

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ “РОССИЙСКИЙ УНИВЕРСИТЕТ
ДРУЖБЫ НАРОДОВ”

Факультет физико-математических и естественных наук

ОТЧЕТ

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

Выполнил: Студент группы: НПИбд-01-21 Студенческий билет:
№1032211403 ФИО студента: Матюхин Григорий Васильевич Дата
выполнения: 28.05.2022

Москва 2022

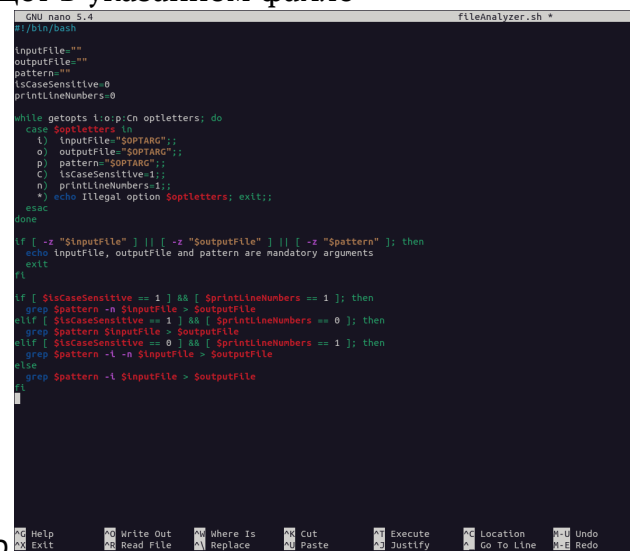
1 Цель работы:

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

2 Выполнение лабораторной работы

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

- `-i` — прочитать данные из указанного файла
- `-o` — вывести данные в указанный файл
- `-p` - шаблон — указать шаблон для поиска
- `-C` — различать большие и малые буквы
- `-n` — выдавать номера строк а затем ищет в указанном файле



```
GNU nano 5.4 fileAnalyzer.sh *
#!/bin/bash

inputFile=""
outputFile=""
pattern=""
isCaseSensitive=0
printLineNumbers=0

while getopts i:o:p:Cn optletters; do
    case $optletters in
        i) inputFile="$OPTARG";;
        o) outputFile="$OPTARG";;
        p) pattern="$OPTARG";;
        C) isCaseSensitive=1;;
        n) printLineNumbers=1;;
        *) echo "Illegal option $optletters"; exit;;
    esac
done

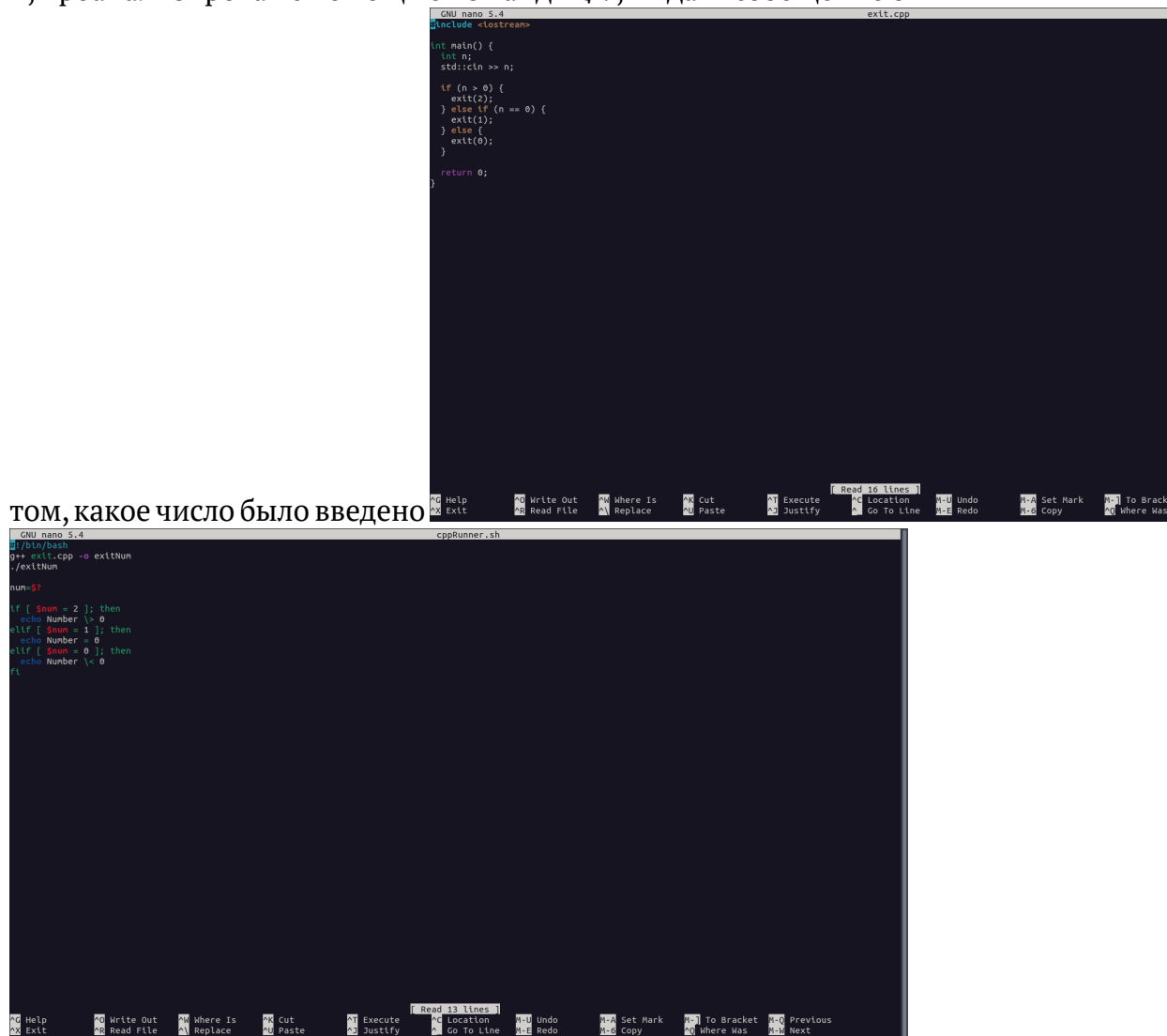
if [ -z "$inputFile" ] || [ -z "$outputFile" ] || [ -z "$pattern" ]; then
    echo "inputFile, outputFile and pattern are mandatory arguments"
    exit
fi

if [ $isCaseSensitive == 1 ] && [ $printLineNumbers == 1 ]; then
    grep $pattern -n $inputFile > $outputFile
elif [ $isCaseSensitive == 1 ] && [ $printLineNumbers == 0 ]; then
    grep $pattern $inputFile > $outputFile
elif [ $isCaseSensitive == 0 ] && [ $printLineNumbers == 1 ]; then
    grep $pattern -l -n $inputFile > $outputFile
else
    grep $pattern -l $inputFile > $outputFile
fi
```

