

# **Лабораторная работа №13**

**Программирование в командном процессоре ОС UNIX. Ветвления и циклы**

Кижваткина Анна Юрьевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Выводы</b>	<b>14</b>

# Список иллюстраций

2.1	Создание директории . . . . .	6
2.2	Создание файла . . . . .	6
2.3	Программа . . . . .	7
2.4	Предоставление доступа . . . . .	7
2.5	Запуск программы . . . . .	8
2.6	Создание файла . . . . .	8
2.7	Программа . . . . .	9
2.8	Программа . . . . .	9
2.9	Запуск программы . . . . .	10
2.10	Создание файла . . . . .	10
2.11	Предоставление доступа . . . . .	10
2.12	Программа . . . . .	11
2.13	Запуск программы . . . . .	12
2.14	Создание файла . . . . .	12
2.15	Предоставление доступа . . . . .	12
2.16	Программа . . . . .	13
2.17	Запуск программы . . . . .	13
2.18	Запуск программы . . . . .	13

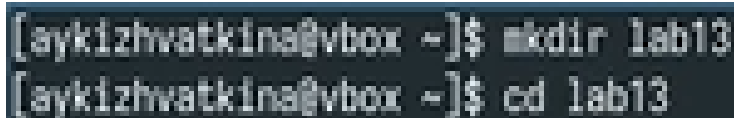
## **Список таблиц**

# 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

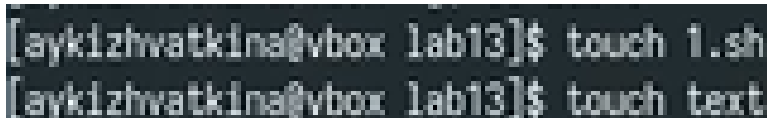
Создаем директорию lab13 и перемещаемся в неё. (рис. 2.1)



```
[aykizhvatkina@vbox ~]$ mkdir lab13  
[aykizhvatkina@vbox ~]$ cd lab13
```

Рис. 2.1: Создание директории

Создаем файл 1.sh и text. (рис. 2.2)



```
[aykizhvatkina@vbox lab13]$ touch 1.sh  
[aykizhvatkina@vbox lab13]$ touch text
```

Рис. 2.2: Создание файла

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

- -iinputfile – прочитать данные из указанного файла;
- -ooutputfile – вывести данные в указанный файл;
- -ршаблон – указать шаблон для поиска;
- -С – различать большие и малые буквы;
- -п – выдавать номера строк.

а затем ищет в указанном файле нужные строки, определяемые ключом -р. (рис. 2.3)

```
mc [aykizhvatkina@vbox]:~/lab13
1.sh [-M--] 11 L:[ 1+ 0 1/ 53] *(11 / 945b) 0010 0[*][x]
#!/bin/bash

while getopts "i:op:Cn\?" opt; do
    case $opt in
        <----->i )
            <-----> inputfile=$OPTARG
            <-----> ;;
        <----->o )
            <-----> outputfile=$OPTARG
            <-----> ;;
        <----->p )
            <-----> pattern=$OPTARG
            <-----> ;;
        <----->C )
            <-----> case_sensitive=true
            <-----> ;;
        <----->n )
            <-----> line_numbers=true
            <-----> ;;
        <----->\? )
            <-----> echo "Неверный параметр: $OPTARG" 1>&2
            <-----> exit 1
            <-----> ;;
        <----->: )
            <-----> echo "Не указан значение для параметра: $OPTARG" 1>&2
            <-----> exit 1
            <-----> ;;
    esac
done

if [ -z "$pattern" ]; then
    <-----> echo "Не указан паттерн для поиска" 1>&2
    <-----> exit 1
fi
```

Рис. 2.3: Программа

Устанавливаем право на выполнение. (рис. 2.4)

```
[aykizhvatkina@vbox lab13]$ chmod +x 1.sh
```

Рис. 2.4: Предоставление доступа

Проверяем выполнение программы. (рис. 2.5)

```

[aykizhvatkina@vbox lab13]$ ./1.sh
Не указан шаблон для поиска
[aykizhvatkina@vbox lab13]$ ./1.sh -p "one"
Не указан входной файл
[aykizhvatkina@vbox lab13]$ ./1.sh -p "one" -i text.txt
one
./1.sh: строка 52: : Нет такого файла или каталога
[aykizhvatkina@vbox lab13]$ ./1.sh -p "one" -i text.txt -n
1:one

```

Рис. 2.5: Запуск программы

Создаем файл 2.sh и 2.c и устанавливаем право на выполнение. (рис. 2.6)

```

[aykizhvatkina@vbox lab13]$ touch 2.c
[aykizhvatkina@vbox lab13]$ touch 2.sh
[aykizhvatkina@vbox lab13]$ chmod +x 2.sh

```

Рис. 2.6: Создание файла

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. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено. (рис. 2.7 и рис. 2.8)



```
mc [aykizhvatkina@vbox]:~/lab13
2.c [BM--] 1 L:[ 1+19 20/ 20] *(388 / 388b) <EOF> [*][X]
#include <stdio.h>
#include <stdlib.h>

int main() {
    int number;
    ....
    printf("Введите значение: ");
    scanf("%d", &number);
    ....
    if (number > 0) {
<-----> printf("Значение больше нуля\n");
<-----> exit(1);
    } else if (number < 0) {
<-----> printf("Значение меньше нуля\n");
<-----> exit(2);
    } else {
<-----> printf("Значение равно нулю\n");
<-----> exit(0);
}
```

