

Лабораторная работа № 11. Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Chvanova A.D.

NEC-2022, 24 May, Moscow

RUDN University, Moscow, Russian Federation

Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Написать 4 программных файла

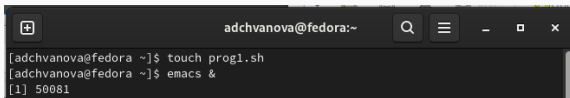
Функция `getopts` включает две специальные переменные среды — `OPTARG` и `OPTIND`. Если ожидается дополнительное значение, то `OPTARG` устанавливается в значение этого аргумента (будет равна `file_in.txt` для опции `i` и `file_out.doc` для опции `o`. `OPTIND` является числовым индексом на упомянутый аргумент. Функция `getopts` также понимает переменные типа массив, следовательно, можно использовать её в функции не только для синтаксического анализа аргументов функций, но и для анализа введённых пользователем данных.

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;
- `-ршаблон` — 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`.

Создаем файл , а также открываем emacs в фоновом режиме. (рис. 1)

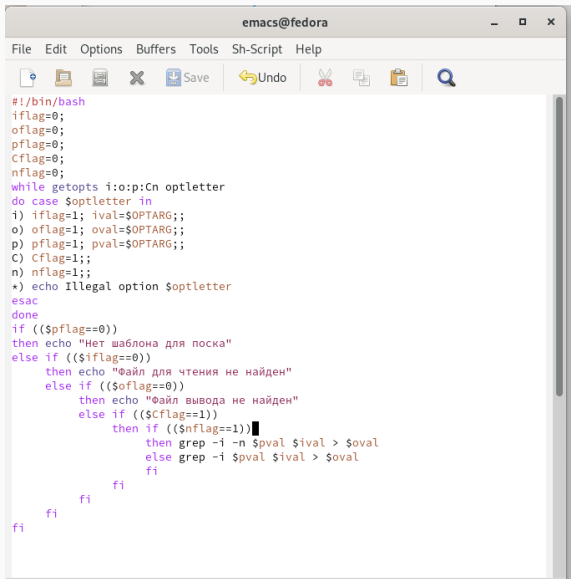
A terminal window with a dark background. The title bar shows the user 'adchvanova@fedora' and the home directory '~'. The terminal contains three lines of text: the first line shows the command 'touch prog1.sh' being executed; the second line shows the command 'emacs &' being executed; the third line shows the process ID '[1] 50081' assigned to the emacs process.

```
adchvanova@fedora:~  
[adchvanova@fedora ~]$ touch prog1.sh  
[adchvanova@fedora ~]$ emacs &  
[1] 50081
```

Figure 1: Создание файла, открытие emacs в фоновом режиме

Выполнение лабораторной работы

Пишем программу.(рис. 2)



The screenshot shows the Emacs editor window titled 'emacs@fedora'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The toolbar contains icons for opening files, saving, undo, and search. The main text area displays a shell script in Bash. The script initializes several flags (iflag, oflag, pflag, Cflag, nflag) to 0. It then enters a 'while' loop using 'getopts' to process command-line options: 'i' for iflag, 'o' for oflag, 'p' for pflag, 'C' for Cflag, and 'n' for nflag. If an illegal option is provided, it echoes an error message. After the loop, it checks if pflag is set; if not, it echoes 'Нет шаблона для поиска'. If pflag is set, it checks if iflag is set; if not, it echoes 'Файл для чтения не найден'. If iflag is set, it checks if oflag is set; if not, it echoes 'Файл вывода не найден'. If both iflag and oflag are set, it checks if Cflag is set. If Cflag is set, it checks if nflag is set. If nflag is set, it uses 'grep' to search for the pattern in the file. If nflag is not set, it uses 'grep' to search for the pattern in the file. The script ends with 'fi' statements to close the nested if-else blocks.

```
#!/bin/bash
iflag=0;
oflag=0;
pflag=0;
Cflag=0;
nflag=0;
while getopts i:o:p:Cn 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 "Нет шаблона для поиска"
else if (($iflag==0))
then echo "Файл для чтения не найден"
else if (($oflag==0))
then echo "Файл вывода не найден"
else if (($Cflag==1))
then if (($nflag==1))
then grep -i -n $pval $ival > $oval
else grep -i $pval $ival > $oval
fi
fi
fi
fi
fi
```

Делаем файл исполняемым и проверяем его работу. (рис. 3)

```
[adchvanova@fedora ~]$ touch text1.txt
[adchvanova@fedora ~]$ touch text2.txt
[adchvanova@fedora ~]$ chmod +x prog1.sh
[adchvanova@fedora ~]$ ./prog1.sh -i text1.txt -o text2.txt -p good -C
[adchvanova@fedora ~]$ cat text2.txt
good
[adchvanova@fedora ~]$ ./prog1.sh -i text1.txt -o text2.txt -p good -Cn
[adchvanova@fedora ~]$ cat text2.txt
2:good
[adchvanova@fedora ~]$
```

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


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.

