

**Федеральное государственное автономное
Образовательное учреждение высшего образования
Российский Университет Дружбы Народов**

Математический университет имени Никольского
Факультет Физико-математических и Естественных наук
Кафедра Прикладной математики и информатики

Отчет по лабораторной работе № 11
“ Программирование в командном
процессоре ОС UNIX. Ветвления и циклы”

Выполнил:
Студент группы НПИМбв-01-10
Адхамова Луиза Шухратовна

Москва
2024 год

Цель работы:

Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Выполнение:

1. Используя команды `grep`, напомним командный файл, который анализирует командную строку с ключами:

- `-iinputfile` — прочитать данные из указанного файла;
- `-ooutputfile` — вывести данные в указанный файл;
- `-rшаблон` — указать шаблон для поиска;
- `-C` — различать большие и малые буквы;
- `-n` — выдавать номера строк.

а затем ищет в указанном файле нужные строки, определяемые ключом `-r` (рис 1.1, 1.2, 1.3, 1.4)

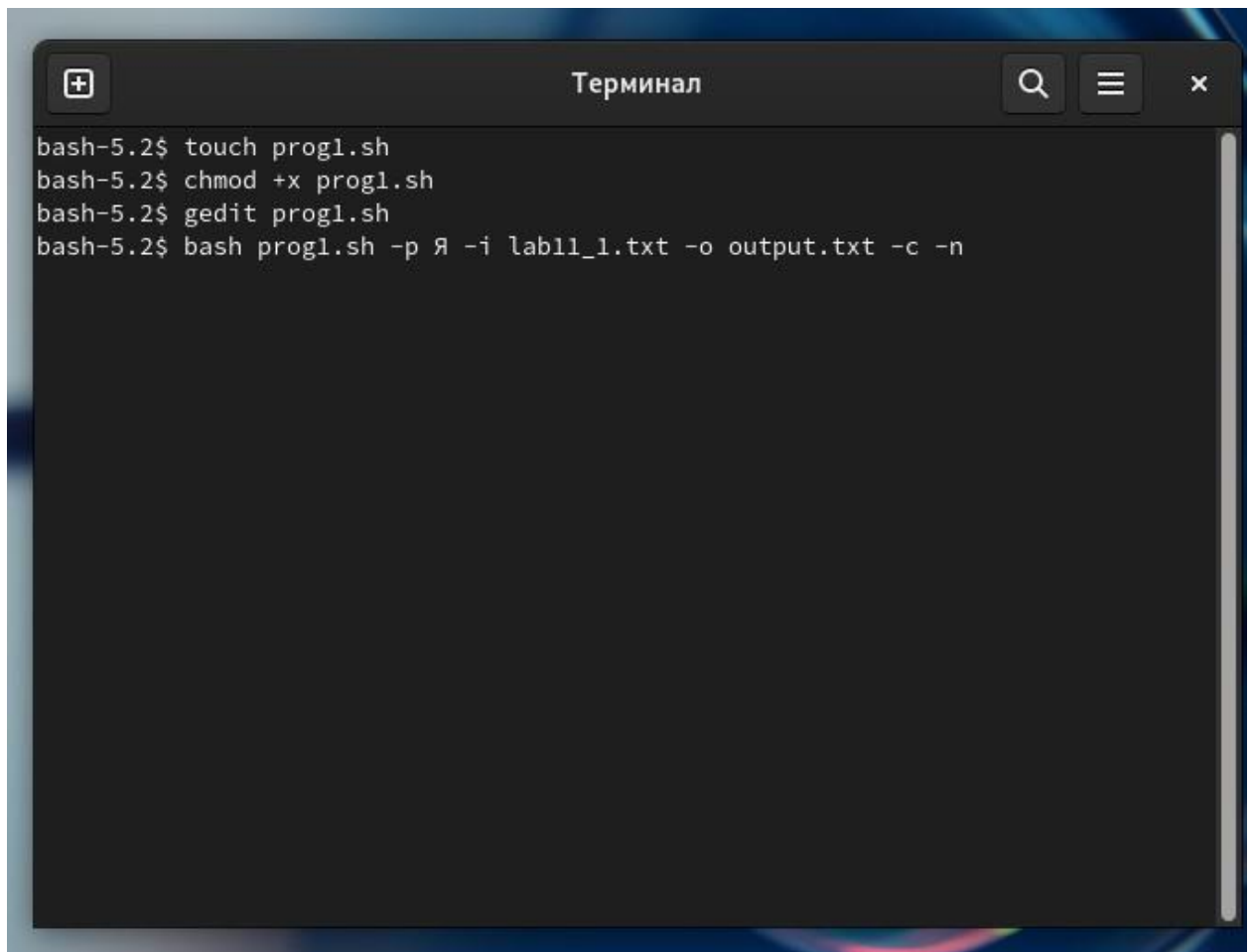


Рисунок 1.1. Файл.