Рис. 2.7: Программа

```
mc [aykizhvatkina@vbox]:~/lab13
2.sh [-M--] 3 L:[ 1+ 2 3/ 12] *(16 / 181b) 0010 0[*][X]
#!/bin/bash

./2

case $? in
    0)
<-----> echo "Значение больше нуля";;
    1)
<-----> echo "Значение меньше нуля";;
    2)
<-----> echo "Значение равно нулю";;
esac
```

Рис. 2.8: Программа

Проверяем выполнение программы. (рис. 2.9)

```
[aykizhvatkina@vbox lab13]$ ./2.sh
Введите число: 5
Число больше нуля
Число больше нуля
[aykizhvatkina@vbox lab13]$ ./2.sh
Введите число: 0
Число равно нулю
Число равно нулю
```

Рис. 2.9: Запуск программы

Создаем файл 3.sh. (рис. 2.10)

```
[aykizhvatkina@vbox lab13]$ touch 3.sh
```

Рис. 2.10: Создание файла

Устанавливаем право на выполнение. (рис. 2.11)

```
[aykizhvatkina@vbox lab13]$ chmod +x 3.sh
```

Рис. 2.11: Предоставление доступа

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). (рис. 2.12)

```

mc [aykizhvatkina@vbox]:~/lab13
3.sh [-M--] 38 L:[ 1+15 16/ 39] *(298 / 686b) 0036 0[*][X]
#!/bin/bash

create_files() {
    local count=$1
    for ((i=1; i<=count; i++)); do
        <-----> echo "fi_$i.log"
        <-----> echo "fi_$i.log"
    done
}

delete_files() {
    local count=$1
    for ((i=1; i<=count; i++)); do
        <-----> if [ -e "fi_$i.log" ]; then
            <-----> rm "fi_$i.log"
            <-----> echo "fi_$i.log"
        <-----> fi
    done
}

if [ $# -eq 0 ]; then
    <-----> echo "No passed arguments, please use program"
    exit 1
fi

action=$1

case $action in
    create)
        <-----> create_files $2
        <-----> ;;
    delete)
        <-----> delete_files $2

```

Рис. 2.12: Программа

Проверяем выполнение программы. (рис. 2.13)

```

[aykizhvatkina@vbox lab13]$ ./3.sh create 5
Создан файл 1.tmp
Создан файл 2.tmp
Создан файл 3.tmp
Создан файл 4.tmp
Создан файл 5.tmp
[aykizhvatkina@vbox lab13]$ m
bash: m: команда не найдена
[aykizhvatkina@vbox lab13]$ mc

[aykizhvatkina@vbox lab13]$ ./3.sh delete 5
[aykizhvatkina@vbox lab13]$ mc

[aykizhvatkina@vbox lab13]$ ./3.sh delete 5
[aykizhvatkina@vbox lab13]$ mc

[aykizhvatkina@vbox lab13]$ ./3.sh delete 5
Удален файл 1.tmp
Удален файл 2.tmp
Удален файл 3.tmp
Удален файл 4.tmp
Удален файл 5.tmp
[aykizhvatkina@vbox lab13]$

```

Рис. 2.13: Запуск программы

Создаем файл 4.sh. (рис. 2.14)

```

[aykizhvatkina@vbox lab13]$ touch 4.sh

```

Рис. 2.14: Создание файла

Устанавливаем право на выполнение. (рис. 2.15)

```

[aykizhvatkina@vbox lab13]$ chmod +x 4.sh

```

Рис. 2.15: Предоставление доступа

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). (рис. 2.16)

```

mc [aykizhvatkina@vbox]:~/lab13
4.sh [BM--] 0 L:[ 1+ 0 1/ 18] *(0 / 526b) 0035 0[*][X]
~/bin/bash

directory=$1
output_archive="archive.tar.gz"
threshold_days=7

if [ -z "$directory" ]; then
    echo "Укажите директорию в качестве аргумента"
    exit 1
fi

if [ ! -d "$directory" ]; then
    echo "Указанная директория не существует"
    exit 1
fi

find "$directory" -type f -mtime -$threshold_days -print0 | tar --null -c
echo "Архивация завершена. Архив создан: $output_archive"

```

Рис. 2.16: Программа

Проверяем выполнение программы. (рис. 2.17 и рис. 2.18)

```

[aykizhvatkina@vbox lab13]$ ./4.sh .
Архивация завершена. Архив создан: archive.tar.gz

```

Рис. 2.17: Запуск программы

Левая панель				Правая панель			
Имя	Размер	Дата правки	Команда	Имя	Размер	Дата правки	Команда
../	-BBERX-	мая 10 15:43	..	../	-BBERX-	апр 18 21:57	..
text.txt	23	мая 10 14:56		1.jpg	45230	апр 18 20:23	
text	23	мая 10 14:32					
archiv-tar.gz	4023	мая 10 15:44					
4.sh	526	мая 10 15:43					

Рис. 2.18: Запуск программы

## **3 Выводы**

С помощью данной лабораторной работы мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.