

Лабораторная работа № 10. Программирование в командном процессоре ОС UNIX. Командные файлы

Chvanova A.D.

NEC-2022, 17 May, Moscow

RUDN University, Moscow, Russian Federation

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

Написать 4 небольших командных файла.

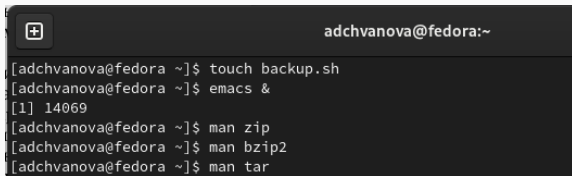
Здесь описываются теоретические аспекты, связанные с выполнением работы. Командный процессор `bash` обеспечивает возможность использования переменных типа строка символов. Имена переменных могут быть выбраны пользователем. Пользователь имеет возможность присвоить переменной значение некоторой строки символов.

Оболочка `bash` поддерживает встроенные арифметические функции. Команда `let` является показателем того, что последующие аргументы представляют собой выражение, подлежащее вычислению. Простейшее выражение — это единичный терм (`term`), обычно целочисленный.

Выполнение лабораторной работы

1. 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.

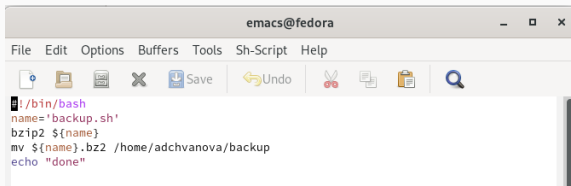
Создаем файл и изучаем архиваторы командой `man`, а также открываем `emacs` в фоновом режиме. (рис. 1)



```
adchvanova@fedora:~  
[adchvanova@fedora ~]$ touch backup.sh  
[adchvanova@fedora ~]$ emacs &  
[1] 14069  
[adchvanova@fedora ~]$ man zip  
[adchvanova@fedora ~]$ man bzip2  
[adchvanova@fedora ~]$ man tar
```

Figure 1: Создание файла и изучение справки о архиваторах, открытие `emacs` в фоновом режиме

Пишем программу.(рис. 2)

The image shows a screenshot of the Emacs text editor window. The title bar at the top reads "emacs@fedora". Below the title bar is a menu bar with the following items: "File", "Edit", "Options", "Buffers", "Tools", "Sh-Script", and "Help". Underneath the menu bar is a toolbar containing several icons: a document with a plus sign, a folder, a floppy disk, a crossed-out square, a "Save" button with a floppy disk icon, an "Undo" button with a curved arrow, a pair of scissors, a document with a plus sign, a document with a checkmark, and a magnifying glass. The main editing area contains a shell script with the following lines:

```
#!/bin/bash
name='backup.sh'
bzip2 ${name}
mv ${name}.bz2 /home/adchvanova/backup
echo "done"
```

Figure 2: Скрипт программы

Делаем файл исполняемым. (рис. 3)

```
[adchvanova@fedora ~]$ chmod +x backup.sh
```

Figure 3: Команда `chmod +x`, которая дает право на исполнение

Запускаем программу и проверяем создание архива. (рис. 4)

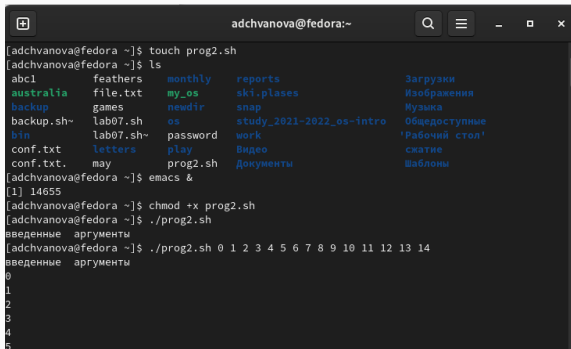
```
[adchvanova@fedora ~]$ cd backup/  
[adchvanova@fedora backup]$ ls  
backup.sh.bz2
```

Figure 4: Запуск программы и проверка создания архива.

