

Лабораторная работа №10

Программирование в командном процессоре ОС UNIX. Командные файлы

Рябцев И.В.; НКАбд-03-22

1 Цель работы

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

2 Задание

- Написать скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в вашем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar. Способ использования команд архивации необходимо узнать, изучив справку.
- Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов.
- Написать командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.
- Написать командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки.

3 Теоретическое введение

Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек: - оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;

- С-оболочка (или `csch`) — надстройка на оболочкой Борна, использующая С-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
- оболочка Корна (или `ksh`) — напоминает оболочку С, но операторы управления программой совместимы с операторами оболочки Борна;
- BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек С и Корна (разработка компании Free Software Foundation).

POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ. Стандарты POSIX разработаны комитетом IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости различных UNIX/Linux-подобных операционных систем и переносимости прикладных программ на уровне исходного кода. POSIX-совместимые оболочки разработаны на базе оболочки Корна. Рассмотрим основные элементы программирования в оболочке `bash`. В других оболочках большинство команд будет совпадать с описанными ниже. [1]

4 Выполнение лабораторной работы

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 1)

- Написать скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию `backur` в вашем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор `zip`, `bzip2` или `tar`. Способ использования команд архивации необходимо узнать, изучив справку. рис. (1-3)

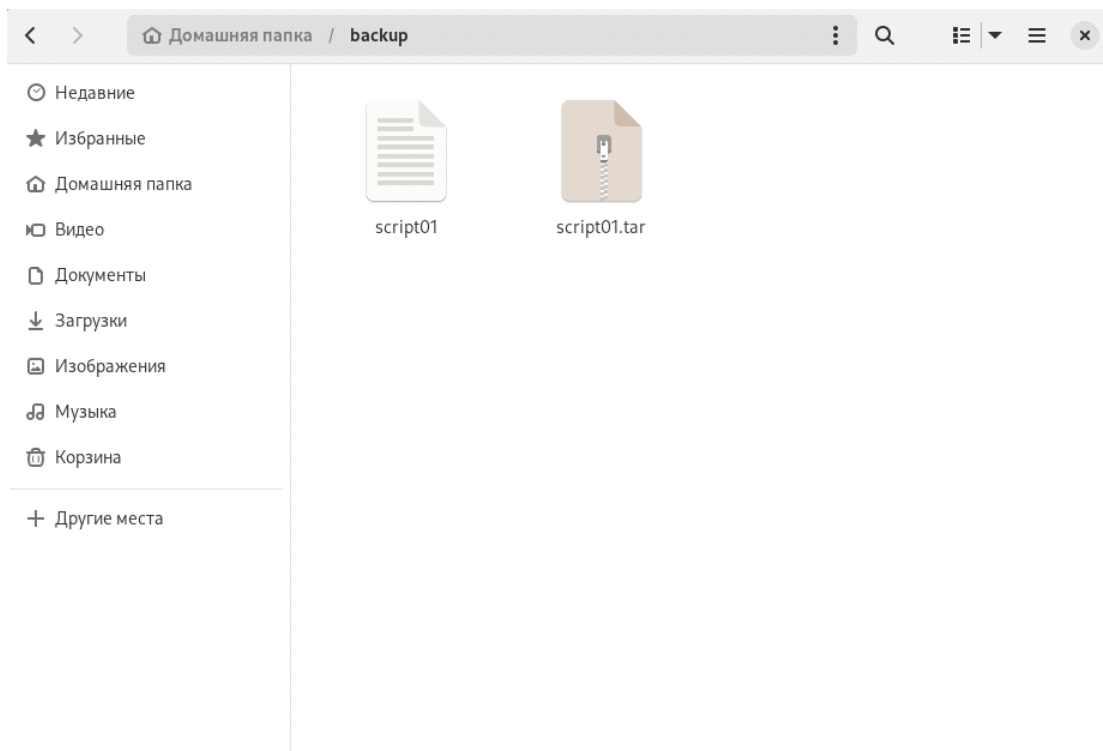


Рис. 1: Текст первой программы

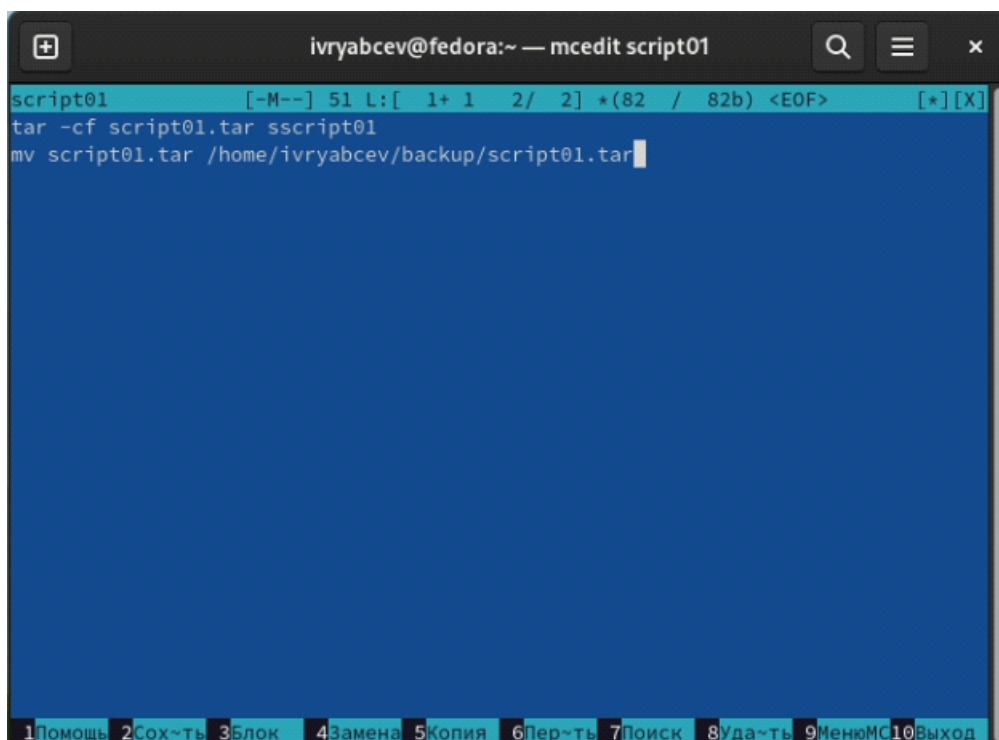
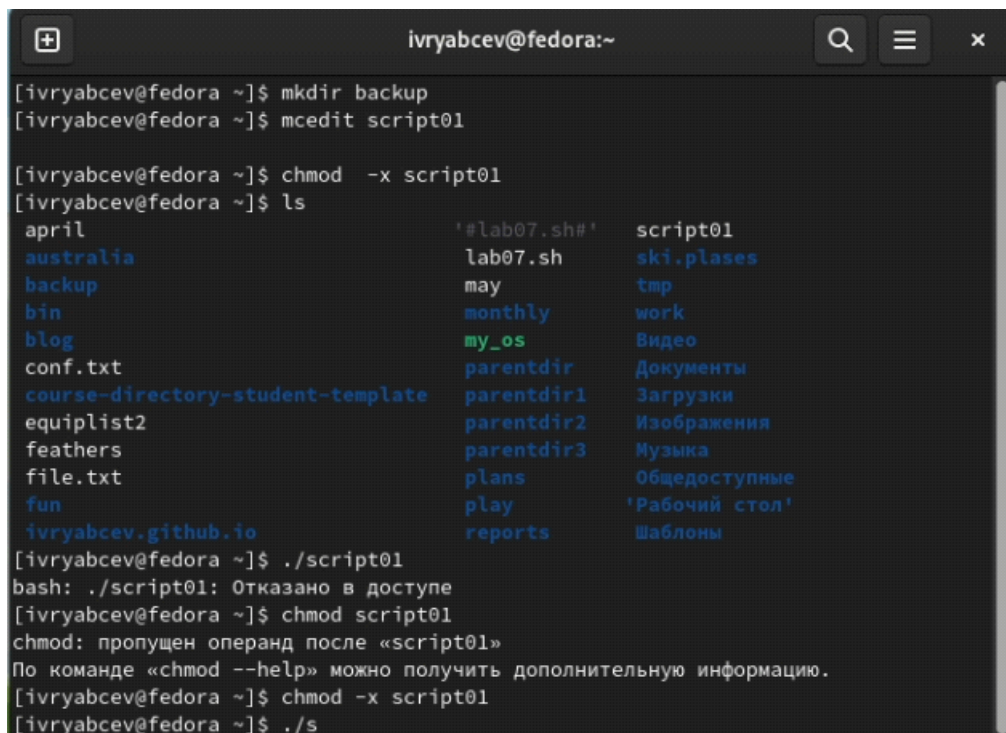


