

# **Презентация к лабораторной работе 12**

**Дисциплина: Операционные системы**

**Куликов Максим Игоревич**

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Выводы</b>	<b>13</b>
<b>5</b>	<b>Контрольные вопросы</b>	<b>14</b>

## Список таблиц

## Список иллюстраций

3.1	Первый скрипт . . . . .	7
3.2	Тест №1 . . . . .	8
3.3	Создание с-файла . . . . .	8
3.4	Второй скрипт . . . . .	9
3.5	Тест №2 . . . . .	9
3.6	Скрипт №3 . . . . .	10
3.7	Тест №3 . . . . .	10
3.8	Тест №3 . . . . .	11
3.9	Тест №4 . . . . .	12

# 1 Цель работы

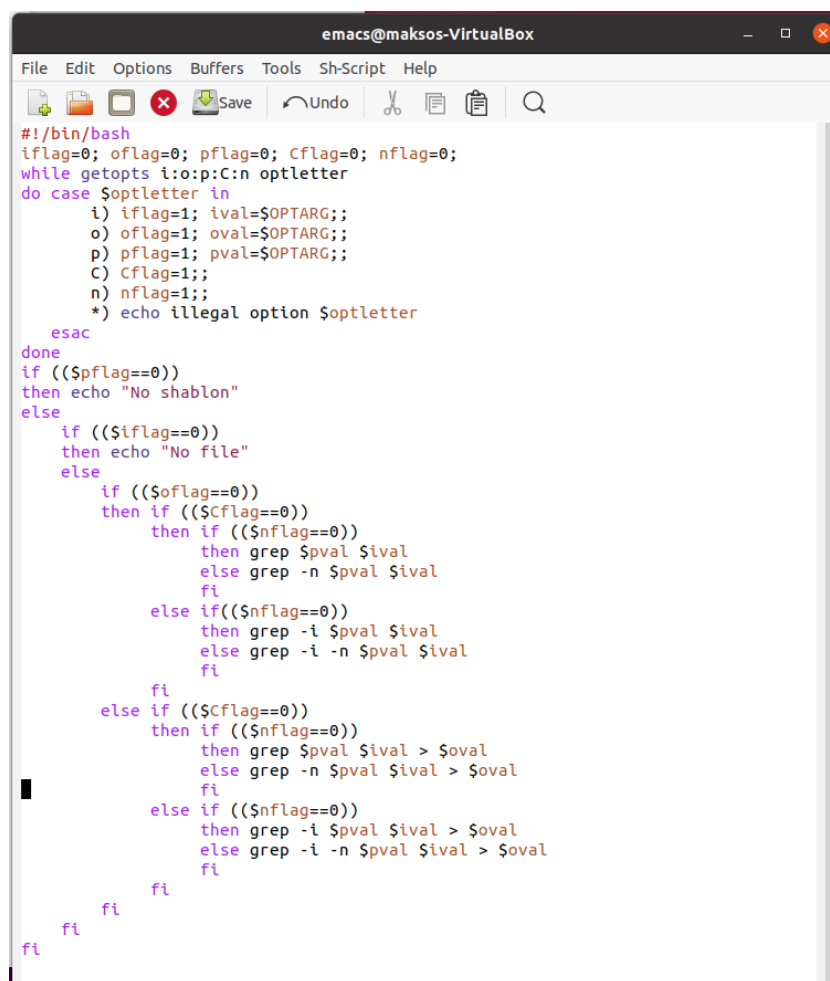
Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы

## **2 Задание**

1. Ознакомиться с теоретическим материалом.
2. Выполнить работу.

