

# **Отчет по лабораторной работе №13**

**Дисциплина: Операционные системы**

Беспутин Глеб Антонович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>13</b>
	<b>Список литературы</b>	<b>14</b>

# Список иллюстраций

4.1	.....	8
4.2	.....	9
4.3	.....	10
4.4	.....	10
4.5	.....	11
4.6	.....	12
4.7	.....	12

## 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 Задание

1. 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; `-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 `-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 о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до  $\infty$  (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 `tag` 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`)

### 3 Теоретическое введение

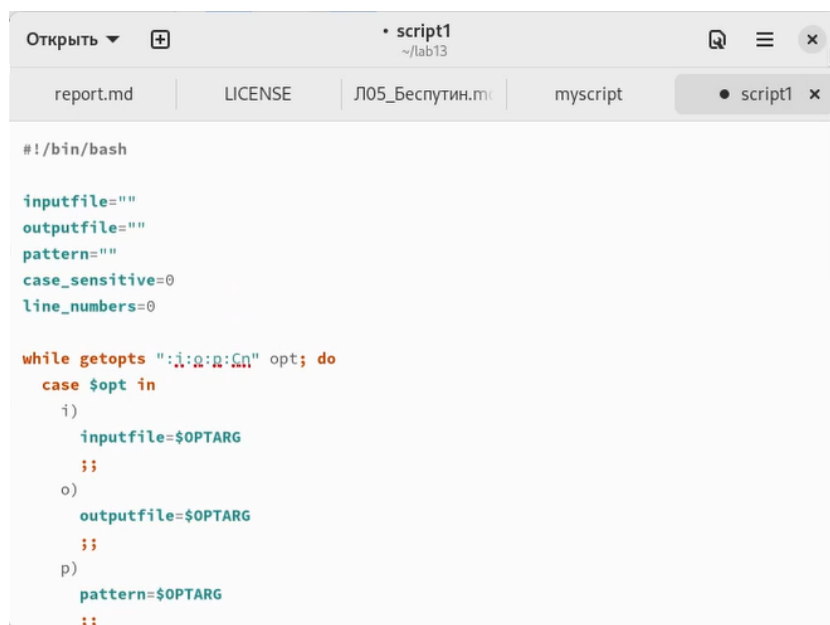
1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами: `-iinputfile` — прочитать данные из указанного файла; `-ooutputfile` — вывести данные в указанный файл; `-rшаблон` — указать шаблон для поиска; `-C` — различать большие и малые буквы; `-n` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-r`.
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. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до  $\infty$  (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 `tag` 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`)

## 4 Выполнение лабораторной работы

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;
- `-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. [4.1]).



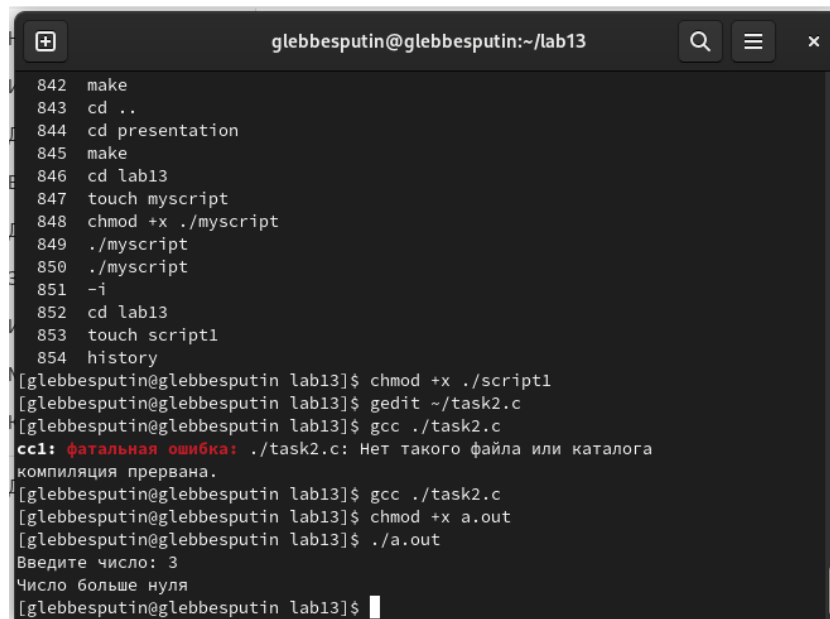
```
#!/bin/bash

inputfile=""
outputfile=""
pattern=""
case_sensitive=0
line_numbers=0

while getopts ":i:o:r:Cn" opt; do
  case $opt in
    i)
      inputfile=$OPTARG
      ;;
    o)
      outputfile=$OPTARG
      ;;
    r)
      pattern=$OPTARG
      ;;
  esac
done
```