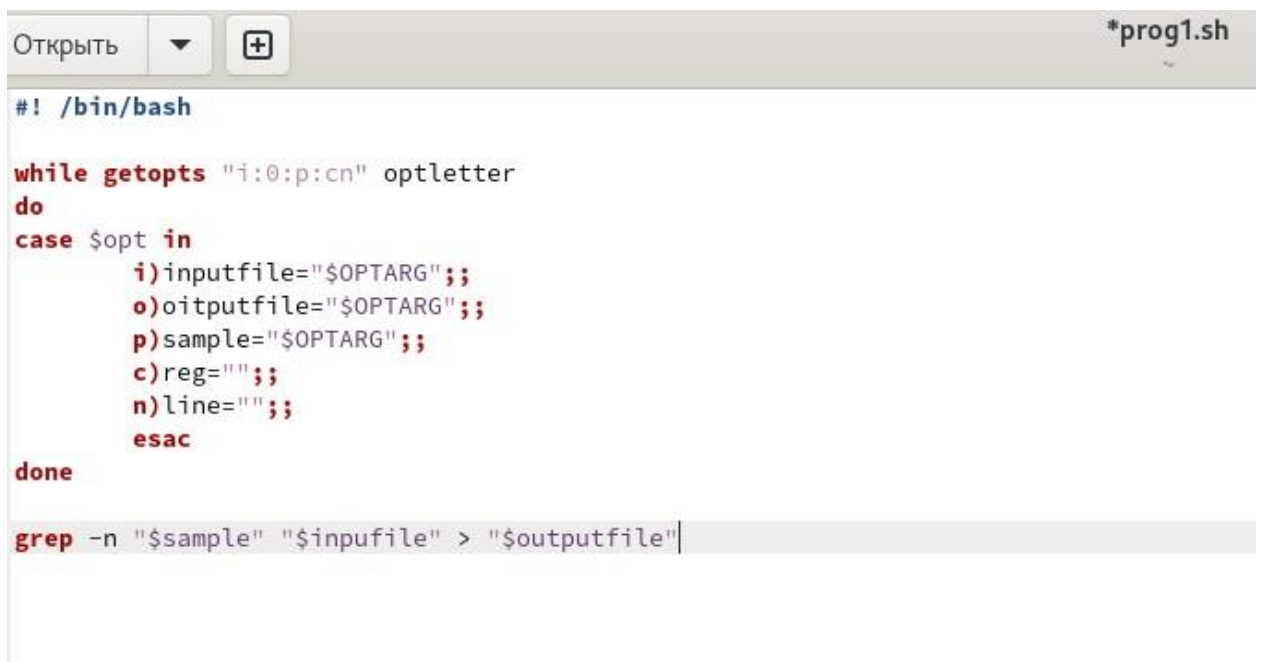
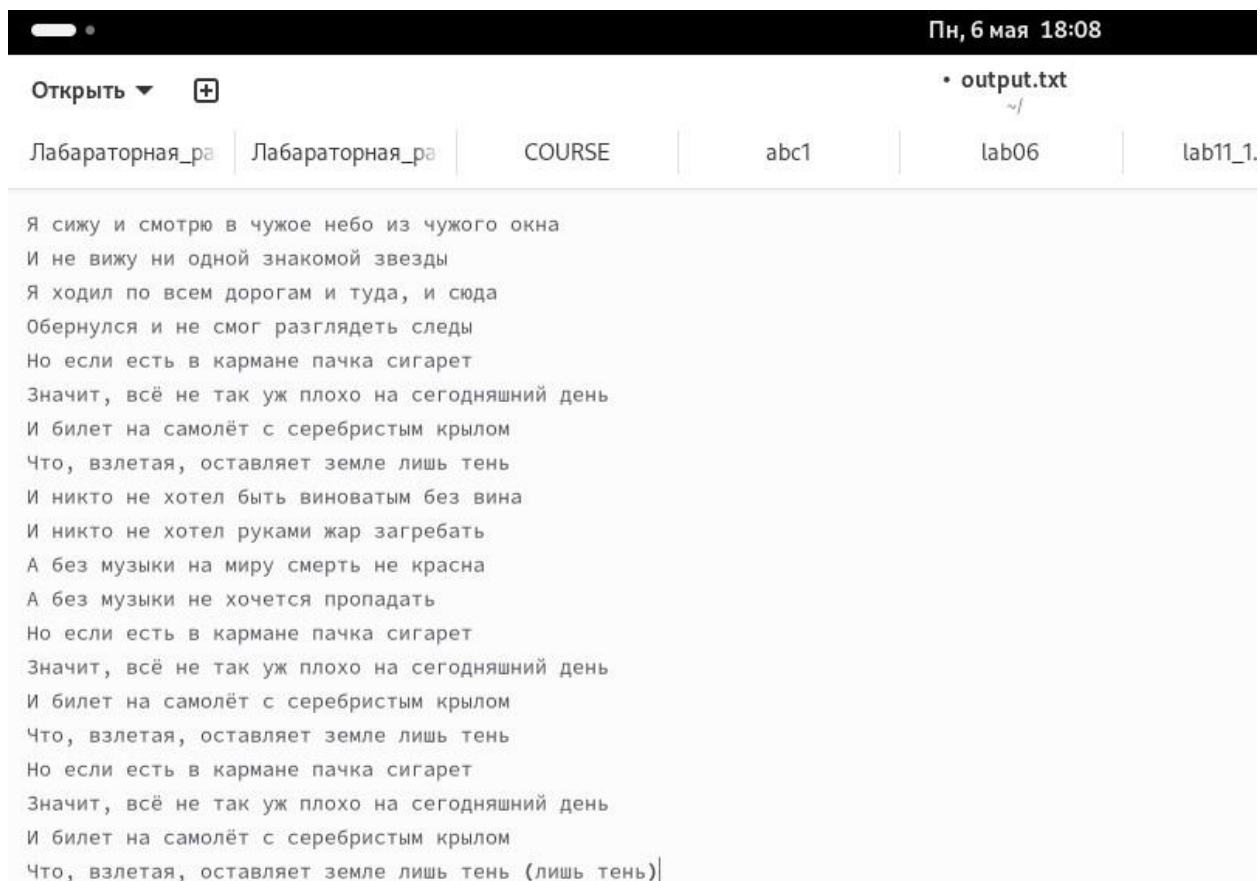


Рисунок 1.2. Программа.