Рис. 2: Проверка работы программы



```
ivryabcev@fedora:~  
[ivryabcev@fedora ~]$ mkdir backup  
[ivryabcev@fedora ~]$ mcedit script01  
  
[ivryabcev@fedora ~]$ chmod -x script01  
[ivryabcev@fedora ~]$ ls  
april                                '#lab07.sh#'      script01  
australia                           lab07.sh          ski.places  
backup                              may              tmp  
bin                                 monthly          work  
blog                               my_os            Видео  
conf.txt                          parentdir        Документы  
course-directory-student-template parentdir1       Загрузки  
equiplist2                        parentdir2       Изображения  
feathers                          parentdir3       Музыка  
file.txt                          plans            Общедоступные  
fun                               play             'Рабочий стол'  
ivryabcev.github.io              reports          Шаблоны  
  
[ivryabcev@fedora ~]$ ./script01  
bash: ./script01: Отказано в доступе  
[ivryabcev@fedora ~]$ chmod script01  
chmod: пропущен операнд после «script01»  
По команде «chmod --help» можно получить дополнительную информацию.  
[ivryabcev@fedora ~]$ chmod -x script01  
[ivryabcev@fedora ~]$ ./s
```

Рис. 3: Создание файла для второй программы, проверка содержимого домашнего каталога

- Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов. (4, 5)

```
script02 [-M--] 1 L: [ 1+ 3 4/ 6] *(33 / 80b) 0101 0x065 [*][X]
count=1
for parametr in "$@"
do
    echo "scount:sparametr"
    count=$((count+1))
done
```

Рис. 4: Текст второй программы

```
fun ivryabcev.github.io play 'Рабочий стол'
reports reports Шаблоны
[ivryabcev@fedora ~]$ cd backup
[ivryabcev@fedora backup]$ ls
script01.tar
[ivryabcev@fedora backup]$ tar -xf script01.tar
[ivryabcev@fedora backup]$ ls
script01 script01.tar
[ivryabcev@fedora backup]$ cd
[ivryabcev@fedora ~]$ mcedit script02

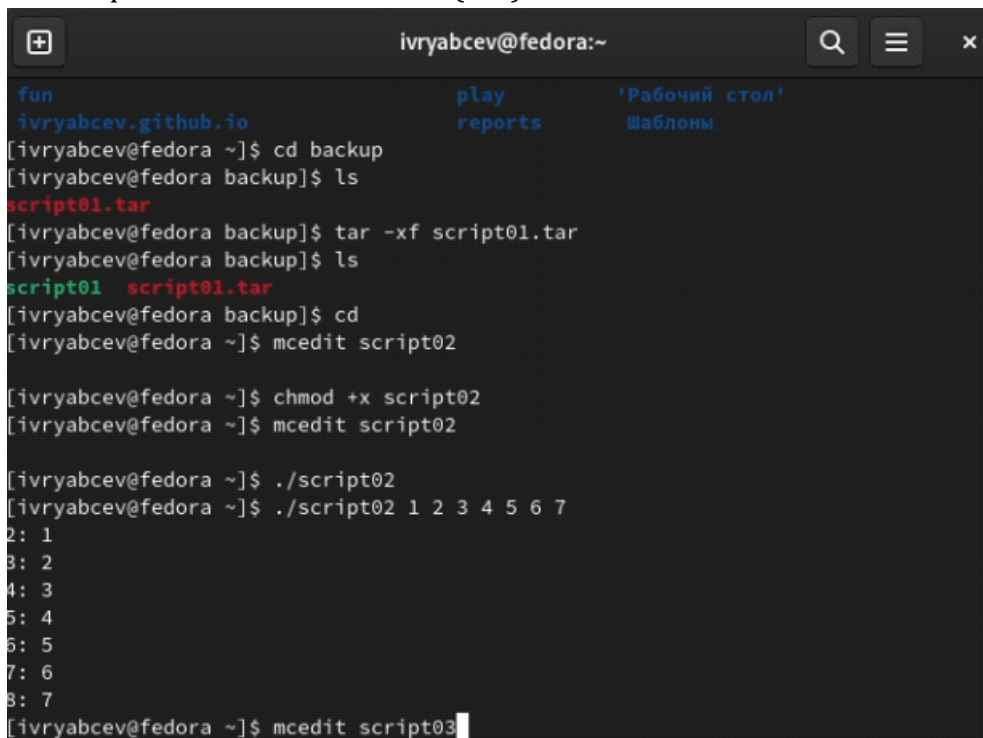
[ivryabcev@fedora ~]$ chmod +x script02
[ivryabcev@fedora ~]$ mcedit script02

[ivryabcev@fedora ~]$ ./script02
[ivryabcev@fedora ~]$ ./script02 1 2 3 4 5 6 7
2: 1
3: 2
4: 3
5: 4
6: 5
7: 6
8: 7
[ivryabcev@fedora ~]$
```