Рис. 4.1:

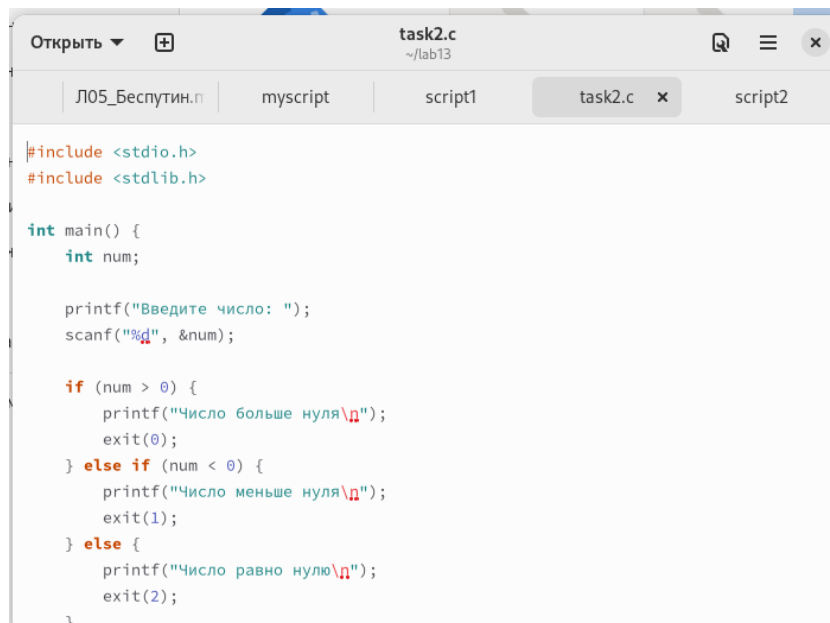




```
842 make
843 cd ..
844 cd presentation
845 make
846 cd lab13
847 touch myscript
848 chmod +x ./myscript
849 ./myscript
850 ./myscript
851 -i
852 cd lab13
853 touch script1
854 history
[glebbesputin@glebbesputin lab13]$ chmod +x ./script1
[glebbesputin@glebbesputin lab13]$ gedit ~/task2.c
[glebbesputin@glebbesputin lab13]$ gcc ./task2.c
cc1: фатальная ошибка: ./task2.c: Нет такого файла или каталога
компиляция прервана.
[glebbesputin@glebbesputin lab13]$ gcc ./task2.c
[glebbesputin@glebbesputin lab13]$ chmod +x a.out
[glebbesputin@glebbesputin lab13]$ ./a.out
Введите число: 3
Число больше нуля
[glebbesputin@glebbesputin lab13]$
```

Рис. 4.2:

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. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.(рис. [4.3]).