Рисунок 1.3. Файл, в котором выполнялась команда.

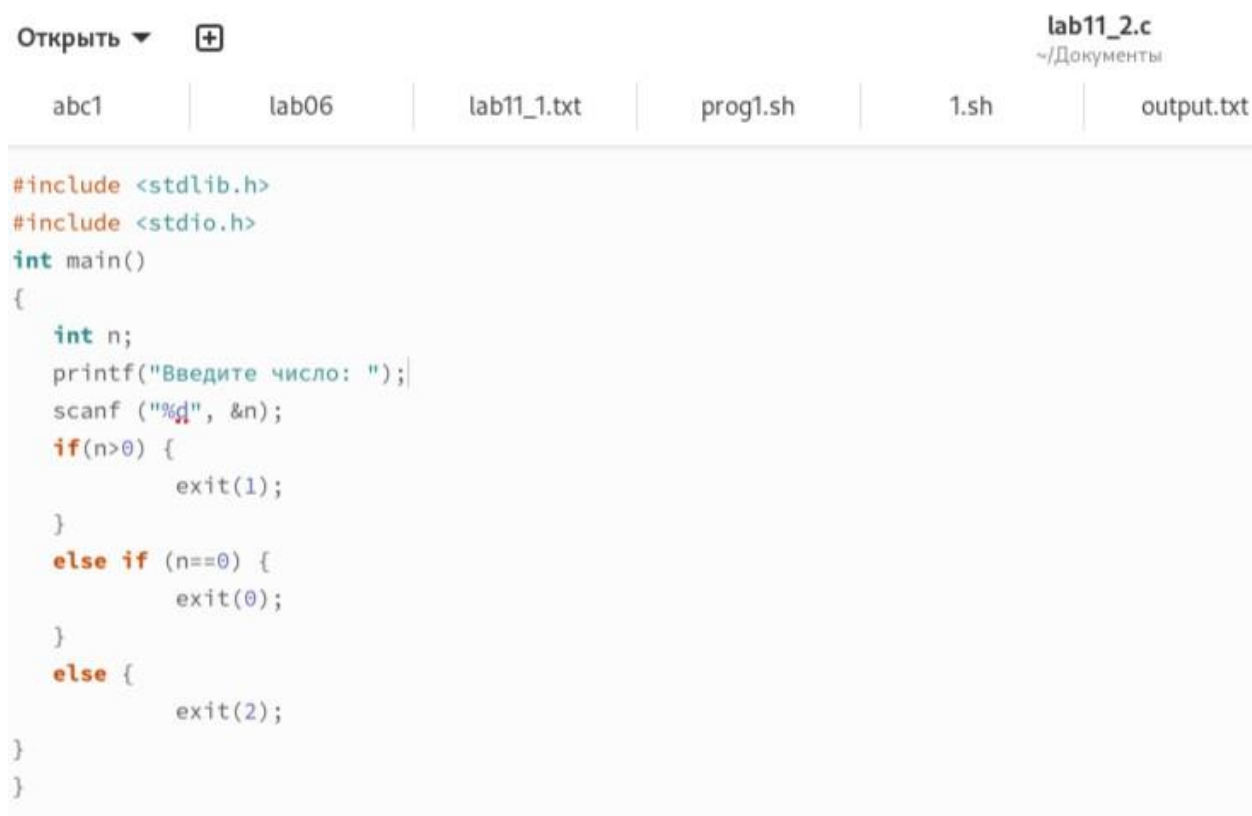


The screenshot shows a terminal window with a dark background. At the top right, the system time is displayed as 'Пн, 6 мая 18:08'. Below the title bar, there is a menu bar with 'Открыть' (Open) and a plus icon. To the right of the menu bar, a file named 'output.txt' is listed. Below the menu bar, there is a tab bar with several tabs: 'Лабараторная_ра', 'Лабараторная_ра', 'COURSE', 'abc1', 'lab06', and 'lab11_1'. The main area of the terminal displays the output of a C program, which is a poem about a person looking out a window and feeling lonely. The text is as follows:

```
Я сижу и смотрю в чужое небо из чужого окна
И не вижу ни одной знакомой звезды
Я ходил по всем дорогам и туда, и сюда
Обернулся и не смог разглядеть следы
Но если есть в кармане пачка сигарет
Значит, всё не так уж плохо на сегодняшний день
И билет на самолёт с серебристым крылом
Что, взлетая, оставляет земле лишь тень
И никто не хотел быть виноватым без вина
И никто не хотел руками жар загребать
А без музыки на миру смерть не красна
А без музыки не хочется пропадать
Но если есть в кармане пачка сигарет
Значит, всё не так уж плохо на сегодняшний день
И билет на самолёт с серебристым крылом
Что, взлетая, оставляет земле лишь тень
Но если есть в кармане пачка сигарет
Значит, всё не так уж плохо на сегодняшний день
И билет на самолёт с серебристым крылом
Что, взлетая, оставляет земле лишь тень (лишь тень)|
```

Рисунок 1.4. Результат.

