

Презентация по лабораторной работе №13

Операционные системы

Федорова А.И

Российский университет дружбы народов, Москва, Россия

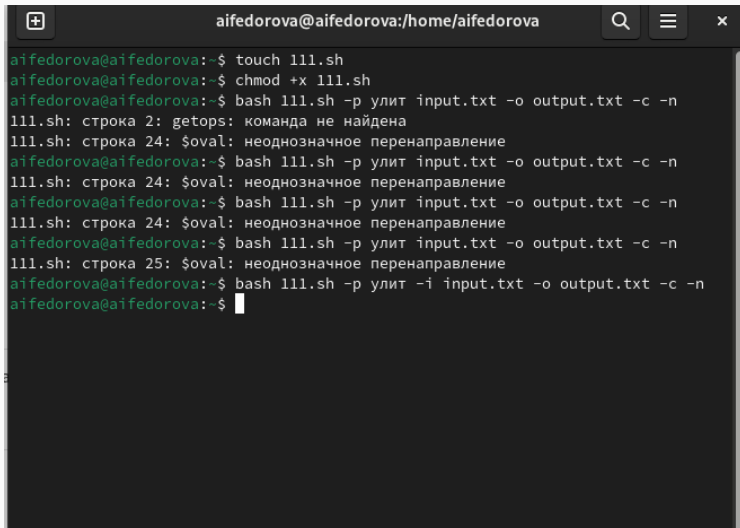
Цель данной лабораторной работы - изучить основы программирования в оболочке ОС UNIX, научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами:
 - `-i`inputfile — прочитать данные из указанного файла;
 - `-o`outputfile — вывести данные в указанный файл;
 - `-r`шаблон — 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`.
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 о коде завершения в оболочку. Команд- ный файл должен вызывать эту программу и, проанализировав с помощью

3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до ∞ (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).
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).

Выполнение работы

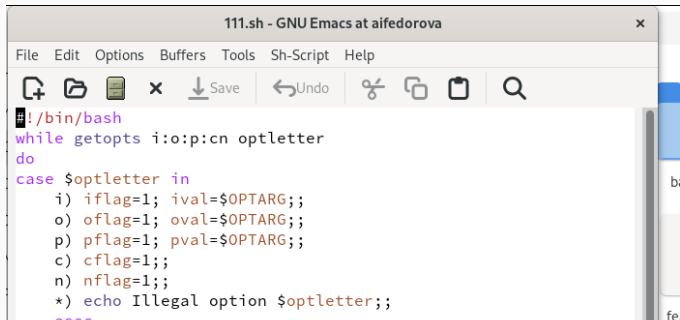
Создаю файл с разрешением на исполнение (рис.1).



```
aifedorova@aifedorova:/home/aifedorova
aifedorova@aifedorova:~$ touch 111.sh
aifedorova@aifedorova:~$ chmod +x 111.sh
aifedorova@aifedorova:~$ bash 111.sh -p улит input.txt -o output.txt -c -n
111.sh: строка 2: getops: команда не найдена
111.sh: строка 24: $oval: неоднозначное перенаправление
aifedorova@aifedorova:~$ bash 111.sh -p улит input.txt -o output.txt -c -n
111.sh: строка 24: $oval: неоднозначное перенаправление
aifedorova@aifedorova:~$ bash 111.sh -p улит input.txt -o output.txt -c -n
111.sh: строка 24: $oval: неоднозначное перенаправление
aifedorova@aifedorova:~$ bash 111.sh -p улит input.txt -o output.txt -c -n
111.sh: строка 25: $oval: неоднозначное перенаправление
aifedorova@aifedorova:~$ bash 111.sh -p улит -i input.txt -o output.txt -c -n
aifedorova@aifedorova:~$
```

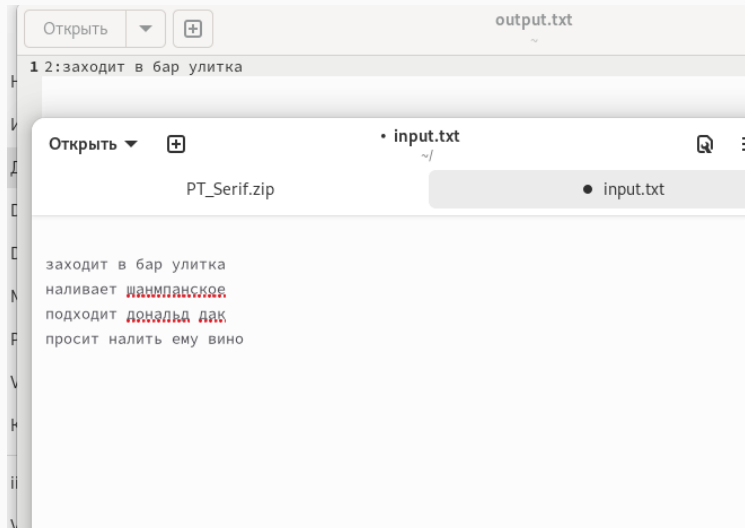
Выполнение работы

Командный файл, с командами `getopts` и `grep`, который анализирует командную строку с ключами: `-i` inputfile — прочитать данные из указанного файла; `-o` outputfile — вывести данные в указанный файл; `-r` шаблон — указать шаблон для поиска; `-C` — различать большие и малые буквы; `-n` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-p` (рис.2).