### 3 Выполнение лабораторной работы

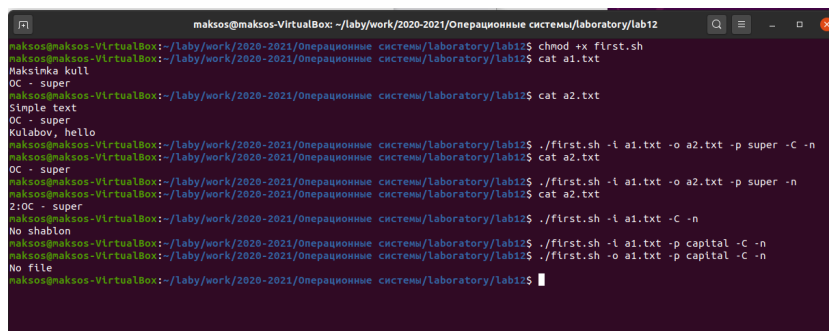
1. Создаю файл с расширением “sh”, в котором пишу скрипт, который анализирует командную строку (рис. -fig. 3.1)



```
#!/bin/bash
iflag=0; oflag=0; pflag=0; Cflag=0; nflag=0;
while getopts i:o:p:C:n 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 $optletter
    esac
done
if (($pflag==0))
then echo "No shablon"
else
if (($iflag==0))
then echo "No file"
else
if (($oflag==0))
then if (($Cflag==0))
then if (($nflag==0))
then grep $pval $ival
else grep -n $pval $ival
fi
else if (($nflag==0))
then grep -i $pval $ival
else grep -i -n $pval $ival
fi
fi
else if (($Cflag==0))
then if (($nflag==0))
then grep $pval $ival > $oval
else grep -n $pval $ival > $oval
fi
else if (($nflag==0))
then grep -i $pval $ival > $oval
else grep -i -n $pval $ival > $oval
fi
fi
fi
fi
```

Рис. 3.1: Первый скрипт

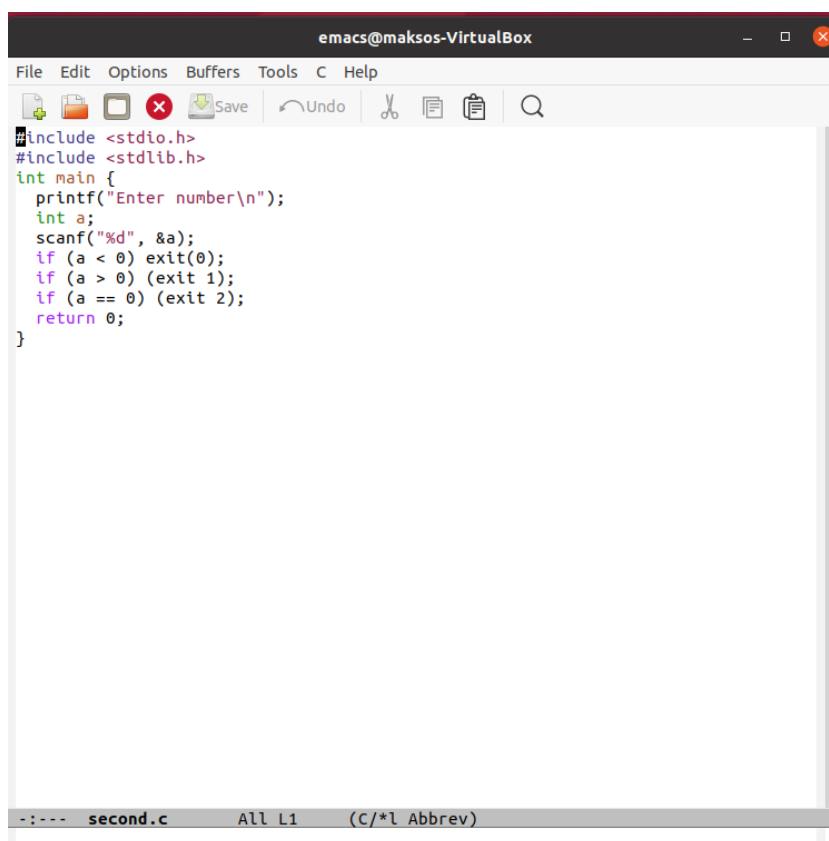
2. Тест скрипта. (рис. -fig. 3.2)

A terminal window titled 'maksos@maksos-VirtualBox' showing a series of commands and their outputs. The user is in the directory ~/laby/work/2020-2021/Операционные системы/laboratory/lab12. The commands include: 'chmod +x first.sh', 'cat a1.txt' (output: 'Maksinka kull'), 'cat a2.txt' (output: 'OC - super', 'Simple text', 'Kulabov, hello'), and several runs of './first.sh' with different arguments like '-i a1.txt -o a2.txt -p super -C -n', '-i a1.txt -o a2.txt -p super -n', '-i a1.txt -C -n', '-i a1.txt -p capital -C -n', and '-o a1.txt -p capital -C -n'. The outputs show the script processing the files and arguments.

```
maksos@maksos-VirtualBox: ~/laby/work/2020-2021/Операционные системы/laboratory/lab12
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ chmod +x first.sh
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ cat a1.txt
Maksinka kull
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ cat a2.txt
OC - super
Simple text
Kulabov, hello
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./first.sh -i a1.txt -o a2.txt -p super -C -n
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ cat a2.txt
OC - super
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./first.sh -i a1.txt -o a2.txt -p super -n
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ cat a2.txt
2:OC - super
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./first.sh -i a1.txt -C -n
No shablon
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./first.sh -i a1.txt -p capital -C -n
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./first.sh -o a1.txt -p capital -C -n
No file
maksos@maksos-VirtualBox:~/laby/work/2020-2021/Операционные системы/laboratory/lab12$
```