Создаем файлы , а также открываем етас в фоновом режиме. Делаем их исполняемыми (рис. 4)

```
[adchvanova@fedora ~]$ touch n2.c  
[adchvanova@fedora ~]$ emacs &  
[2] 51257  
[adchvanova@fedora ~]$ touch prog2.sh  
[adchvanova@fedora ~]$ emacs &  
[3] 51509  
[adchvanova@fedora ~]$ chmod +x prog2.sh
```

Figure 4: Создание файлов , открытие emacs в фоновом режиме. Команда `chmod +x`, которая дает право на исполнение

Пишем программу на языке C.(рис. 5)

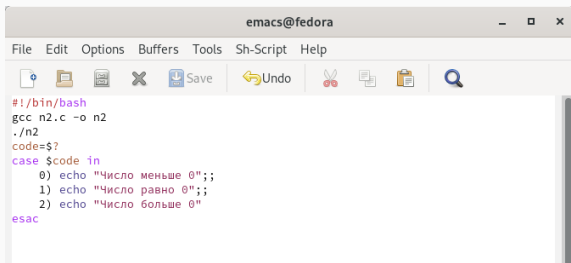
The image shows a screenshot of the Emacs text editor window. The title bar at the top reads "emacs@fedora". Below the title bar is a menu bar with "File", "Edit", "Options", "Buffers", "Tools", "C", and "Help". Under the menu bar is a toolbar with icons for opening a file, saving, undo, redo, and search. The main editing area contains the following C code:

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    printf("Введите число\n");
    int n;
    scanf("%d", &n);
    if(n<0) exit(0);
    if(n==0) exit(1);
    if(n>0) exit(2);
    return 0;
}
```

A cursor is visible at the end of the closing brace of the main function.

Figure 5: Скрипт программы на C

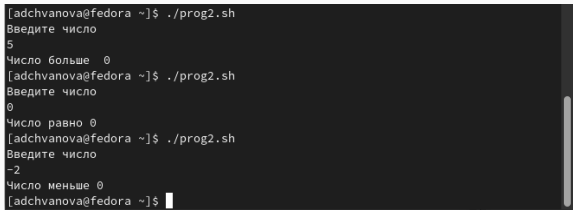
Пишем командный файл .(рис. 6)

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 icons for opening a file, saving a file, undoing an action, redoing an action, and searching. The main editing area contains a shell script with the following content:

```
#!/bin/bash
gcc n2.c -o n2
./n2
code=$?
case $code in
  0) echo "Число меньше 0";;
  1) echo "Число равно 0";;
  2) echo "Число больше 0"
esac
```

Figure 6: Скрипт программы

Работа программы. (рис. 7)



```
[adchvanova@fedora ~]$ ./prog2.sh
Введите число
5
Число больше 0
[adchvanova@fedora ~]$ ./prog2.sh
Введите число
0
Число равно 0
[adchvanova@fedora ~]$ ./prog2.sh
Введите число
-2
Число меньше 0
[adchvanova@fedora ~]$
```

Figure 7: Работа программы

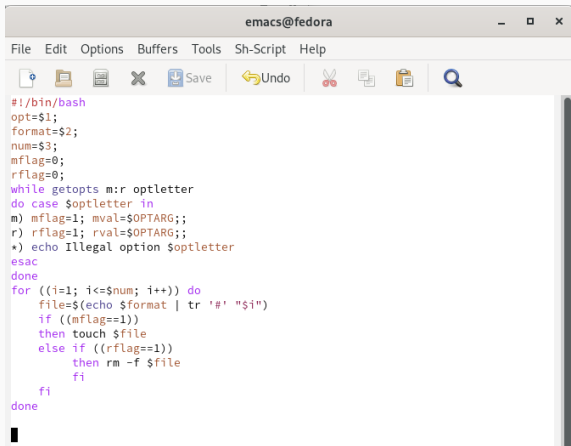
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до [?] (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).

Создаем файл , а также открываем етас в фоновом режиме. Делаем их исполняемыми (рис. 8)


```
[adchvanova@fedora ~]$ touch prog3.sh
[2]-  Завершён      emacs
[3]+  Завершён      emacs
[adchvanova@fedora ~]$ emacs &
[2] 52545
[adchvanova@fedora ~]$ chmod +x prog3.sh
```

Figure 8: Создание файла , открытие emacs в фоновом режиме. Команда chmod +x, которая дает право на исполнение

Пишем программу.(рис. 9)