2. Напишем на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции `exit(n)`, передавая информацию в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено (рис. 2.1, 2.2, 2.3)



```
#include <stdlib.h>
#include <stdio.h>
int main()
{
    int n;
    printf("Введите число: ");
    scanf ("%d", &n);
    if(n>0) {
        exit(1);
    }
    else if (n==0) {
        exit(0);
    }
    else {
        exit(2);
    }
}
```

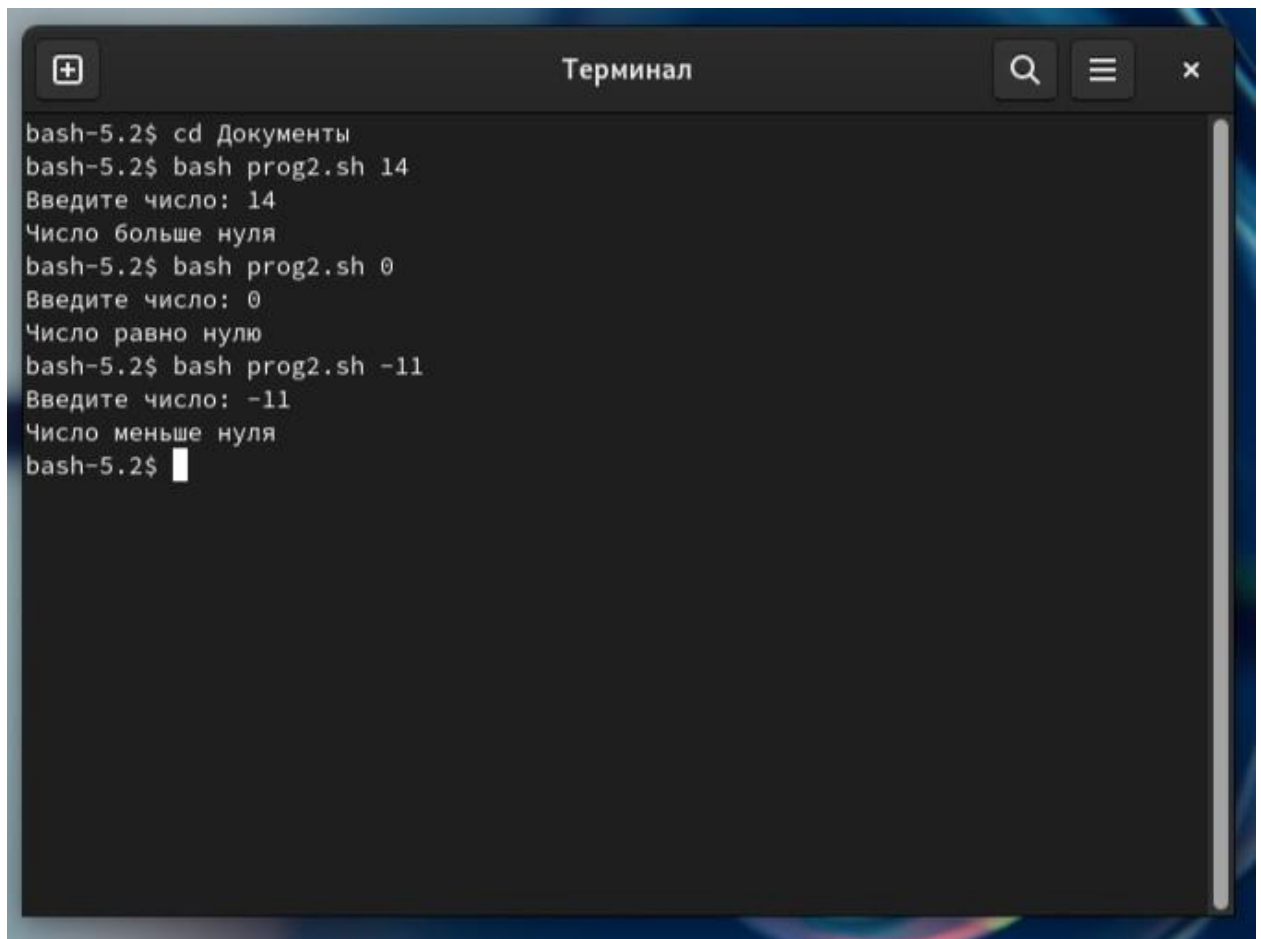
Рисунок 2.1. Программа на Си.



```
#!/bin/bash

gcc -o cprog lab11_2.c
./cprog
case $? in
0) echo "Число равно нулю";;
1) echo "Число больше нуля";;
2) echo "Число меньше нуля";;
esac
```

Рисунок 2.2. Программный файл.



```
bash-5.2$ cd Документы
bash-5.2$ bash prog2.sh 14
Введите число: 14
Число больше нуля
bash-5.2$ bash prog2.sh 0
Введите число: 0
Число равно нулю
bash-5.2$ bash prog2.sh -11
Введите число: -11
Число меньше нуля
bash-5.2$
```

Рисунок 2.3. Результат.