Рис. 5: Проверка работы второй программы

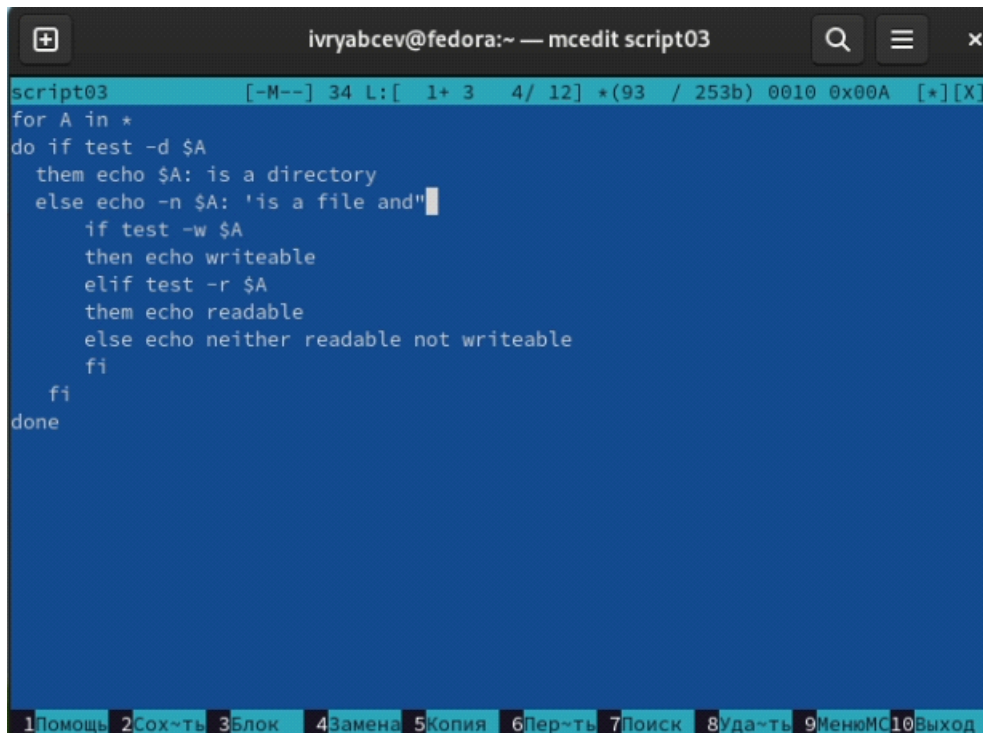
- Написать командный файл — аналог команды `ls` (без использования самой этой команды и команды `dir`). Требуется, чтобы он выдавал информацию о

нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога. (6-8)