Выполнение лабораторной работы

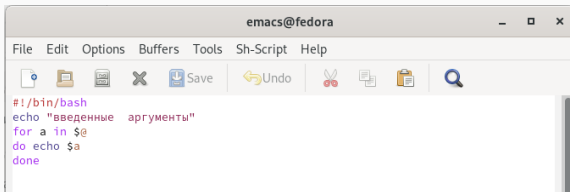
2. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов.

Создаем файл, а также открываем emacs в фоновом режиме. После написания программы делаем файл исполняемым и запускаем. (рис. 5)



```
adchvanova@fedora:~  
[adchvanova@fedora ~]$ touch prog2.sh  
[adchvanova@fedora ~]$ ls  
abcl1      feathers    monthly    reports    Заргрузки  
australia  file.txt   my_os      ski.places  Изображения  
backup     games      newdir     snap       Музыка  
backup.sh~ lab07.sh   os         study_2021-2022_os-intro  Общедоступные  
bin        lab07.sh~ password   work       'Рабочий стол'  
conf.txt   letters    play       Видео      скатие  
conf.txt   may        prog2.sh   Документы  Шаблоны  
[adchvanova@fedora ~]$ emacs &  
[1] 14655  
[adchvanova@fedora ~]$ chmod +x prog2.sh  
[adchvanova@fedora ~]$ ./prog2.sh  
введенные аргументы  
[adchvanova@fedora ~]$ ./prog2.sh 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14  
введенные аргументы  
0  
1  
2  
3  
4  
5
```


Пишем программу.(рис. 6)

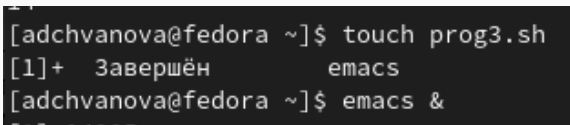
The image shows a screenshot of the Emacs text editor window. The title bar at the top reads "emacs@fedora". Below the title bar is a menu bar with the following items: "File", "Edit", "Options", "Buffers", "Tools", "Sh-Script", and "Help". Underneath the menu bar is a toolbar containing several icons: a file icon, a folder icon, a document icon, a close icon, a save icon labeled "Save", an undo icon labeled "Undo", a scissors icon, a copy icon, a paste icon, and a search icon. The main editing area contains a shell script written in a monospaced font. The script is as follows:

```
#!/bin/bash
echo "введенные  аргументы"
for a in $@
do echo $a
done
```

Figure 6: Скрипт программы

3. 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.

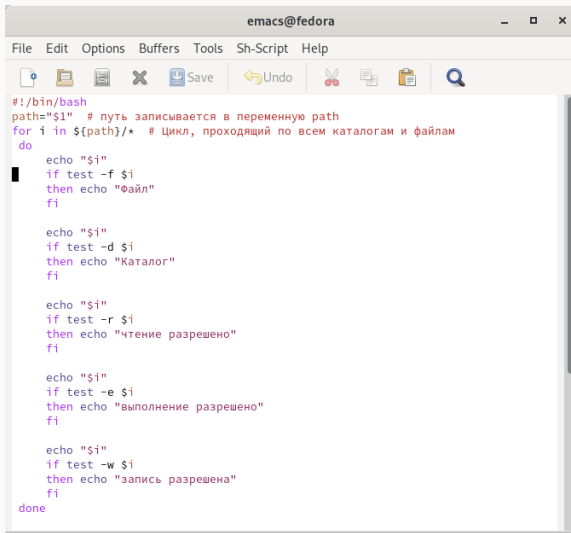
Создаем файл , а также открываем emacs в фоновом режиме. (рис. 7)



```
[adchvanova@fedora ~]$ touch prog3.sh
[1]+  Завершён          emacs
[adchvanova@fedora ~]$ emacs &
```

Figure 7: Создание файла, открытие emacs в фоновом режиме

Пишем программу.(рис. 8)

