

Лабораторная работа №13

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Жибицкая Евгения Дмитриевна

Российский университет дружбы народов, Москва, Россия

Цель

Изучение основ программирования в оболочке ОС UNIX. Приобретение навыков в написании более сложных командных файлов с использованием логических управляющих конструкций и циклов.

Ход работы

Пункт 2. Задание

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.

```
foot
[edzhbitskaya@edzhbitskaya ~]$ touch 13-2.c
[edzhbitskaya@edzhbitskaya ~]$ touch 13-2.sh
[edzhbitskaya@edzhbitskaya ~]$ emacs
(emacs:4413): Gtk-CRITICAL **: 12:52:36.162: gtk_distribute_natural_allocation: assertion 'extra_space >= 0' failed
[edzhbitskaya@edzhbitskaya ~]$ emacs
```

Рис. 1: Запуск редактора

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

int main(){
    printf("Введите число");
    int x;
    scanf("%d", &x);
    if (x<0) exit (2);
    if (x>0) exit (1);
    if (x==0) exit (0);
    return 0;
}
```

Рис. 2: Код на с#

Пункт 2. Результат

```
#!/bin/bash
gcc 13-2.c -o 13-2
./13-2
case $? in
  0) echo "Число равно нулю";;
  1) echo "Число больше нуля";;
  2) echo "Число меньше нуля";;
esac
```

Рис. 3: Программа 2

```
./13-2.sh: строка 3: ./13-2: нет такого файла или каталога
[edzhibitskaya@edzhibitskaya ~]$ ./13-2.sh
Введите число 4
Число больше нуля
[edzhibitskaya@edzhibitskaya ~]$ ./13-2.sh
Введите число -9
Число меньше нуля
[edzhibitskaya@edzhibitskaya ~]$
```

Рис. 4: Вывод

Пункт 1. Задание

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`.

Пункт 1. Выполнение

```
[edzhibitskaya@edzhibitskaya ~]$ touch f1.txt f2.txt
[edzhibitskaya@edzhibitskaya ~]$ gedit f1.txt
[edzhibitskaya@edzhibitskaya ~]$ chmod +x 13-1.sh
[edzhibitskaya@edzhibitskaya ~]$ ./13-1.sh -i f1.txt -o f2.txt -p корабль -C -n
[edzhibitskaya@edzhibitskaya ~]$ cat f2.txt
2:Вез корабль карамель, наскочил корабль на мель,
матросы две недели карамель на мели ели.
[edzhibitskaya@edzhibitskaya ~]$ gedit f1.txt
[edzhibitskaya@edzhibitskaya ~]$ ./13-1.sh -i f1.txt -o f2.txt -p мама -C -n
[edzhibitskaya@edzhibitskaya ~]$ cat f2.txt
1:мама мыла раму
[edzhibitskaya@edzhibitskaya ~]$
```

Рис. 5: Работа с файлом + вывод

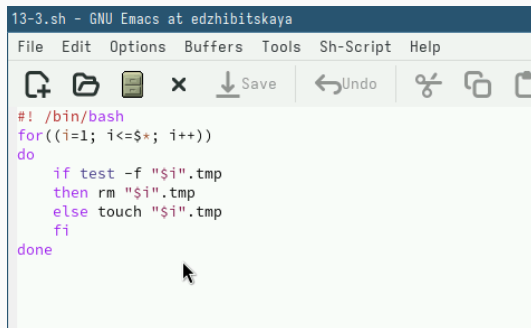
```
#!/bin/bash
iflag=0; oflag=0; pflag=0; cflag=0; nflag=0;
while getopts i:o:p:Cn optletter
do case $optletter in
    i) iflag=1; ival=$OPTARG;;
    o) oflag=1; oval=$OPTARG;;
    p) pflag=1; pval=$OPTARG;;
    C) cflag=1;;
    n) nflag=1;;
    *) echo illegal option $optletter
       esac
done
if (($pflag==0))
then echo "Not found"
else
    if (($iflag==0))
    then echo "File not found"
    else
        if ((oflag==0))
        then if (($cflag==0))
            then if (($nflag==0))
                then grep $pval $ival
            else
                grep -n $pval $ival
            fi
        else if (($nflag==0))
            then grep -l $pval $ival
        else grep -i -n $pval $ival
        fi
    fi
    else if (($cflag==0))
    then if (($nflag==0))
```

Рис. 6: Программа 1

Пункт 3. Задание

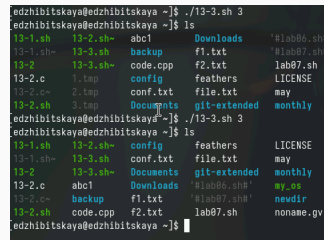
Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до ☐ (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если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

Пункт 3. Реализация



```
13-3.sh - GNU Emacs at edzhibitskaya
File Edit Options Buffers Tools Sh-Script Help
[Icons] Save Undo [Icons]
#!/bin/bash
for((i=1; i<=3; i++))
do
    if test -f "$i".tmp
    then rm "$i".tmp
    else touch "$i".tmp
    fi
done
```

Рис. 7: Программа 3



```
edzhibitskaya@edzhibitskaya ~]$ ./13-3.sh 3
edzhibitskaya@edzhibitskaya ~]$ ls
13-1.sh 13-2.sh~ abc1 Downloads '#lab06.sh#
13-1.sh~ 13-3.sh backup f1.txt '#lab07.sh#
13-2 13-3.sh~ code.cpp f2.txt lab07.sh
13-2.c 1.tmp config feathers LICENSE
13-2.c~ 2.tmp conf.txt file.txt may
13-2.sh 3.tmp Documents git-extended monthly
edzhibitskaya@edzhibitskaya ~]$ ./13-3.sh 3
edzhibitskaya@edzhibitskaya ~]$ ls
13-1.sh 13-2.sh~ config feathers LICENSE
13-1.sh~ 13-3.sh conf.txt file.txt may
13-2 13-3.sh~ Documents git-extended monthly
13-2.c abc1 Downloads '#lab06.sh# my_os
13-2.c~ backup f1.txt '#lab07.sh# newdir
13-2.sh code.cpp f2.txt lab07.sh noname.gv
edzhibitskaya@edzhibitskaya ~]$
```

Рис. 8: Выполнение

Пункт 4. Задание

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`).

Пункт 4. Код

```
#!/bin/bash
find $* -mtime -7 -type f > FILES.txt
tar -cf archive.tar -T FILES.txt
```

Рис. 9: Исполняемый файл

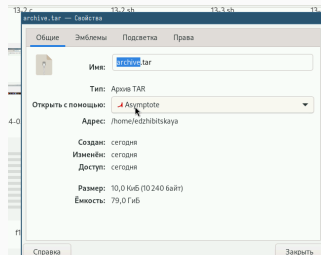


Рис. 10: Создание архива

Вывод

В ходе работы мы изучили основные программирования в оболочке UNIX, написали несколько более сложных командных файлов.