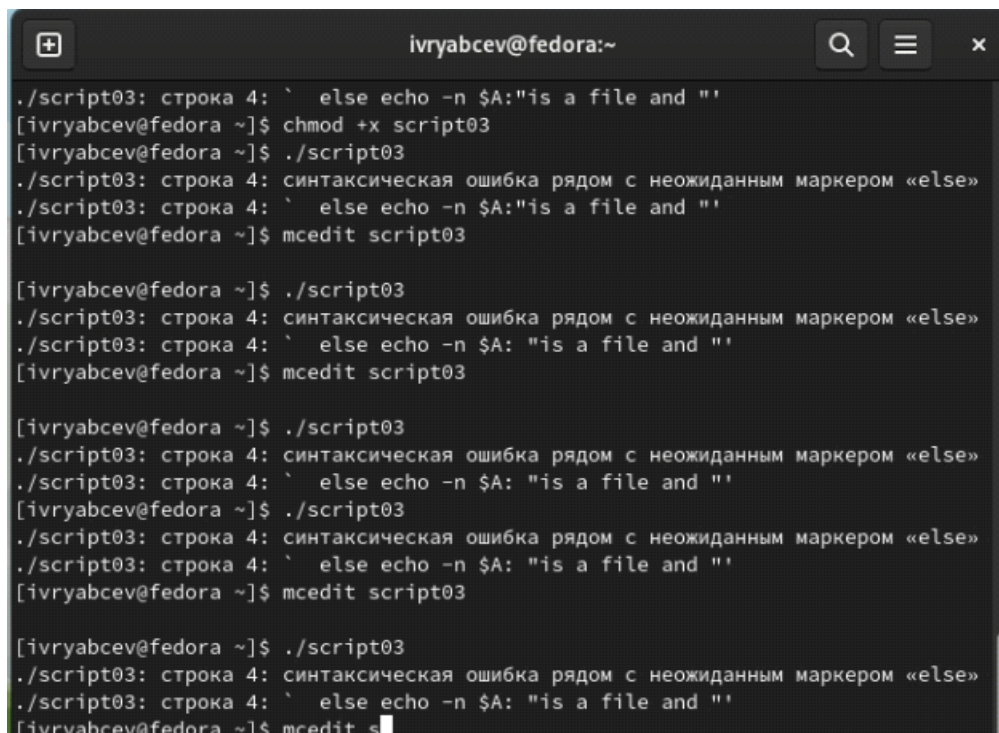
```
ivryabcev@fedora:~  
fun play 'Рабочий стол'  
ivryabcev.github.io reports Шаблоны  
[ivryabcev@fedora ~]$ cd backup  
[ivryabcev@fedora backup]$ ls  
script01.tar  
[ivryabcev@fedora backup]$ tar -xf script01.tar  
[ivryabcev@fedora backup]$ ls  
script01 script01.tar  
[ivryabcev@fedora backup]$ cd  
[ivryabcev@fedora ~]$ mcedit script02  
  
[ivryabcev@fedora ~]$ chmod +x script02  
[ivryabcev@fedora ~]$ mcedit script02  
  
[ivryabcev@fedora ~]$ ./script02  
[ivryabcev@fedora ~]$ ./script02 1 2 3 4 5 6 7  
2: 1  
3: 2  
4: 3  
5: 4  
6: 5  
7: 6  
8: 7  
[ivryabcev@fedora ~]$ mcedit script03
```

Рис. 6: Создание файла для третьей программы



```
ivryabcev@fedora:~ — mcedit script03  
script03 [-M--] 34 L: [ 1+ 3 4/ 12] *(93 / 253b) 0010 0x00A [*][X]  
for A in *  
do if test -d $A  
then echo $A: is a directory  
else echo -n $A: 'is a file and"  
if test -w $A  
then echo writeable  
elif test -r $A  
then echo readable  
else echo neither readable not writeable  
fi  
fi  
done  
1Помощь 2Сох~ть 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюМС10Выход
```

