Министерство образования и науки РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Омский государственный технический университет»

Кафедра «Информатика и вычислительная техника»

**ОТЧЕТ**

По лабораторной работе №1

дисциплина «Операционные системы»

Выполнил**:** студент гр. ПИН-221

Кит Д.В.

Проверил: ст. преподаватель

Горшенин А.Ю

2023

**Цель работы**. Ознакомление со средствами стандартного ввода-вывода и их внешнего к исполняемой программе использованию путем переадресации стандартного ввод-вывода из командной строки.

**Краткие теоретические сведения**.

Процесс текстового взаимодействия с пользователем при непосредственном командном общении с операционной системой выполняется в терминах записи и чтения в файл. При этом вывод в текстовую консоль представляется как запись в файл, а ввод — как чтение файла. Файл, из которого осуществляется чтение, называется стандартным потоком ввода или просто стандартным вводом (standard input или stdin), а в который осуществляется запись — стандартным потоком вывода (упрощенно – стандартным выводом – standard output или stdout). Стандартные потоки — воображаемые файлы, позволяющие осуществлять взаимодействие с пользователем как чтение и запись в файл. Кроме потоков ввода и вывода, существует еще и стандартный поток ошибок (стандартный вывод ошибок – standard error или stderr), на который выводятся все сообщения об ошибках и те информативные сообщения о ходе работы программы, которые не могут быть выведены в стандартный поток вывода. По умолчанию входной поток связан с клавиатурой, а выходной поток и поток сообщений об ошибках направлены на терминал пользователя. Другими словами, вся выходная информация запущенной пользователем команды или программы, а также все сообщения об ошибках, выводятся в окно терминала. Для ввода из или вывода в стандартный поток не требуется открытие файла, поскольку благодаря предусмотрительности разработчиков такие абстрактные файлы открываются самой операционной системой, причем незаметно для пользователей. Стандартные потоки в большинстве операционных систем, к которым не относятся ОС Windows, связаны c файловыми дескрипторами, имеющими номера 0, 1 и 2. В операционной системе Windows использован более сложный и громоздкий для программиста подход. Для получения хэндлов стандартного ввода-вывода следует использовать специальную функцию API с именем GetStdHandle. Функция имеет единственный аргумент, задаваемый одной из системных констант STD\_INPUT\_HANDLE, STD\_OUTPUT\_HANDLE, STD\_ERROR\_HANDLE. Возвращаемое по выполнению этой функции значение и дает реальное текущее значение хэндла, которое далее можно использовать в системных функциях ввода и вывода ReadFile и WriteFile. Сами эти возвращаемые значения сложным образом зависят от текущих обстоятельств. Стандартный ввод-вывод не обязательно использовать по умолчанию, выдавая с помощью него текст в терминал, а вводя данные только с клавиатуры. Замечательной возможностью является переназначение его составляющих частей на любой обычный файл. Это переназначение выполняется на этапе вызова исполняемой программы, а детали переназначения указываются в командной строке вызова программы. В простейшем и большинстве профессиональных вариантов общения с ОС такой вызов задается с помощью командного интерпретатора непосредственно в командной консоли. Он заключается в том, что в вызов программы, кроме собственно ее имени, включается дополнительный служебный текст со специальным синтаксисом. В простейших случаях такой вызов имеет в командной строке общения с ОС следующий вид:

имя\_исполняемой\_программы параметры\_вызова\_программы

Все семантические части такого вызова разделяются пробелами, а не скобками или запятыми, как в большинстве языков программирования.

Для переадресации стандартного ввода и стандартного вывода в командной строке используются вспомогательные символы '<' и '>', соответственно.

Для переадресации стандартного вывода используется запись вида

имя\_исполняемой\_программы >имя\_файла\_для\_вывода

Для переадресации стандартного ввода используется запись вида

имя\_исполняемой\_программы <имя\_файла\_с\_вводимыми\_данными

Для переадресации стандартного потока ошибок используется запись вида

имя\_исполняемой\_программы 2>имя\_файла

Предоставляется также возможность направлять данные об ошибках в

стандартный поток обычного вывода. Для этого предназначена запись вида

имя\_исполняемой\_программы 2>&1

При использовании переназначений с помощью символов обозначений

>имя\_файла предыдущее содержимое файла с именем имя\_файла теряется

(внутренними процедурами файл открывается для записи с усечением до нуля

предыдущего содержимого). В ряде случаев, особенно при протоколировании

ошибок, может представлять интерес сохранение предыдущего содержимого

указанного таким образом файла и приписывание в его конец новых данных.

