

Структура научной презентации

Простейший шаблон

Голощапов Я.В.

22 апреля 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Голощапов Ярослав Вячеславович
- Студент 1-го курса
- Российский университет дружбы народов
- <https://github.com/yvgoloschapov>

Цель работы

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

Задание

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 `-р`.

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.

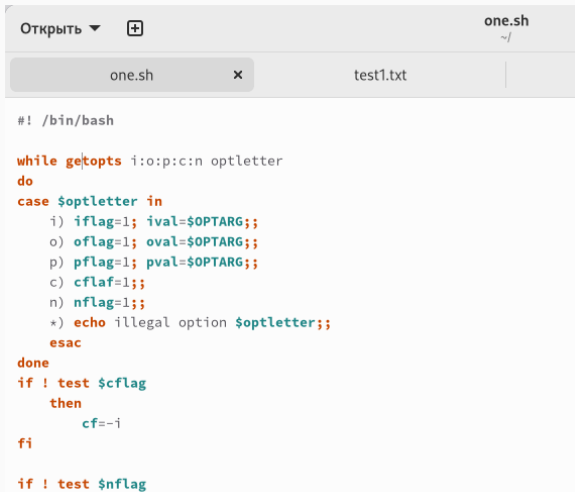
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до `N` (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы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

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`).

Выполнение лабораторной работы

Выполнение лабораторной работы

Я создал файл one.sh для первой программы, а также текстовые файлы, откуда будет браться информация. Написал код программы.



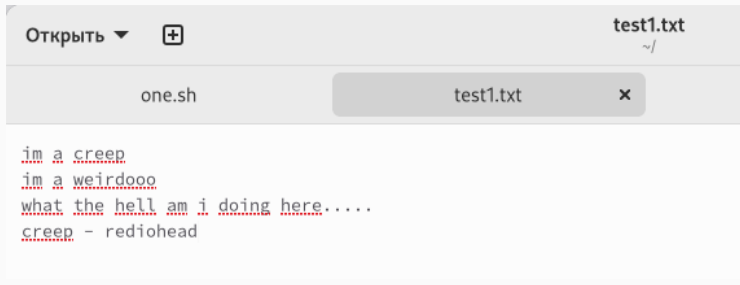
```
#!/bin/bash

while getopts i:op: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 ! test $cflag
then
cf=-i
fi

if ! test $nflag
```

Выполнение лабораторной работы

Запустил программу и проверил, работает ли она . Программа работает.



```
im a creep
im a weirdooo
what the hell am i doing here....
creep - rediohead
```

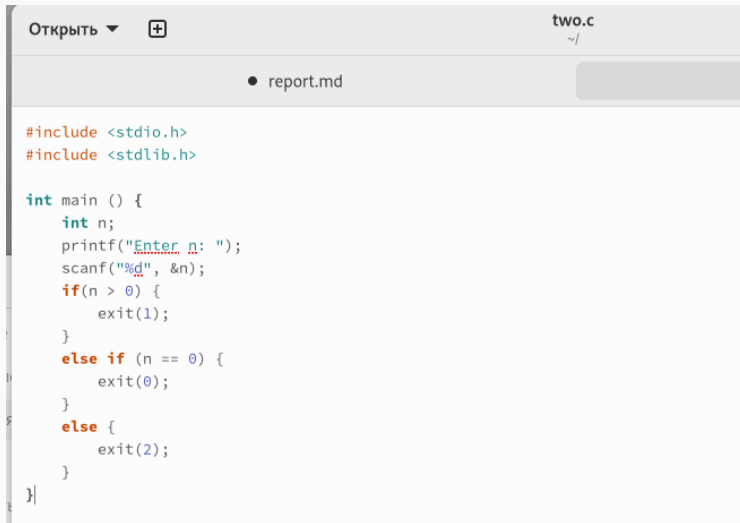
Figure 2: Исходный текстовый файл



```
im a creep
```

Выполнение лабораторной работы

Создал файлы для второй программы: two.sh и two.c и заполнил код.

A screenshot of a code editor window. The title bar at the top shows 'Открыть' (Open) with a dropdown arrow and a plus icon, followed by the filename 'two.c' and a tilde symbol '~/'. Below the title bar is a tab bar with a single tab labeled 'report.md'. The main area of the editor contains C code. The code includes two header files: <stdio.h> and <stdlib.h>. The main function is defined as 'int main () {' and contains three conditional branches. The first branch checks if 'n > 0' and calls 'exit(1)'. The second branch checks if 'n == 0' and calls 'exit(0)'. The third branch is an 'else' case that calls 'exit(2)'. The code is color-coded: keywords are blue, comments are green, and string literals are red. The code ends with a closing brace '}' for the main function.

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

int main () {
    int n;
    printf("Enter n: ");
    scanf("%d", &n);
    if(n > 0) {
        exit(1);
    }
    else if (n == 0) {
        exit(0);
    }
    else {
        exit(2);
    }
}
```

Проверил работу командного файла.

```
two.sh: строка 4: ./cprog: Нет такого файла или каталога
[yvgolothapov@fedora ~]$ bash two.sh
Enter n: 10
N > 0
[yvgolothapov@fedora ~]$ bash two.sh
Enter n: 0
N = 0
[yvgolothapov@fedora ~]$ bash two.sh
Enter n: -5
N < 0
[yvgolothapov@fedora ~]$
```

Figure 5: Проверка работы

Создал третий файл three.sh и написал код.



```
#!/bin/bash
for ((i=1; i<=$*; i++))
do
if test -f "$i".tmp
then rm "$i".tmp
else touch "$i.tmp"
fi
done
```

Figure 6: Создание третьего файла

Запустил файл. Программа работает.

```
[yvgolothapov@fedora ~]$ bash three.sh 2  
[yvgolothapov@fedora ~]$ bash three.sh 2  
[yvgolothapov@fedora ~]$
```

Figure 7: Запуск программы

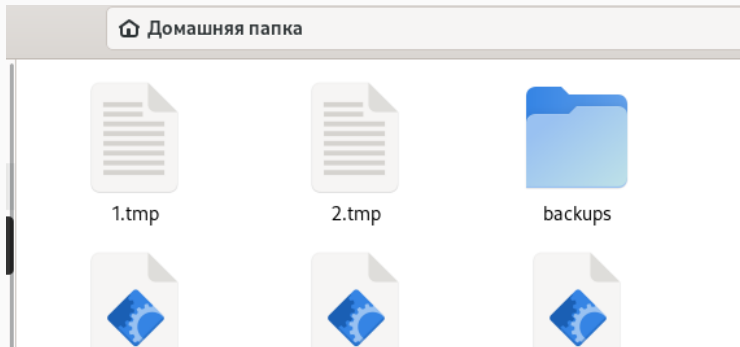


Figure 8: Созданные файлы

Создал файл four.sh.



```
#!/bin/bash

find $* -mtime -7 -mtime +0 -type f > FILES.txt
tar -cf archive.tar -T FILES.txt
```

Figure 9: Файл 4

Программа работает.

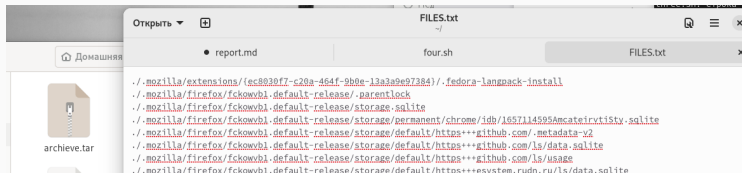


Figure 10: Результат работы программы

Выводы

Я изучил основы программирования в оболочке ОС UNIX, а также научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

...