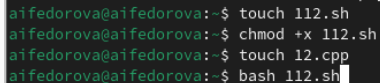


```
#!/bin/bash
while getopts i:o:p:cn optletter
do
case $optletter in
i) iflag=1; ival=$OPTARG;;
o) oflag=1; oval=$OPTARG;;
p) pflag=1; pval=$OPTARG;;
c) cflag=1;;
n) nflag=1;;
*) echo Illegal option $optletter;;
esac
done
```

Результат работы программы в файле output.txt (рис.3).



Создаю исполняемый файл для второй программы, также создаю файл 12.c для программы на Си (рис. 4).

A terminal window with a dark background and green text. It shows four lines of commands and their execution. The first line creates a file named 112.sh. The second line sets permissions for 112.sh to be executable. The third line creates a file named 12.cpp. The fourth line starts a shell for 112.sh, with a cursor visible at the end of the command.

```
aifedorova@aifedorova:~$ touch 112.sh
aifedorova@aifedorova:~$ chmod +x 112.sh
aifedorova@aifedorova:~$ touch 12.cpp
aifedorova@aifedorova:~$ bash 112.sh
```

Рис. 4: Создание файла

Выполнение работы

Пишу программу на языке Си, 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 (рис.5).

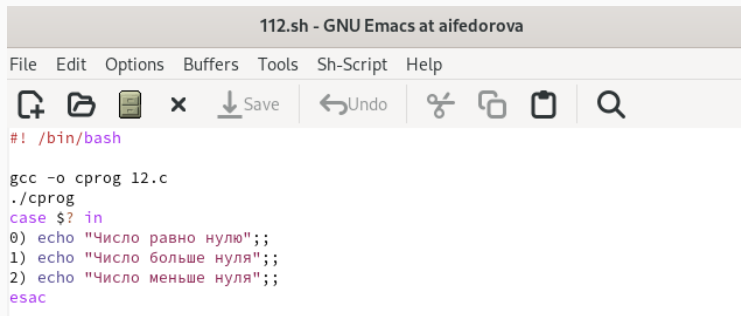


```
#include <stdlib.h>
#include <stdio.h>

int main () {
    int n
    printf ("Введите число: ");
    scanf("%d", &n);
    if (n>0) {
        exit(1);
    }
    else if (n==0) {
        exit(0);
    }
    else {
        exit(2);
    }
}
```

Выполнение работы

Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено (рис.6).

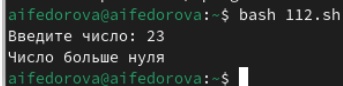


```
112.sh - GNU Emacs at aifedorova
File Edit Options Buffers Tools Sh-Script Help
[Icons: New, Open, Save, Close, Save All, Undo, Cut, Copy, Paste, Find]
#!/bin/bash

gcc -o cprog 12.c
./cprog
case $? in
0) echo "Число равно нулю";;
1) echo "Число больше нуля";;
2) echo "Число меньше нуля";;
esac
```

Рис. 6: Код программы

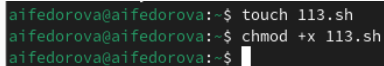
Программа работает корректно (рис. fig:007).

A terminal window with a dark background. The prompt is 'aifedorova@aifedorova:~\$'. The user has entered 'bash 112.sh'. The program outputs 'Введите число: 23' and 'Число больше нуля'. The prompt returns to 'aifedorova@aifedorova:~\$' with a cursor.

```
aifedorova@aifedorova:~$ bash 112.sh
Введите число: 23
Число больше нуля
aifedorova@aifedorova:~$
```

Рис. 7: Результат работы программы

Создаю исполняемый файл для третьей программы (рис.fig:008).

A terminal window with a dark background and green text. It shows three lines of commands being executed in a shell. The first line creates a file named 113.sh using the 'touch' command. The second line sets execute permissions for the file using 'chmod +x'. The third line shows the prompt with a cursor, indicating the command has finished.