Для решений этой задачи предлагается использовать вместо одного символа '>'

два таких символов подряд и без разрыва между ними.

В частности команда

имя\_исполняемой\_программы <файл1 >файл2 2>>файл3

указывает брать исходные данные в файле файл1, помещать данные стандартного вывода в файл файл2 и дописывать в конец файла файл3 сообщения стандартного потока ошибок текущей команды.

В операционных системах Windows имя\_исполняемой\_программы, при

нахождении пользователя, взаимодействующего с ОС именно в том каталоге, где находится запускаемая программа, задается просто ее собственным именем, причем допускается как запись с расширением имени, так и без него. В частности, запуск программы с именем lab1.exe может осуществляться как командной строкой

lab1.exe

так и командной строкой

lab1

В операционных системах Unix/Linux перед собственным именем программы должен явно указываться каталог ее нахождения. В частности, для текущего каталога используется сокращенное обозначение в виде

./собственное\_имя\_исполняемой\_программы

При это символ «точка» обозначает текущий каталог, а разделитель «/» используется как служебный для обозначения файла с учетом имени каталога.

В отличие от непрофессионального использования ОС с помощью щелчков мыши на графических изображениях файлов, профессиональное взаимодействие с операционными системами осуществляется преимущественно в тестовом режиме. Это содержательно полностью соответствует профессиональному общению между людьми, полноценное взаимодействие между которыми осуществляется с помощью естественных языков и специальных текстов, в частности математических. Использование графических образов в общении и специальных жестов характерно скорее для чисто биологических систем, при отсутствии у них «второй сигнальной системы», как термина, введенного русским физиологом И.П. Павловым. Заметим, что мозг животного отвечает лишь на непосредственные зрительные, звуковые и другие раздражения или их следы, возникающие ощущения составляют первую сигнальную систему действительности. Общение исключительно через графический интерфейс и манипуляторы типа «мышь» используют практически только эту древнюю систему. В процессе эволюции животного мира на этапе становления и начального развития вида Homo sapiens произошло качественное видоизменение системы сигнализации, переход к символам и знакам, имеющим текстовый последовательный характер, появились специализированные языки общения. Частным случаем их являются текстовые языки общения с операционными системами. В операционных системах типа Linux и Unix действующее значение хэндла стандартного ввода или вывода позволяет внутри выполняемой программы узнать, связан ли этот хэндл с обычным файлом или же с консолью, называвшейся ранее «терминалом». Для этих целей в составе API OS присутствует вспомогательная функция с именем isatty. Эта функция возвращает ненулевое значение, когда указанный в ней аргументом хэндл отвечает терминалу, а не файлу. В частности, проверив на очередном шаге выполнения нулевой хэндл, соответствующий стандартному вводу путем выполнения оператора

if (isatty(0)) оператор\_действий

можно выполнить те действия, которые программист планирует на случай текущего использования для ввода данных непосредственно клавиатуры.

ТЕКСТ ПРОГРАММЫ

Задание 1

#include "windows.h"

#include "stdio.h"

#include <iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL,"rus");

HANDLE inputHandle = GetStdHandle(STD\_INPUT\_HANDLE);

HANDLE outputHandle = GetStdHandle(STD\_OUTPUT\_HANDLE);

HANDLE errHandle = GetStdHandle(STD\_ERROR\_HANDLE);

if ((inputHandle == INVALID\_HANDLE\_VALUE) || (outputHandle == INVALID\_HANDLE\_VALUE))

{

if (errHandle != INVALID\_HANDLE\_VALUE) WriteFile(errHandle, "Error!", 7, NULL, NULL);

return 0;

}

char buf[200];

cout << "Input handle - " << inputHandle << "\n";

cout << "Output handle - " << outputHandle << "\n";

cout << "error handle- " << errHandle << "\n";

fflush(stdout);

WriteFile(outputHandle, "Input text\n ", 13, NULL, NULL);

DWORD readed;

ReadFile(inputHandle, buf, 200, &readed, NULL);

WriteFile(outputHandle, "Text: ", 6, NULL, NULL);

WriteFile(outputHandle, buf, readed, NULL, NULL);

return 0;

}

Задание 2.