3. Напишем командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют) (рис. 3.1, 3.2).



Рисунок 3.1. Программа.

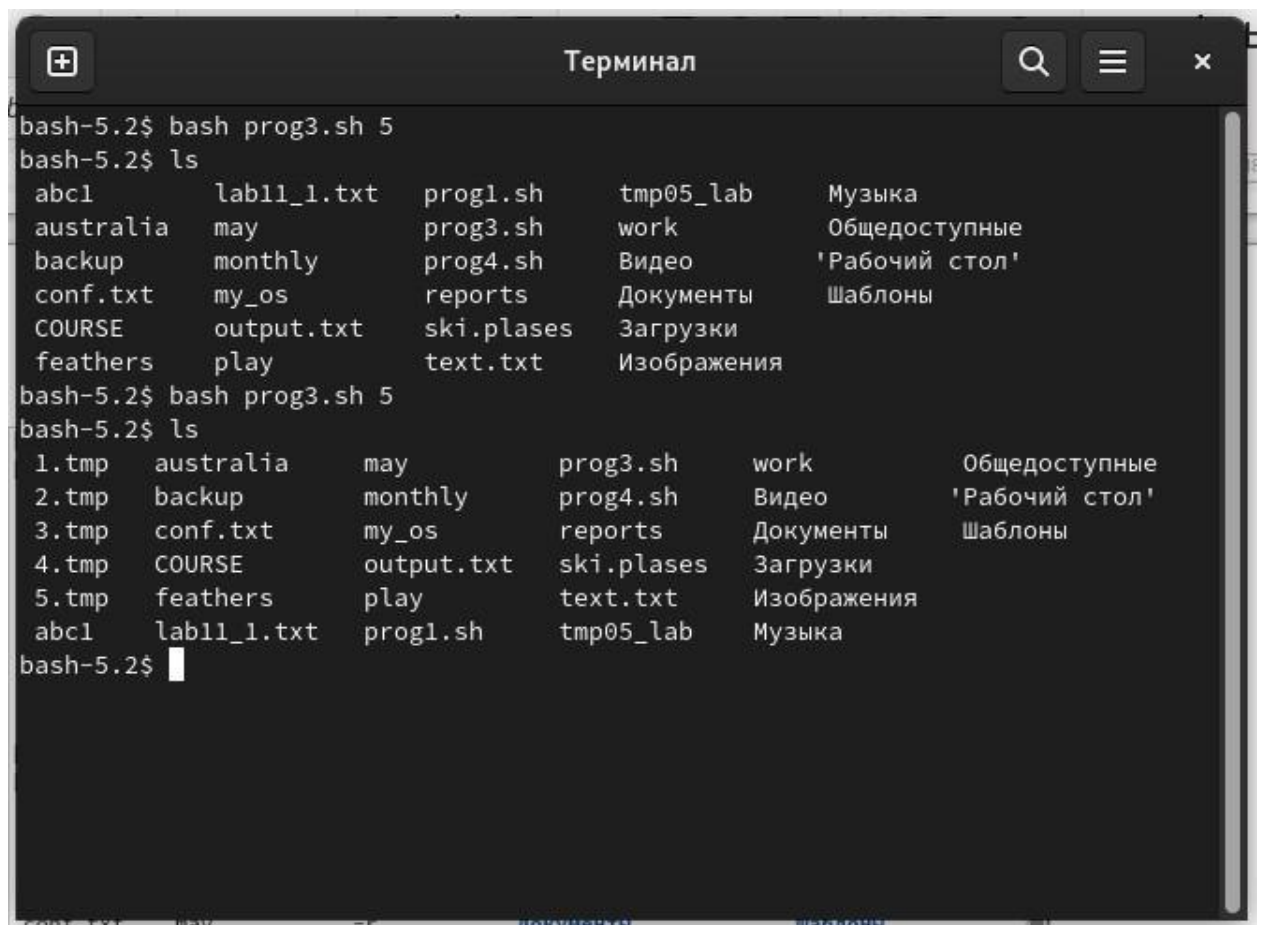


Рисунок 3.2. Результат.

4. Напишем командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицируем его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find) (рис. 4.1, 4.2, 4.3, 4.4).

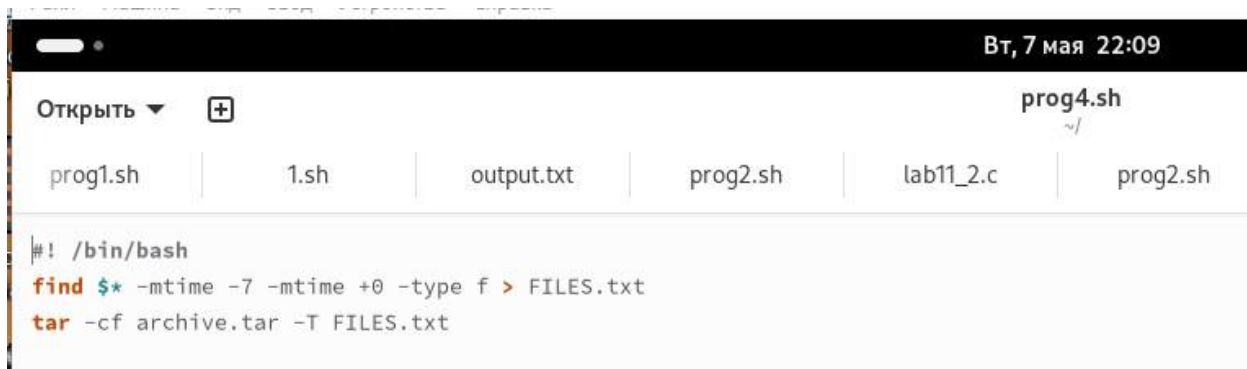


Рисунок 4.1. Программа.