The image shows a screenshot of the Emacs text editor window titled 'emacs@fedora'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The toolbar contains icons for creating a new file, opening a file, saving, undo, redo, and search. The main text area contains a shell script written in a syntax-highlighted format. The script starts with a shebang line, followed by a comment and a variable assignment. It then enters a loop that iterates over all files and directories in the current path. For each item, it checks if it is a file, a directory, and if it is readable or writable, printing the results to the terminal.

```
#!/bin/bash
path="$1" # путь записывается в переменную path
for i in ${path}/* # Цикл, проходящий по всем каталогам и файлам
do
    echo "$i"
    if test -f $i
    then echo "Файл"
    fi

    echo "$i"
    if test -d $i
    then echo "Каталог"
    fi

    echo "$i"
    if test -r $i
    then echo "чтение разрешено"
    fi

    echo "$i"
    if test -e $i
    then echo "выполнение разрешено"
    fi

    echo "$i"
    if test -w $i
    then echo "запись разрешена"
    fi
done
```

Figure 8: Скрипт программы

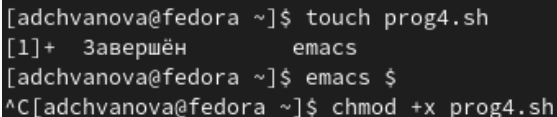
Делаем файл исполняемым. Запускаем программу (рис. 9)

```
[adchvanova@fedora ~]$ ./prog3.sh ~  
/home/adchvanova/abc1  
файл  
/home/adchvanova/abc1  
/home/adchvanova/abc1  
чтение разрешено  
/home/adchvanova/abc1  
выполнение разрешено  
/home/adchvanova/abc1  
запись разрешена  
/home/adchvanova/australia  
файл  
/home/adchvanova/australia  
/home/adchvanova/australia  
чтение разрешено  
/home/adchvanova/australia  
выполнение разрешено  
/home/adchvanova/australia  
запись разрешена  
/home/adchvanova/backup  
/home/adchvanova/backup  
каталог  
/home/adchvanova/backup  
чтение разрешено
```

Figure 9: Запуск программы

4. 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.

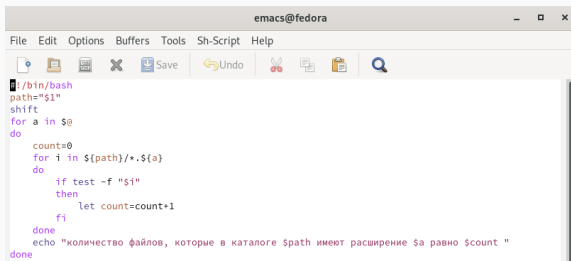
Создаем файл , а также открываем emacs в фоновом режиме. После написания программы добавляем права на исполнение (рис. 10)



```
[adchvanova@fedora ~]$ touch prog4.sh
[1]+  Завершён          emacs
[adchvanova@fedora ~]$ emacs $
^C[adchvanova@fedora ~]$ chmod +x prog4.sh
```

Figure 10: Создание файла, открытие emacs в фоновом режиме, добавление прав на исполнение

Пишем программу.(рис. 11)

The image shows a screenshot of the Emacs text editor running on a Fedora system. The window title is 'emacs@fedora'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. Below the menu bar is a toolbar with icons for file operations and editing. The main text area contains a shell script written in Bash. The script is color-coded: keywords like 'for', 'do', 'done', 'if', 'then', 'fi', 'let', and 'echo' are in purple; variables and file paths are in pink; and the rest of the code is in black. The script counts the number of files in a directory with a specific extension.

```
#!/bin/bash
path="$1"
shift
for a in $@
do
  count=0
  for i in ${path}/*.${a}
  do
    if test -f "$i"
    then
      let count=count+1
    fi
  done
  echo "количество файлов, которые в каталоге $path имеют расширение $a равно $count "
done
```

Figure 11: Скрипт программы

Запускаем программу (рис. 12)

```
[adchvanova@fedora ~]$ ./prog4.sh ~ txt pdf sh
количество файлов, которые в каталоге /home/adchvanova имеют расширение txt равно 2
количество файлов, которые в каталоге /home/adchvanova имеют расширение pdf равно 0
количество файлов, которые в каталоге /home/adchvanova имеют расширение sh равно 4
[adchvanova@fedora ~]$
```

Figure 12: Запуск программы

Мы изучили основы программирования в оболочке ОС UNIX/Linux. Научились писать небольшие командные файлы. (скрипт, который при запуске делает резервную копию самого себя; скрипт, обрабатывающий любое произвольное число аргументов командной строки; командный файл — аналог команды `ls`; 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)

Спасибо за внимание!