Рис. 7: Текст третьей программы

A terminal window titled 'ivryabcev@fedora:~' with search, menu, and close icons. It shows a series of commands and their outputs. The user runs './script03', which outputs an error: './script03: строка 4: ` else echo -n \$A:"is a file and "''. Then they run 'chmod +x script03' and './script03' again, getting the same error. They then use 'mcedit script03' to edit the file. After editing, they run './script03' again, but the error persists. The error message is repeated multiple times as the user continues to run the script and edit it. The script content shown in the error messages is: './script03: строка 4: ` else echo -n \$A: "is a file and "''.

```
./script03: строка 4: ` else echo -n $A:"is a file and "'
[ivryabcev@fedora ~]$ chmod +x script03
[ivryabcev@fedora ~]$ ./script03
./script03: строка 4: синтаксическая ошибка рядом с неожиданным маркером «else»
./script03: строка 4: ` else echo -n $A:"is a file and "'
[ivryabcev@fedora ~]$ mcedit script03

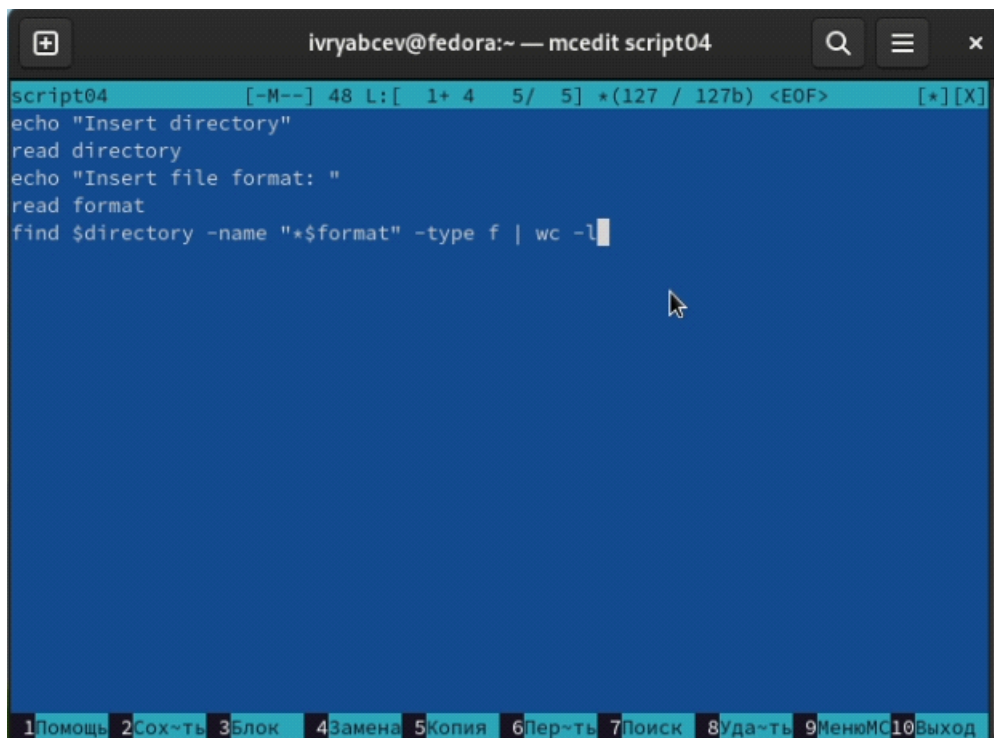
[ivryabcev@fedora ~]$ ./script03
./script03: строка 4: синтаксическая ошибка рядом с неожиданным маркером «else»
./script03: строка 4: ` else echo -n $A: "is a file and "'
[ivryabcev@fedora ~]$ mcedit script03

[ivryabcev@fedora ~]$ ./script03
./script03: строка 4: синтаксическая ошибка рядом с неожиданным маркером «else»
./script03: строка 4: ` else echo -n $A: "is a file and "'
[ivryabcev@fedora ~]$ mcedit script03

[ivryabcev@fedora ~]$ ./script03
./script03: строка 4: синтаксическая ошибка рядом с неожиданным маркером «else»
./script03: строка 4: ` else echo -n $A: "is a file and "'
[ivryabcev@fedora ~]$ mcedit s
```

Рис. 8: Проверка работы третьей программы

- Написать командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки. (9, 10)