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

char str[2000];

cout << "Input text: ";

cin.getline(str, 1000);

FILE\* f = fopen("File.txt", "wt+");

cout <<"You wrote: \n"<< str << "\n"<< "\ntext has written";

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

{

if (str[i] == '\0') break;

fputc(str[i], f);

}

fclose(f);

return 0;

}

СКРИНШОТЫ ВЫПОЛНЕНИЯ РАБОТАЮЩЕЙ ПРОГРАММЫ

Задание 1.

На рисунке 1 указан скрин компиляции программ и работы программы

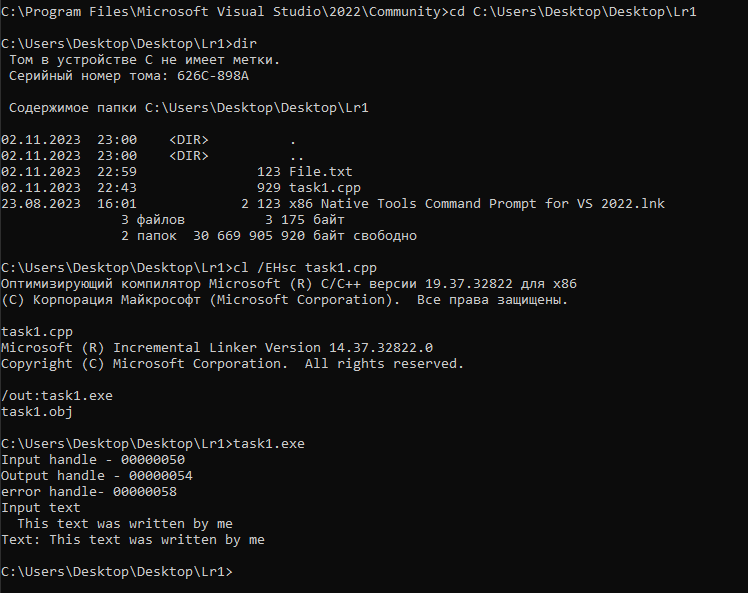


Рисунок 1 – процесс работы программы

На рисунке 2 показан процесс перенаправленного ввода в windows

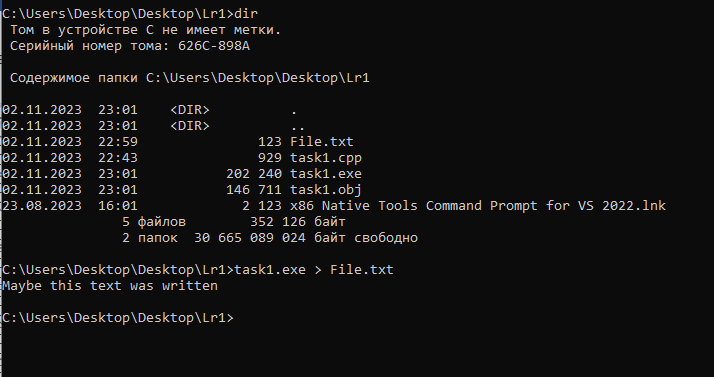


Рисунок 2 – процесс перенаправленного ввода в windows

На рисунке 3 показано содержимое File.txt

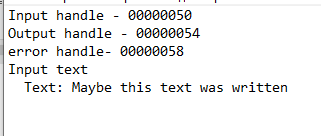
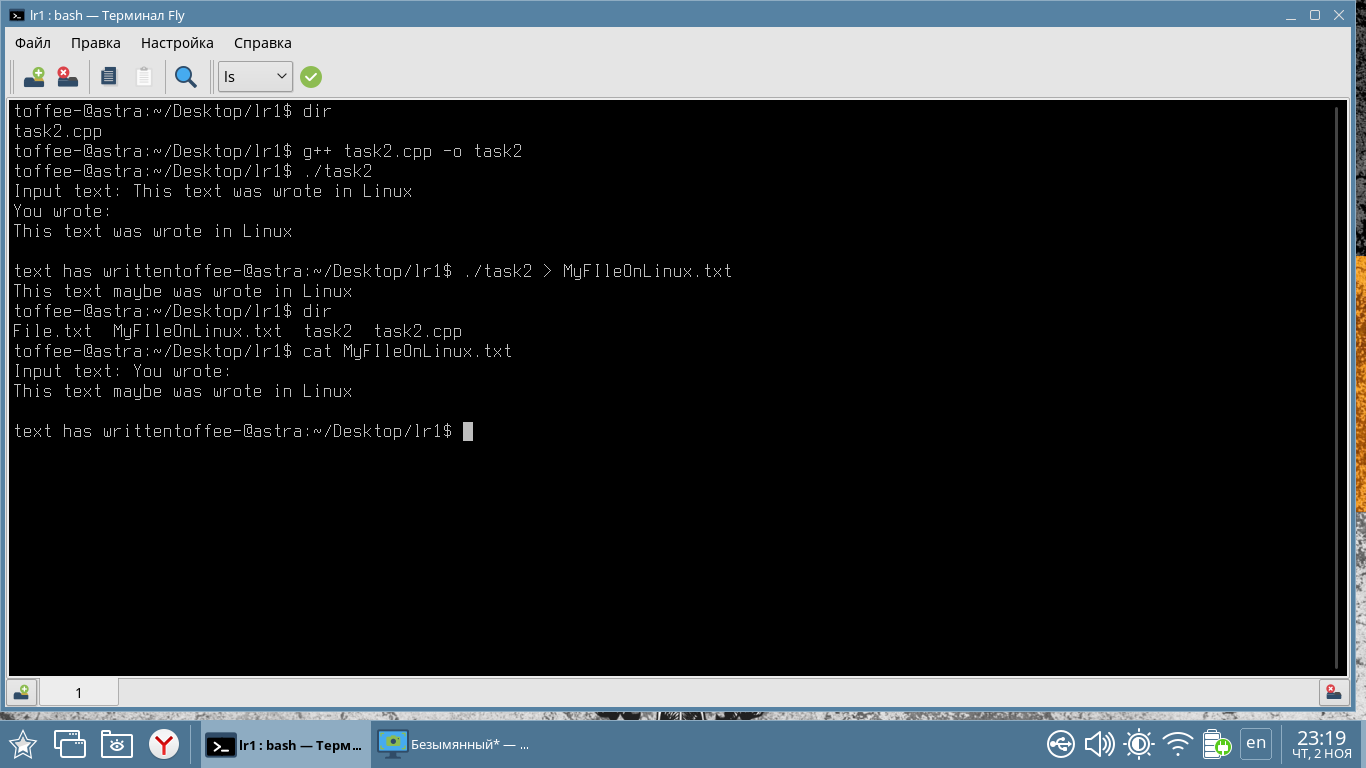
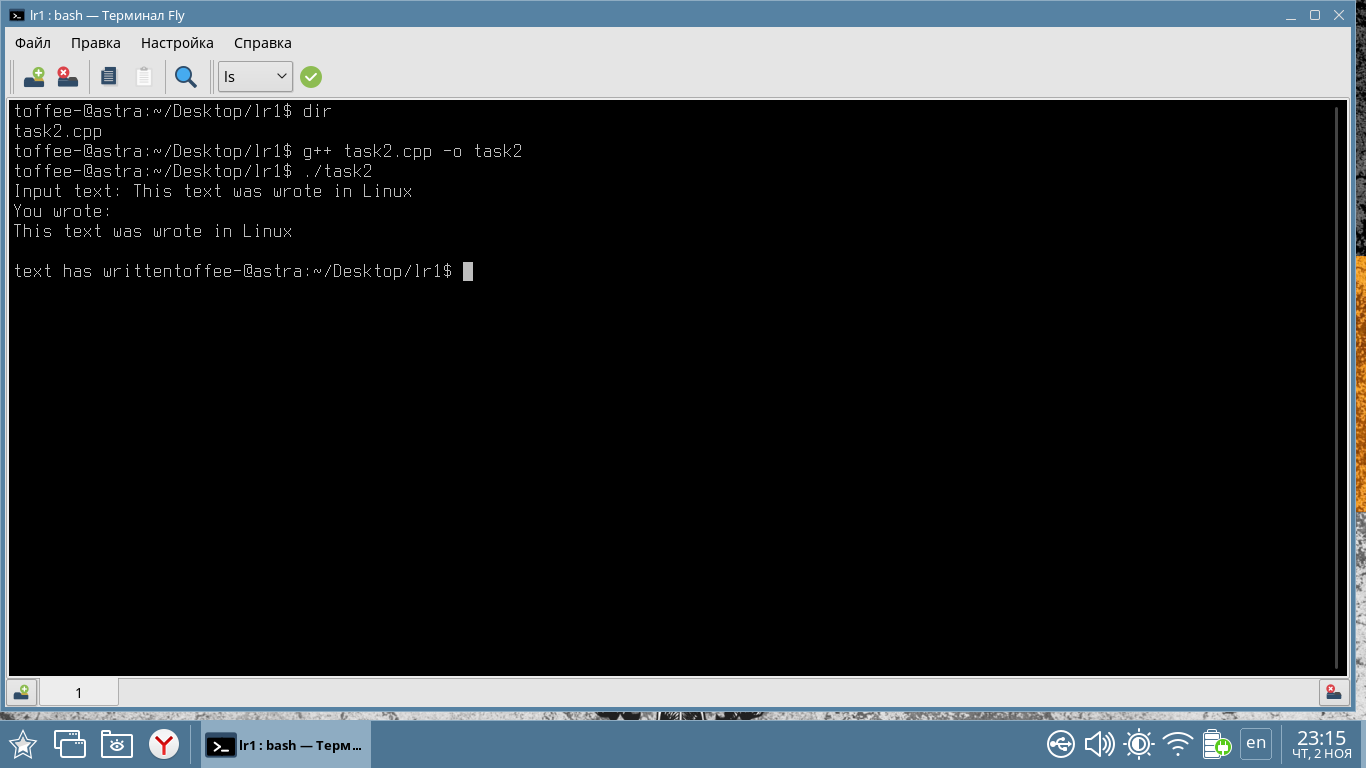