Рис. 3.2: Тест №1

3. Создаю файл с расширением “с”. Открываю его через emacs. Программа анализирует входящее значение. (рис. -fig. 3.3)

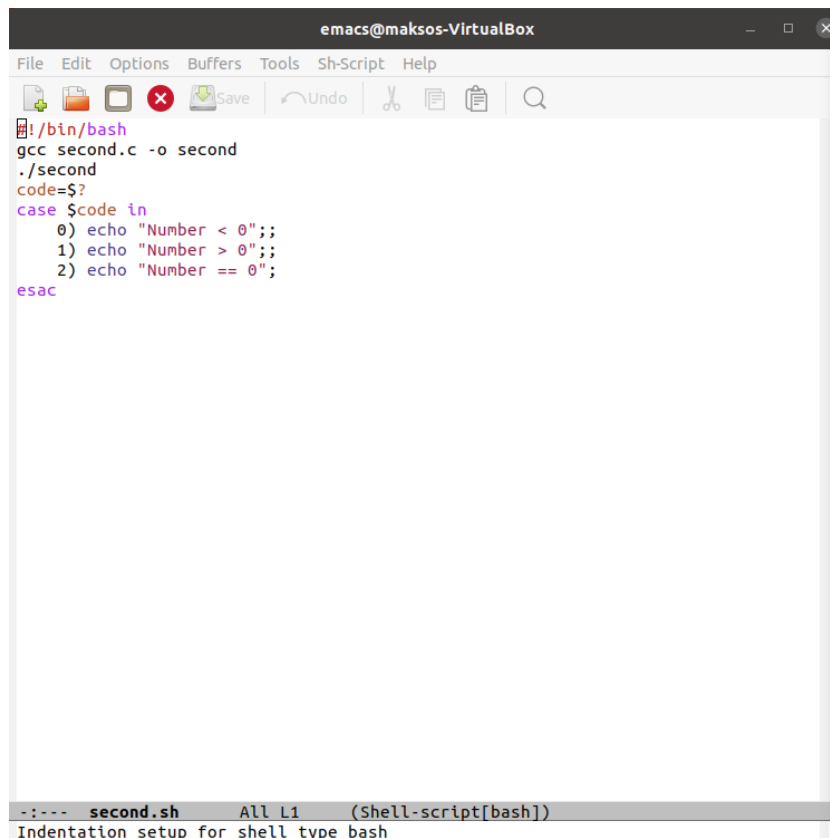
An Emacs editor window titled 'emacs@maksos-VirtualBox' showing a C program named 'second.c'. The code includes <stdio.h> and <stdlib.h>, and has a main function that prompts the user to 'Enter number\n', reads an integer 'a', and then checks if 'a' is less than 0 (exit 0), greater than 0 (exit 1), or equal to 0 (exit 2), before returning 0. The status bar at the bottom shows '-:--- second.c All L1 (C/\*l Abbrev)'.

```
emacs@maksos-VirtualBox
File Edit Options Buffers Tools C Help
[Icons] Save Undo [Icons]
#include <stdio.h>
#include <stdlib.h>
int main {
    printf("Enter number\n");
    int a;
    scanf("%d", &a);
    if (a < 0) exit(0);
    if (a > 0) (exit 1);
    if (a == 0) (exit 2);
    return 0;
}
-:--- second.c All L1 (C/*l Abbrev)
```

Рис. 3.3: Создание с-файла

5. Создаю файл с расширением “sh”, в котором пишу скрипт, который работает с с-файлом, который написал ранее. (рис. -fig. 3.4)