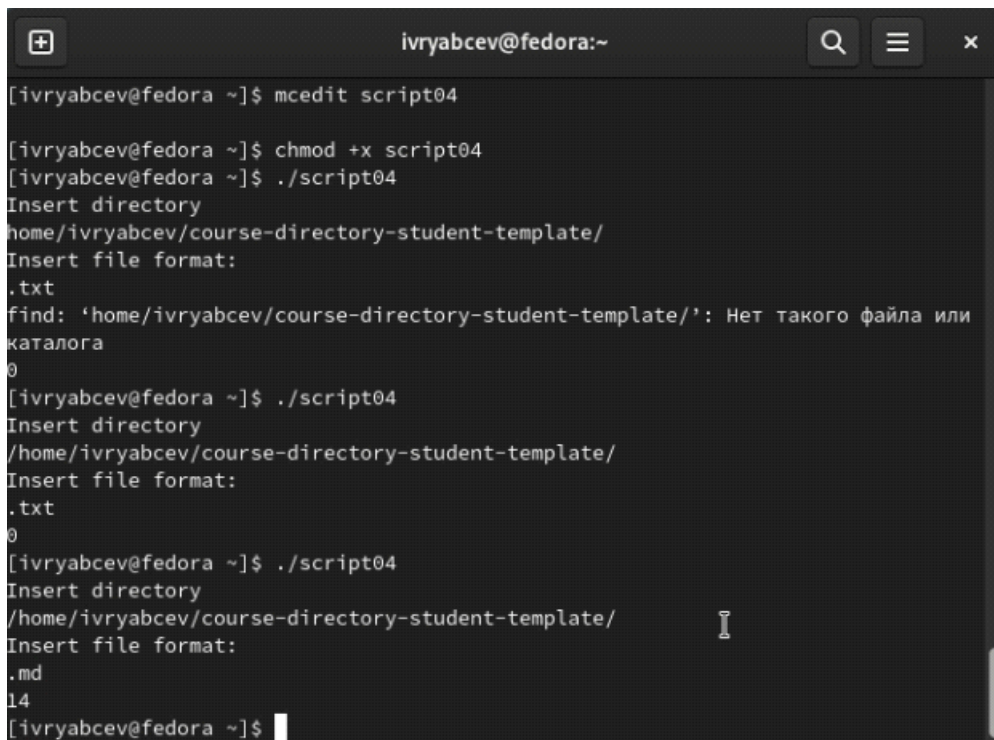


The screenshot shows a terminal window titled "ivryabcev@fedora:~ — mcedit script04". The editor displays the content of a file named "script04". The code is as follows:

```
script04      [-M--] 48 L:[ 1+ 4  5/  5] *(127 / 127b) <EOF>      [*][X]  
echo "Insert directory"  
read directory  
echo "Insert file format: "  
read format  
find $directory -name "$format" -type f | wc -l
```

At the bottom of the window, there is a menu bar with the following items: 1Помощь, 2Сох~ть, 3Блок, 4Замена, 5Копия, 6Пер~ть, 7Поиск, 8Уда~ть, 9МенюМС, 10Выход.

Рис. 9: Текст четвертой программы



The screenshot shows a terminal window titled "ivryabcev@fedora:~". The user has executed the following commands:

```
[ivryabcev@fedora ~]$ mcedit script04  
[ivryabcev@fedora ~]$ chmod +x script04  
[ivryabcev@fedora ~]$ ./script04
```

The output of the first execution is:

```
Insert directory  
home/ivryabcev/course-directory-student-template/  
Insert file format:  
.txt  
find: 'home/ivryabcev/course-directory-student-template/': Нет такого файла или каталога  
0
```

The user then runs the script again:

```
[ivryabcev@fedora ~]$ ./script04
```

The output is:

```
Insert directory  
/home/ivryabcev/course-directory-student-template/  
Insert file format:  
.txt  
0
```

On the third run:

```
[ivryabcev@fedora ~]$ ./script04
```

The output is:

```
Insert directory  
/home/ivryabcev/course-directory-student-template/  
Insert file format:  
.md  
14
```

The terminal prompt is now [ivryabcev@fedora ~]\$.

Рис. 10: Проверка работы четвертой программы

5 Выводы

В процессе выполнения лабораторной работы я изучил основы программирования в оболочке ОС UNIX/Linux. Научился писать небольшие командные файлы.

Список литературы

1. Лабораторная работа № 10. Программирование в командном процессоре ОС UNIX. Командные файлы [Электронный ресурс]. URL: <https://esystem.rudn.ru/>.