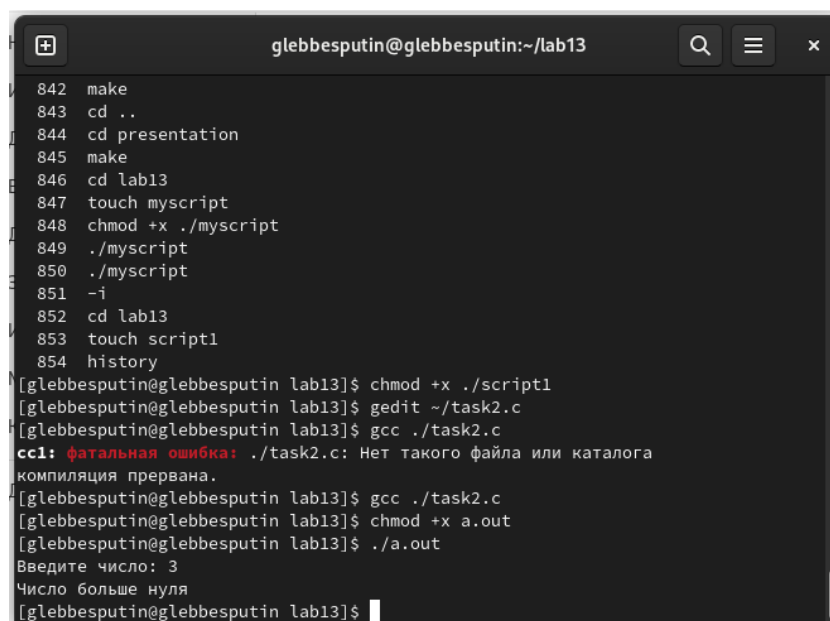
```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int num;

    printf("Введите число: ");
    scanf("%d", &num);

    if (num > 0) {
        printf("Число больше нуля\n");
        exit(0);
    } else if (num < 0) {
        printf("Число меньше нуля\n");
        exit(1);
    } else {
        printf("Число равно нулю\n");
        exit(2);
    }
}
```

Рис. 4.3:



```
842 make
843 cd ..
844 cd presentation
845 make
846 cd lab13
847 touch myscript
848 chmod +x ./myscript
849 ./myscript
850 ./myscript
851 -i
852 cd lab13
853 touch script1
854 history
[glebbesputin@glebbesputin lab13]$ chmod +x ./script1
[glebbesputin@glebbesputin lab13]$ gedit ~/task2.c
[glebbesputin@glebbesputin lab13]$ gcc ./task2.c
cc1: фатальная ошибка: ./task2.c: Нет такого файла или каталога
компиляция прервана.
[glebbesputin@glebbesputin lab13]$ gcc ./task2.c
[glebbesputin@glebbesputin lab13]$ chmod +x a.out
[glebbesputin@glebbesputin lab13]$ ./a.out
Введите число: 3
Число больше нуля
[glebbesputin@glebbesputin lab13]$
```

Рис. 4.4:

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)(рис. [4.5]).



```
#!/bin/bash

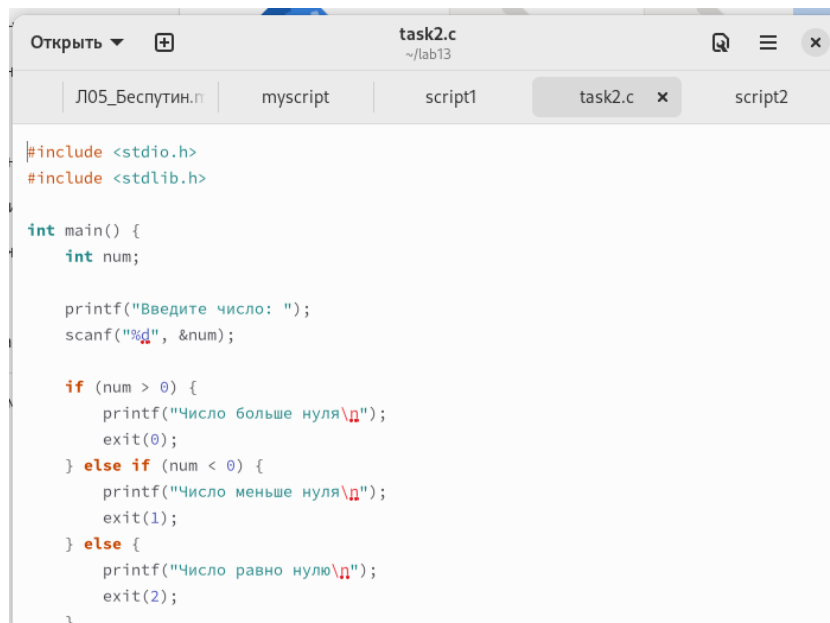
if [ $# -eq 0 ]; then
    echo "Usage: ./script N"
    exit 1
fi

N=$1
for ((i=1; i<=N; i++)); do
    touch "$i.tmp"
    echo "Created file $i.tmp"
done

read -p "Хотите удалить файлы? (y/n): " answer
if [ "$answer" = "y" ]; then
    for ((i=1; i<=N; i++)); do
        rm "$i.tmp"
        echo "Deleted file $i.tmp"
    done
fi
```