```
#!/bin/bash
opt=$1;
format=$2;
num=$3;
mflag=0;
rflag=0;
while getopts m:r optletter
do case $optletter in
m) mflag=1; mval=$OPTARG;;
r) rflag=1; rval=$OPTARG;;
*) echo Illegal option $optletter
esac
done
for ((i=1; i<=$num; i++)) do
    file=$(echo $format | tr '#' "$i")
    if ((mflag==1))
    then touch $file
    else if ((rflag==1))
    then rm -f $file
    fi
fi
done
```

Figure 9: Скрипт программы на C

Работа программы (рис. 10 - 11)

```
[adchvanova@fedora ~]$ ./prog3.sh -m hh#.txt 3
[adchvanova@fedora ~]$ ls
010-lab_shell_prog_1.pdf  my_os      pr.sh~
abc1                      n2         reports
australia                n2.c      ski.places
backup                   n2.c~     snap
backup.sh~              newdir     study_2021-2022_os-intro
bin                      os         text1.txt
conf.txt                password   text2.txt
conf.txt.              play       work
feathers                pr10.3.sh Видео
file.txt               pr10.3.sh~ Документы
games                  prog1.sh  Загрузки
hh1.txt                prog1.sh~ Изображения
hh2.txt                prog2.sh  Музыка
hh3.txt                prog2.sh~ Общедоступные
lab07.sh~              prog3.sh  'Рабочий стол'
letters                prog3.sh~ сжатие
may                    prog4.sh~ Шаблоны
monthly                pr.sh
```

Figure 10: Работа программы

Выполнение лабораторной работы

```
[adchvanova@fedora ~]$ ./prog3.sh -m hh#.txt 3
[adchvanova@fedora ~]$ ls
010-lab_shell_prog_1.pdf  my_os      pr.sh-
abc1                      n2         reports
australia                n2.c      ski.places
backup                   n2.c~     snap
backup.sh~              newdir     study_2021-2022_os-intro
bin                      os         text1.txt
conf.txt                password   text2.txt
conf.txt.              play       work
feathers                pr10.3.sh  Видео
file.txt               pr10.3.sh~ Документы
games                  prog1.sh   Загрузки
hh1.txt                prog1.sh~  Изображения
hh2.txt                prog2.sh   Музыка
hh3.txt                prog2.sh~  Общедоступные
lab07.sh~              prog3.sh   'Рабочий стол'
letters                prog3.sh~  сжатие
may                    prog4.sh~  Шаблоны
monthly                pr.sh
```

Figure 11: Работа программы

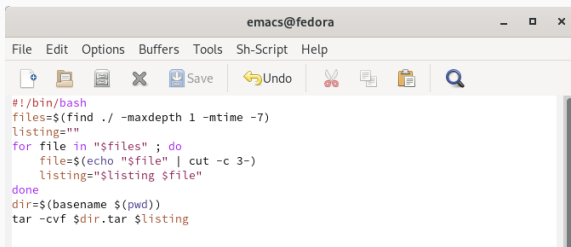
4. Написать командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду `find`). Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию

Создаем файл , а также открываем emacs в фоновом режиме. Делаем их исполняемыми (рис. 12)

```
[adchvanova@fedora ~]$ touch prog4.sh
[2]+  Завершён      emacs
[adchvanova@fedora ~]$ emacs &
[2] 53628
[adchvanova@fedora ~]$ chmod +x prog4.sh
```

Figure 12: Создание файла , открытие emacs в фоновом режиме. Команда `chmod +x`, которая дает право на исполнение

Пишем программу.(рис. 13)

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 "File", "Edit", "Options", "Buffers", "Tools", "Sh-Script", and "Help". Underneath the menu bar is a toolbar with icons for creating a new file, opening a file, saving a file, closing a file, saving the current file, undoing the last action, cutting, copying, pasting, and searching. The main editing area contains a shell script written in bash. The script uses the 'find' command to locate files, loops through them to build a list, and then uses 'tar' to create an archive of the files.

```
#!/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
```

Figure 13: Скрипт программы на С

Работа программы (рис. 14)

```
[adchvanova@fedora ~]$ ./prog4.sh
Загрузки/
Загрузки/Отчет Лабораторная работа 2.docx
Загрузки/study_2021-2022_os-intro-master.zip
Загрузки/presentation.md
Загрузки/report_lab4.md
Загрузки/28-04-2022_13-26-12.zip
Загрузки/lab3report (3).md
Загрузки/28-04-2022_13-26-12/
```

Figure 14: Работа программы

Мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Спасибо за внимание!