Рисунок 3 – содержимое File.txt

Задание 2.

На рисунке 4 показан процесс компиляции программы, работы программы с вводом текста от пользователя в консоли, а также перенаправленный ввод в файл MyFileOnLinux.txt

Также на рисунке показано содержимое этого файла после работы программы.

Рисунок 4 – процесс работы второй программы

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. В ходе построения исходной программы, использующей стандартный ввод-вывод, файловые дескрипторы или хэндлы записываются в операционной системе. Они могут быть созданы и использованы в различных частях программы, в зависимости от её логики и требований. Например, при открытии файла для чтения или записи, операционная система предоставляет файловый дескриптор или хэндл, который можно использовать для взаимодействия с этим файлом.
2. Переадресация стандартного ввода-вывода используется для изменения источника или места назначения данных, которые обрабатываются программой. Это полезно, когда необходимо перенаправить данные из файла в программу или вывести результаты выполнения программы в файл, а не на экран. Это также позволяет создавать потоки данных между различными программами, связывая их вместе.
3. Существует три разновидности стандартного ввода-вывода:

* Стандартный ввод (stdin): используется для ввода данных в программу. Обычно связан с клавиатурой или другим устройством ввода по умолчанию.
* Стандартный вывод (stdout): используется для вывода данных из программы. По умолчанию связан с экраном или консолью.
* Стандартная ошибка (stderr): используется для вывода сообщений об ошибках и отладочных сообщений. Также связан с экраном или консолью по умолчанию.

1. Вызов исполняемой программы в командной строке ОС Windows отличается от вызова в командной строке Linux следующими особенностями:

* В Windows исполняемая программа может иметь расширение файла (например, .exe), и для её вызова необходимо указывать полное имя файла с расширением.
* В Linux исполняемая программа не требует обязательного расширения файла и вызывается по имени файла без указания расширения.
* В Windows использование обратного слеша \ используется для разделения пути к файлу, а в Linux используется прямой слеш /.
* В Windows для указания параметров командной строки используется символ / (например, program.exe /param), а в Linux используется символ - или -- (например, program --param).

ВЫВОД

Ознакомился со средствами стандартного ввода-вывода и их внешнего к исполняемой программе использованию путем переадресации стандартного ввод-вывода из командной строки.

Были разработаны программы, две. Одна на Linux, другая на Windows. Программы работают с текстом, т.е. получают на вход текст и выводят в файл. Предусмотрены случаи если ввод или вывод были переназначены. Так же для пользователя выводились уведомления об вводе или чтении.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Флоренсов А.Н. Методические указания по лабораторным работам дисциплины

«ОПЕРАЦИОННЫЕ СИСТЕМЫ». Учеб. пос. – Омск, Изд-во ОмГТУ, 2019. – 45 с.