Рис. 4.5:

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. [4.6]).



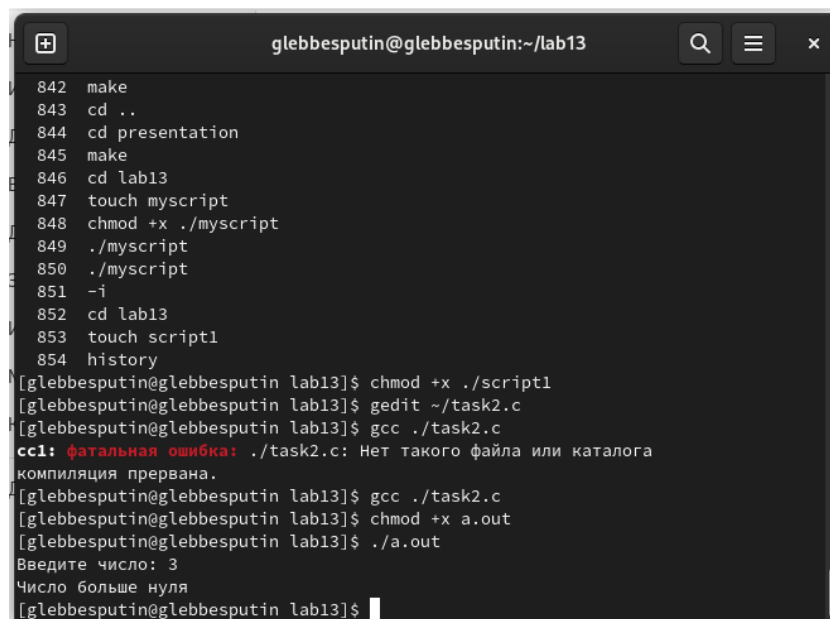
```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int num;

    printf("Введите число: ");
    scanf("%d", &num);

    if (num > 0) {
        printf("Число больше нуля\n");
        exit(0);
    } else if (num < 0) {
        printf("Число меньше нуля\n");
        exit(1);
    } else {
        printf("Число равно нулю\n");
        exit(2);
    }
}
```

Рис. 4.6:



```
glebbesputin@glebbesputin:~/lab13
842 make
843 cd ..
844 cd presentation
845 make
846 cd lab13
847 touch myscript
848 chmod +x ./myscript
849 ./myscript
850 ./myscript
851 -i
852 cd lab13
853 touch script1
854 history
[glebbesputin@glebbesputin lab13]$ chmod +x ./script1
[glebbesputin@glebbesputin lab13]$ gedit ~/task2.c
[glebbesputin@glebbesputin lab13]$ gcc ./task2.c
cc1: фатальная ошибка: ./task2.c: Нет такого файла или каталога
компиляция прервана.
[glebbesputin@glebbesputin lab13]$ gcc ./task2.c
[glebbesputin@glebbesputin lab13]$ chmod +x a.out
[glebbesputin@glebbesputin lab13]$ ./a.out
Введите число: 3
Число больше нуля
[glebbesputin@glebbesputin lab13]$
```

Рис. 4.7:

## 5 Выводы

Я изучил основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

## **Список литературы**