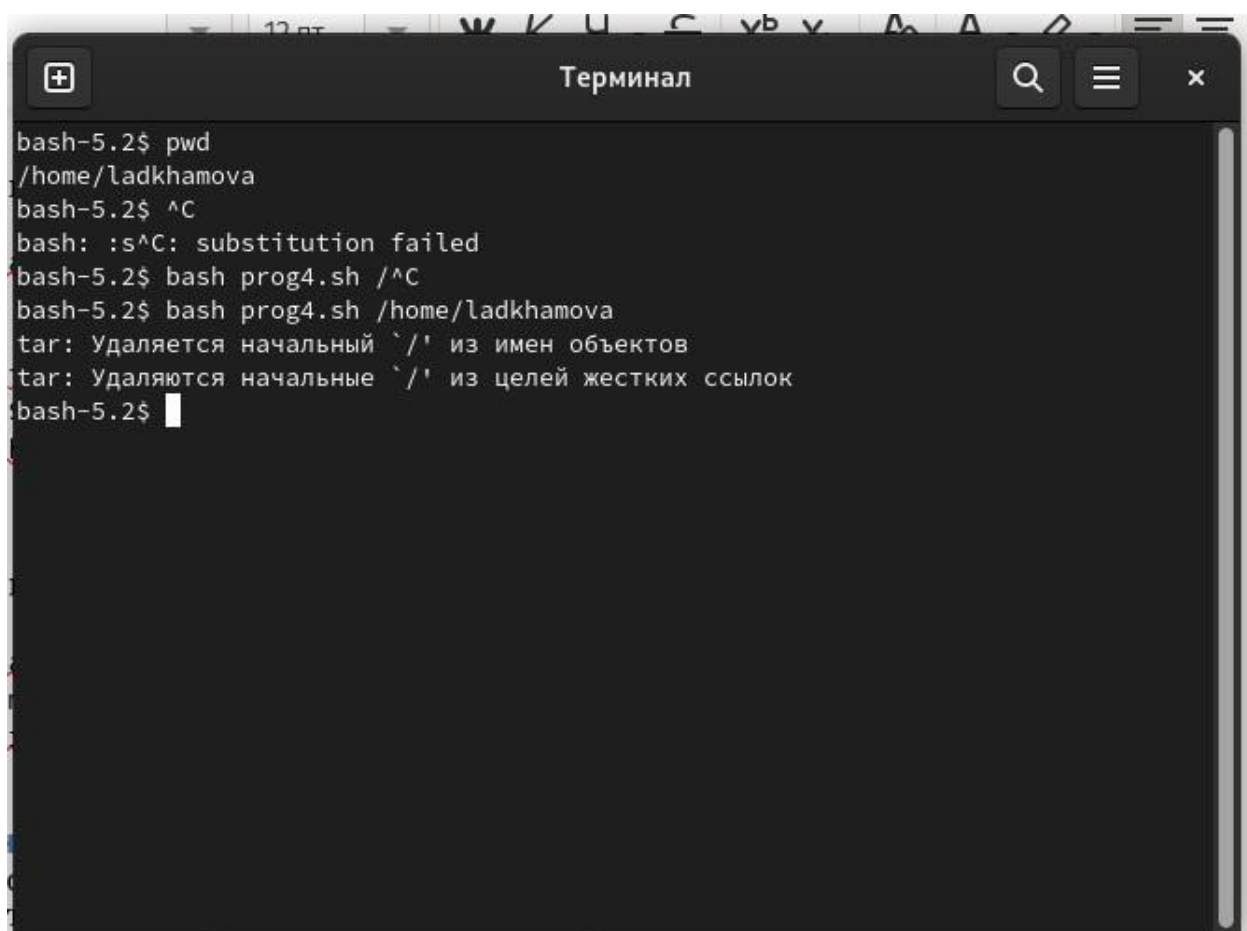


Рисунок 4.2. Текст программы.