нужные строки, определяемые ключом `-p`

2. Написать 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 коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о



The image contains two screenshots of a terminal window. The top screenshot shows the source code of a C++ program named `exit.cpp` in the `GNU nano 5.4` editor. The code defines a `main` function that reads an integer `n` from standard input and calls `exit` with `n` as an argument. The code is as follows:

```
int main() {
    int n;
    std::cin >> n;

    if (n > 0) {
        exit(2);
    } else if (n == 0) {
        exit(1);
    } else {
        exit(0);
    }

    return 0;
}
```

The bottom screenshot shows the execution of the program. The prompt is `g++ exit.cpp -o exitNum`, followed by `./exitNum`. The user enters `num=57`. The program then checks the value of `$?` and prints the number of lines read. The output is:

```
if [ $num = 2 ]; then
    echo Number \> 0
elif [ $num = 1 ]; then
    echo Number = 0
elif [ $num = 0 ]; then
    echo Number \< 0
fi
```

том, какое число было введено

3. Написать 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) fileCreator

4. Написать командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать

команду `find`)



```
GNU nano 5.4 compressor.sh
#!/bin/bash
directory=$1
find $directory -mtime -7 | tar -cf compress.tar $directory
```

3 Вывод

В ходе работы я изучил основы программирования в оболочке ОС UNIX. Научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

4 Контрольные вопросы

1. `getpods` - эта утилита анализирует аргументы команд из исполняемого файла

2. Следующие метасимволы используют для генерации имен файлов:

- `*` - любая или пустая последовательность символов
- `?` - один любой символ
- `[. . .]` - любой из символов указанных в квадратных скобках с перечислением или указанием диапазона
- `cat N*` - выдает все файлы начинающиеся с `N`
- `cat *N*` - выдает все файлы содержащие `N`
- `cat` - выдаст все файлы с однобуквенным расширением `hello.o`, `hello.c`, но не `hello.cpp`
- `program.?` - выдаст `program.com`
- `cat [a-d]*` - выдаст файлы которые начинаются с буквы `a` и заканчиваются `d`

3. Операторы управления действиями - `>` (вывод информации), `<` (ввод информации), `&` (управляет потоком исполнения команд), `&&` (запускает исполнения команды или команд в фоне), `|` (передает данные между программами), `||` (проверяет код завершения предыдущей команды)

4. Команда `break` служит для прерывания цикла и передает управление программой команде, которая идет следующей за циклом

5.

- `false` - логическое нет, отрицание, то есть дальнейшую остановку программы или переход в другую ветвь ветвления программы в зависимости от условий
- `true` - логическое да, согласие на дальнейшее исполнение

программы согласно заданным условия

6. Строка `if test -f man$s/$i.$s` означает условие для проверки существования файла `man`
7. `while` - выполняет цикл пока указанное в нем условие истинно (1, true), а `until` - выполняет цикл пока указанное в нем условие ложно (0, false)