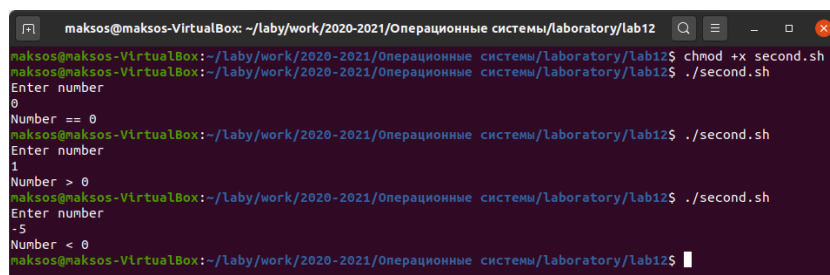




```
#!/bin/bash
gcc second.c -o second
./second
code=$?
case $code in
  0) echo "Number < 0";;
  1) echo "Number > 0";;
  2) echo "Number == 0";;
esac
```

Рис. 3.4: Второй скрипт

7. Тестирую скрипт №2 (рис. -fig. 3.5)



```
maksos@maksos-VirtualBox: ~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ chmod +x second.sh
maksos@maksos-VirtualBox: ~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./second.sh
Enter number
0
Number == 0
maksos@maksos-VirtualBox: ~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./second.sh
Enter number
1
Number > 0
maksos@maksos-VirtualBox: ~/laby/work/2020-2021/Операционные системы/laboratory/lab12$ ./second.sh
Enter number
-5
Number < 0
maksos@maksos-VirtualBox: ~/laby/work/2020-2021/Операционные системы/laboratory/lab12$
```

Рис. 3.5: Тест №2

8. Текст скрипта №3. Он создаёт файлы и удаляет их по запросу (рис. -fig. ??)

```

#!/bin/bash
opt=$1;
format=$2;
number=$3;
function Files() {
    for ((i=1; i<=$number; i++)) do
        file=$(echo $format | tr '#' 'i')
        if [ $opt == "-r" ]
        then
            rm -f $file
        elif [ $opt == "-c" ]
        then
            touch $file
        fi
    done
}
Files

```

Рис. 3.6: Скрипт №3

9. Проверяю работоспособность скрипта №3 (рис. -fig. ??)

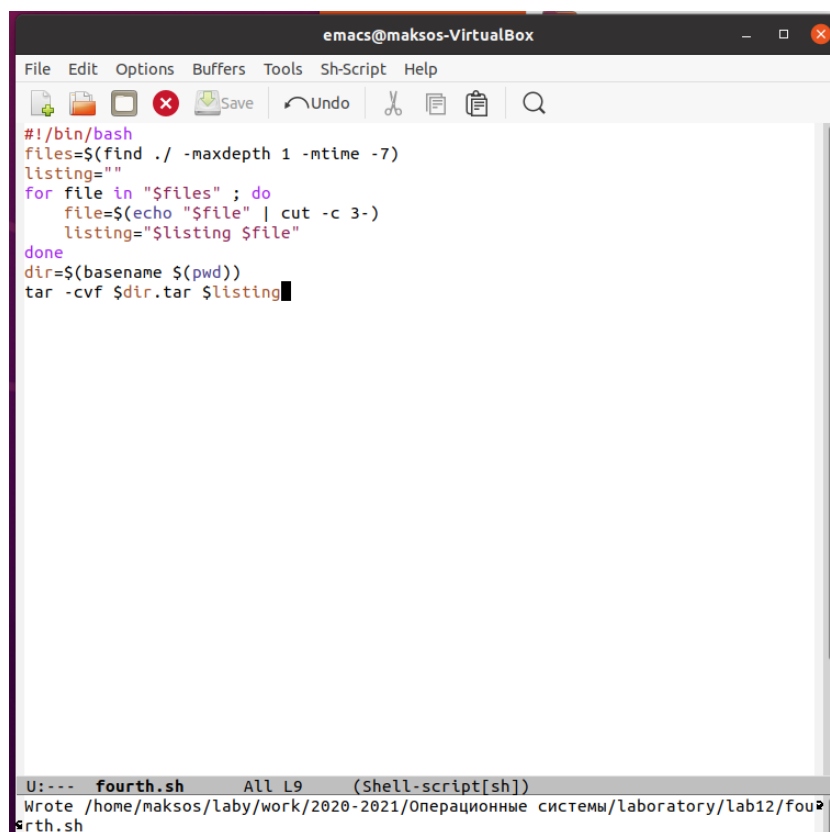
```

maksos@maksos-VirtualBox: ~/lab/work/2020-2021/Операционные системы/laboratory/lab12$ chmod +x third.sh
maksos@maksos-VirtualBox: ~/lab/work/2020-2021/Операционные системы/laboratory/lab12$ ./third.sh -c kul#.txt 5
first.sh  kul1.txt  kul4.txt  presentation12.md  second.c  second.sh-  third.sh-
image    kul3.txt  Makefile  second          second.c-  third
maksos@maksos-VirtualBox: ~/lab/work/2020-2021/Операционные системы/laboratory/lab12$ ./third.sh -r kul#.txt 5
first.sh  image  presentation12.md  second  second.c-  second.sh-  third.sh
first.sh- Makefile  report12.md      second.c  second.sh  third      third.sh-
maksos@maksos-VirtualBox: ~/lab/work/2020-2021/Операционные системы/laboratory/lab12$

