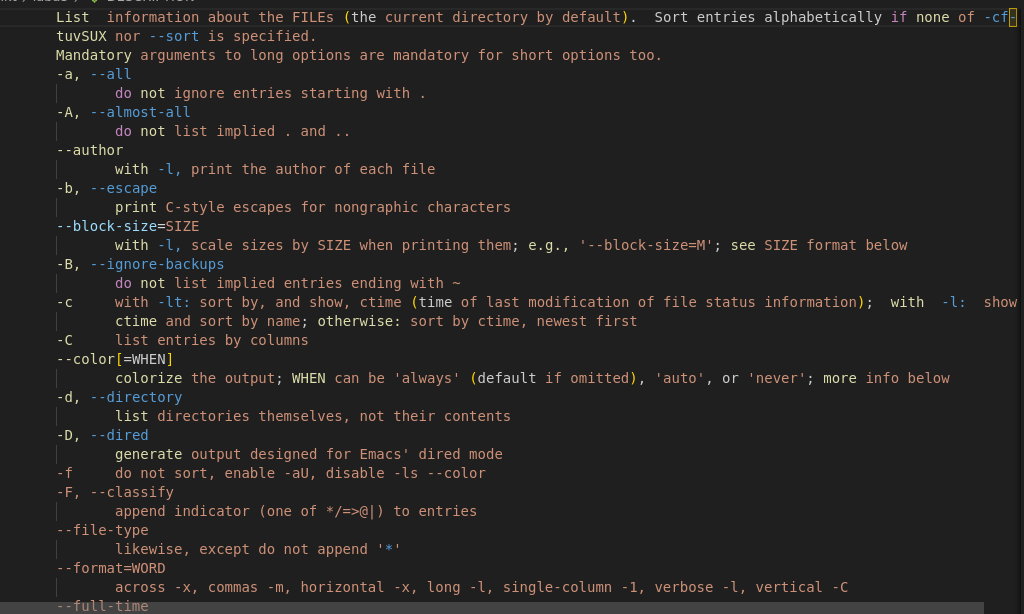
}

return 0;

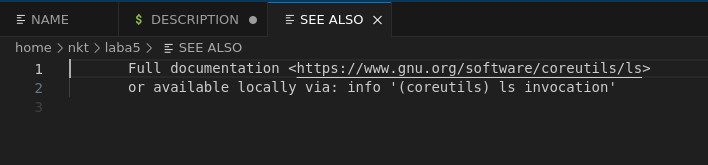
}





Запись в папку DESCRIPTION

Запись в папку SEE ALSO



Запись в папку REPORTING BUGS