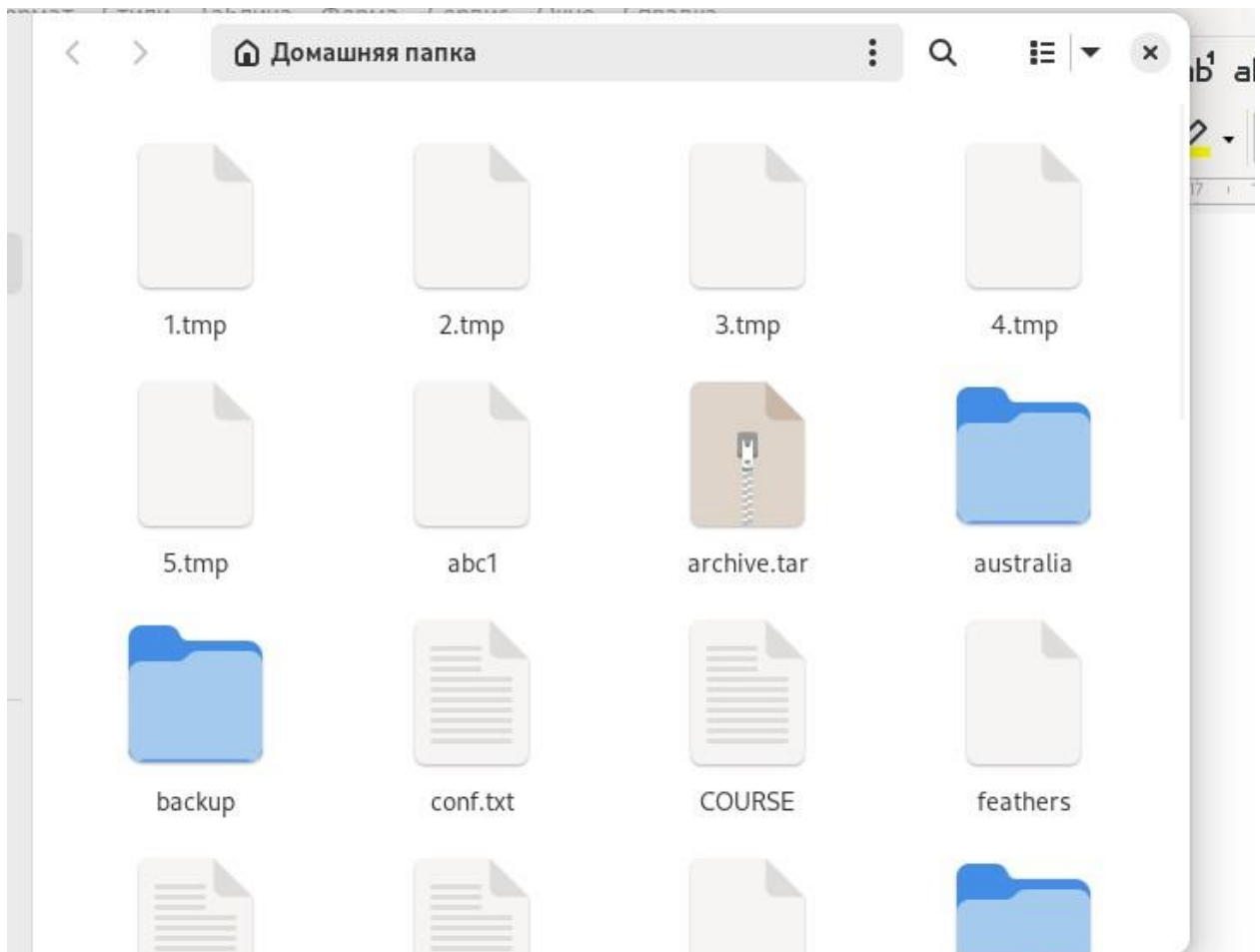


Рисунок 4.3. Созданный архив и файл.