```

Рис. 3.7: Тест №3

10. Скрипт №4. Она создаёт архив из файлов, хранящихся в каталоге. (рис. -fig. 3.5)



```
#!/bin/bash
files=$(find ./ -maxdepth 1 -mtime -7)
listing=""
for file in "$files" ; do
    file=$(echo "$file" | cut -c 3-)
    listing="$listing $file"
done
dir=$(basename $(pwd))
tar -cvf $dir.tar $listing
```

U:--- **fourth.sh** All L9 (Shell-script[sh])  
Wrote /home/maksos/laby/work/2020-2021/Операционные системы/laboratory/lab12/fourth.sh

Рис. 3.8: Тест №3

11. Тестирую скрипт №4. (рис. -fig. ??)

```
maksos@maksos-VirtualBox:~/Catalog1$ ls -l
итого 16
-rwxrwxr-x 1 maksos maksos 907 мая 25 22:42 first.sh
-rwxrwxr-x 1 maksos maksos 218 мая 25 23:58 fourth.sh
-rwxrwxr-x 1 maksos maksos 152 мая 25 23:20 second.sh
-rwxrwxr-x 1 maksos maksos 244 мая 25 23:40 third.sh
maksos@maksos-VirtualBox:~/Catalog1$ sudo ~/fourth.sh
third.sh
second.sh
fourth.sh
first.sh
maksos@maksos-VirtualBox:~/Catalog1$ tar -tf Catalog1.tar
third.sh
second.sh
fourth.sh
first.sh
maksos@maksos-VirtualBox:~/Catalog1$
```

Рис. 3.9: Тест №4

## 4 Выводы

Изучил основы программирования в оболочке ОС UNIX/Linux. Научился писать небольшие командные файлы.

## 5 Контрольные вопросы

1. есьма необходимой при программировании является команда `getopts`, которая принимает `string variable [arg ... ]`. Флаги – это опции командной строки, обычно начинающиеся с `-`. Например, `-F` является флагом для команды `ls -F`. Иногда эти флаги имеют аргументы. Например, `-s` – это список возможных букв и чисел соответствующего флага. Например, `cat ifile_in.txt -ofile_out.doc -L -t -r` Вот как выглядит использование `getopts`.
2. При перечислении имен файлов текущего каталога можно использовать следующие символы:
  - `*` – соответствует произвольной, в том числе и пустой строке;
  - `?` – соответствует любому одному символу;
  - `[c1-c1]` – соответствует любому символу, лексикографически находящемуся между символами `c1` и `c2`.
  - `echo *` – выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды `ls`;
  - `ls .c` – выведет все файлы с последними двумя символами, равными `.c`.
  - `echo prog.?` – выдаст все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются `prog.`
  - `[a-z]` – соответствует произвольному имени файла в текущем каталоге, начинающемуся с любой строчной буквы латинского алфавита.
3. Часто бывает необходимо обеспечить проведение каких-либо действий циклически и управление дальнейшими действиями в зависимости от результатов проверки некоторого условия. Для решения подобных задач язык программирования `bash` предоставляет Вам 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 дела являются операторами языка программирования `bash`. Поэтому при описании языка программирования `bash` термин оператор будет использоваться наравне с термином команда.

4. 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 перестает быть правильным. Пример бесконечного цикла `while`, с прерыванием в момент, когда файл перестает существовать: `while true do if [! -f $file] then break fi sleep 10 done`
5. Команды ОС UNIX возвращают код завершения, значение которого может быть использовано для принятия решения о дальнейших действиях. Команда `test`, например, создана специально для использования в командных файлах. Единственная функция этой команды заключается в выработке кода завершения.
6. Введенная строка означает условие существования файла `mans/i.$s`
7. Если речь идет о 2-х параллельных действиях, то это `while`. когда мы показываем, что сначала делается 1-е действие. потом оно заканчивается при наступлении 2-го действия, применяем `until`.