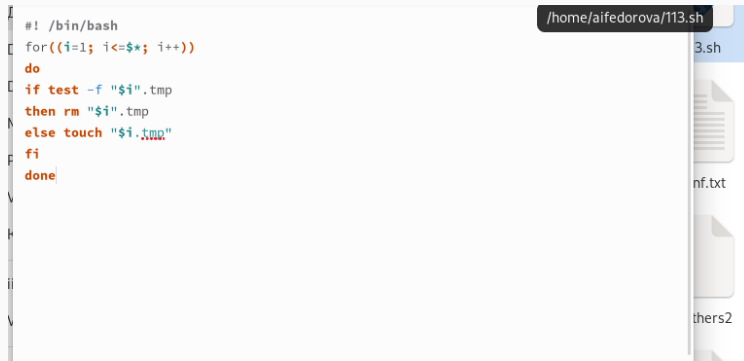
```
aifedorova@aifedorova:~$ touch 113.sh  
aifedorova@aifedorova:~$ chmod +x 113.sh  
aifedorova@aifedorova:~$
```

Рис. 8: Создание файла

Выполнение работы

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.fig:009).

```
#!/bin/bash
for((i=1; i<=$*; i++))
do
if test -f "$i".tmp
then rm "$i".tmp
else touch "$i.tmp"
fi
done
```

The image shows a terminal window with a bash script. The script is designed to create a series of files named 1.tmp through N.tmp, where N is the number of arguments passed to the script. It uses a for loop to iterate from 1 to \$*. For each iteration, it checks if a file with the current number and .tmp extension already exists using the 'test' command. If it exists, it removes the file with 'rm'. If it does not exist, it creates the file with 'touch'. The script ends with 'done'. To the right of the terminal, a file manager sidebar is visible, showing the current directory as /home/aifedorova/113.sh. Below the directory name, there are icons for files named 3.sh, nf.txt, and thers2.

Выполнение работы

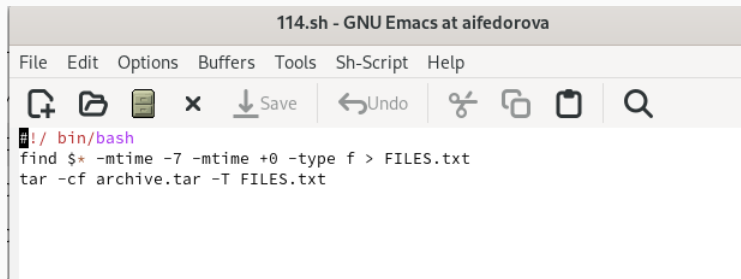
Проверяю, что программа создала файлы и удалила их при соответствующих запросах (рис. fig:010).

```
aifedorova@aifedorova:~$ bash 113.sh 4
aifedorova@aifedorova:~$ ls
'#111.sh#'  backup      my_os      prog2.sh~
111.sh      bin         newfile    prog3.sh
111.sh~     conf.txt   '#newfile3#' prog4.sh
112.sh      cprog      '#newfile4#' prog4.sh~
112.sh~     Desktop    onix        reports
'#113.sh#'  Downloads  output.txt  ski.places
113.sh      feathers   pandoc-3.1.11.1 work
12.c        feathers2  pandoc-3.1.11.1-linux-amd64.tar.gz Видео
12.c~       file.txt   pandoc-3.1.12.1 Документы
1.tmp       input.txt  pandoc-crossref Загрузки
2.tmp       lab07.sh   pandoc-crossref.1 Изображения
3.tmp       lab07.sh~  Pictures     Музыка
4.tmp       LICENSE    play         Общедоступные
abc1        may        prog1.sh     'Рабочий стол'
australia   monthly    prog2.sh     Шаблоны
aifedorova@aifedorova:~$
```

Рис. 10: Результат работы программы

Выполнение работы

Создаю исполняемый файл для четвертой программы. Это 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`) (рис. fig:011).

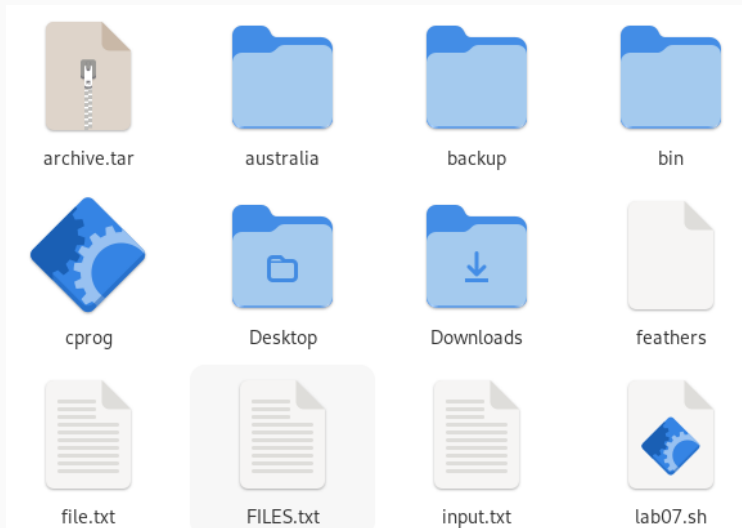


```
114.sh - GNU Emacs at aifedorova
File Edit Options Buffers Tools Sh-Script Help
[Icons: New, Open, Save, Close, Save As, Undo, Cut, Copy, Paste, Find]
#!/ bin/bash
find $* -mtime -7 -mtime +0 -type f > FILES.txt
tar -cf archive.tar -T FILES.txt
```

Рис. 11: Код программы

Выполнение работы

Проверяю работу программы (рис. fig:012).



При выполнении данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX, научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Спасибо за внимание!