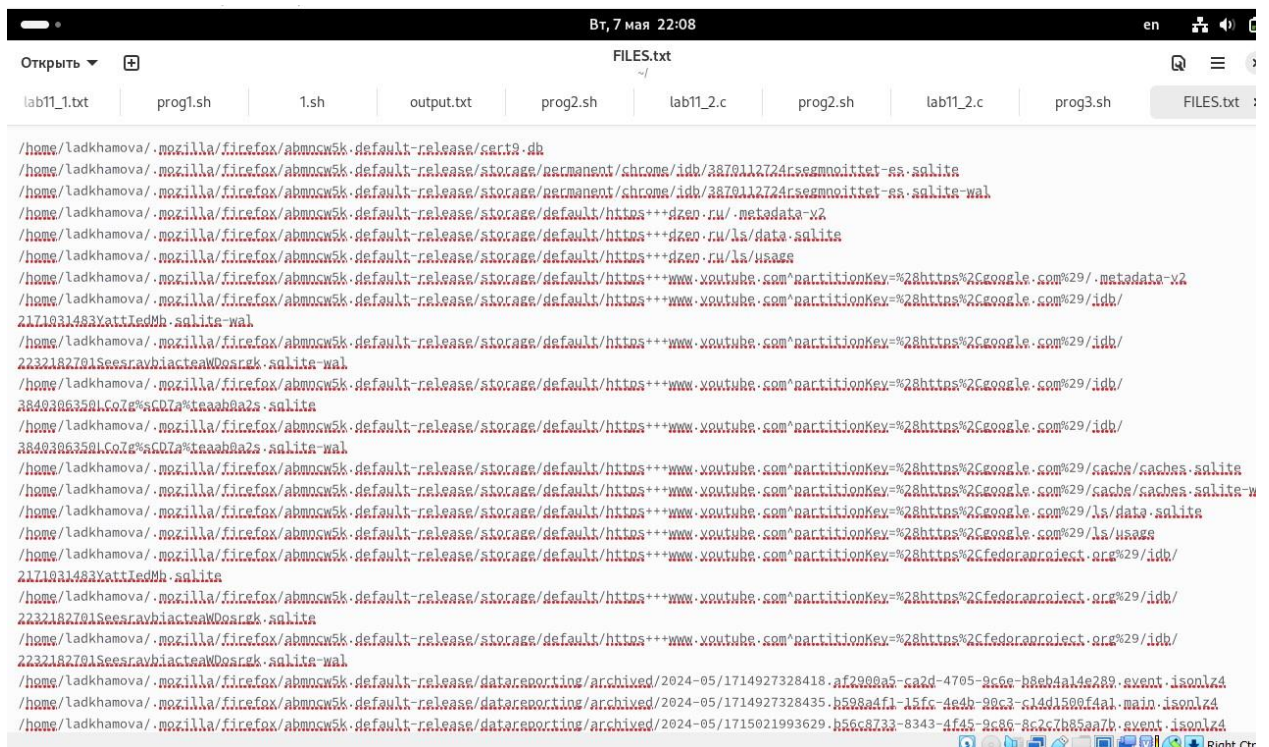


Рисунок 4.4. FILES.txt.

Вывод:

Изучила основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Ответы на контрольные вопросы:

1. Каково предназначение команды `getopts`?

Осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных. Синтаксис команды следующий: `getopts option-string variable [arg ...]` Флаги – это опции командной строки, обычно помеченные знаком минус; Например, `-F` является флагом для команды `ls -F`. Иногда эти флаги имеют аргументы, связанные с ними. Программы интерпретируют эти флаги, соответствующим образом изменяя свое поведение. Строка опций `option-string` — это список возможных букв и чисел соответствующего флага. Если ожидается, что некоторый флаг будет сопровождаться некоторым аргументом, то за этой буквой должно следовать двоеточие. Соответствующей переменной присваивается буква данной опции. Если команда `getopts` может распознать аргумент, она возвращает истину. Принято включать `getopts` в цикл `while` и анализировать введенные данные с помощью оператора `case`.

2. Какое отношение метасимволы имеют к генерации имён файлов?

При перечислении имён файлов текущего каталога можно использовать следующие символы: `-` соответствует произвольной, в том числе и пустой строке; `?` — соответствует любому одинарному символу; `[c1-c2]` — соответствует любому символу, лексикографически находящемуся между символами `c1` и `c2`. Например, `echo *` — выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды `ls`; `ls .c` — выведет все файлы с последними двумя символами, совпадающими с `.c`. `echo prog.?` — выведет все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются `prog.` `[a-z]` — соответствует произвольному имени файла в текущем каталоге, начинающемуся любой строчной буквы латинского алфавита.

3. Какие операторы управления действиями вы знаете?

Часто бывает необходимо обеспечить проведение каких-либо действий циклически и управление дальнейшими действиями в зависимости отрезультатов проверки некоторого условия. Для решения подобных задач язык программирования `bash` предоставляет возможность использовать такие управляющие конструкции, как `for`, `case`, `if` и `while`. С точки зрения командного процессора эти управляющие конструкции являются обычными командами и могут использоваться как при создании командных файлов, так и при работе в интерактивном режиме. Команды, реализующие подобные конструкции, по сути, являются операторами языка программирования `bash`. Поэтому при описании языка программирования `bash` термин оператор будет использоваться наравне с термином команда. Команды ОС

UNIX возвращают код завершения, значение которого может быть использовано для принятия решения о дальнейших действиях. Команда `test`, например, создана специально для использования в командных файлах. Единственной функцией этой команды заключается в выработке кода завершения.

4. Какие операторы используются для прерывания цикла?

Два несложных способа позволяют вам прерывать циклы в оболочке `bash`. Команда `break` завершает выполнение цикла, а команда `continue` завершает данную итерацию блока операторов. Команда `break` полезна для завершения цикла `while` в ситуациях, когда условие перестаёт быть правильным. Команда `continue` используется в ситуациях, когда больше нет необходимости выполнять блок операторов, но вы можете захотеть продолжить проверять данный блок на других условных выражениях.

5. Для чего нужны команды `false` и `true`?

Следующие две команды ОС UNIX используются только совместно с управляющими конструкциями языка программирования `bash`: это команда `true`, которая всегда возвращает код завершения, равный нулю (т.е. истина), и команда `false`, которая всегда возвращает код завершения, не равный нулю (т.е. ложь).

6. Что означает строка `if test -f mans/i.$s`, встречающаяся в командном файле?

Строка `if test -f mans/i.s`, `mans/i.s` и является ли этот файл обычным файлом. Если данный файл является каталогом, то команда вернет нулевое значение (ложь).

7. Объясните различия между конструкциями `while` и `until`.

Выполнение оператора цикла `while` сводится к тому, что сначала выполняется последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `while`, а затем, если последняя выполненная команда из этой последовательности команд возвращает нулевой код завершения (истина), выполняется последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `do`, после чего осуществляется безусловный переход на начало оператора цикла `while`. Выход из цикла будет осуществлён тогда, когда последняя выполненная команда из последовательности команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `while`, возвратит ненулевой код завершения (ложь). При замене в операторе цикла `while` служебного слова `while` на `until` условие, при выполнении которого осуществляется выход из цикла, меняется на противоположное. В остальном оператор цикла `while` и оператор цикла `until` идентичны.