Лабораторная работа №11

Программирование в командном процессоре ОС UNIX. Ветвления и циклы.

Латыпова Диана. НФИбд-02-21

Содержание

# 1 Цель работы

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

# 2 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;
* -pшаблон — 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.

1. 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.
2. 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).
3. 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).

# 3 Выполнение лабораторной работы

Для начале я перешла в папку lab11:

**cd ucheba/lab11**

1. Запустила редактор emacs 11\_1. Здесь написала скрипт 1 задания (рис. 1)

**Скрипт 1 задания:**

#!/bin/bash

while getopts “i:o:p:cn” opt

do

case $opt in  
 i)inputfile="$OPTARG";;  
 o)outputfile="$OPTARG";;  
 p)sample="$OPTARG";;  
 c)reg="";;  
 n)line="";;  
  
esac

done

grep -n “$sample" "$inputfile” > “$outputfile”

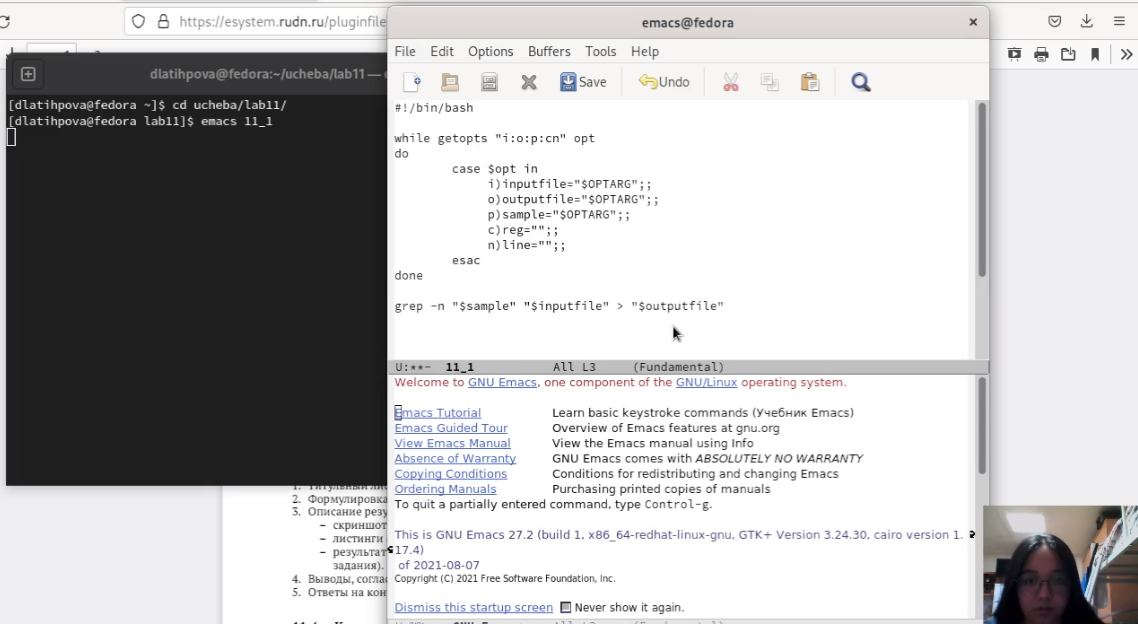


Рис. 1: Скрипт 1 задания

Сначала предоставила права на выполнение файла 11\_1. И запустила файл(рис. 2):

1 **chmod +x 11\_1**

2 **./11\_1 -i conf.txt -o result.txt -p n etconf -c -n**

3 **ls**

4 **cat result.txt**

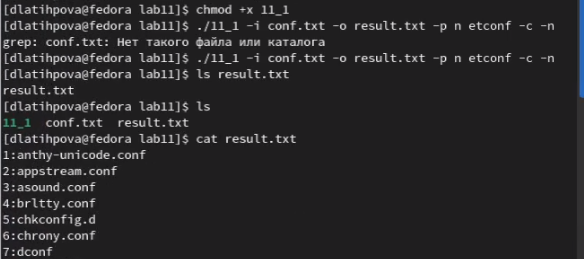


Рис. 2: Результат 1 скрипта

Все корректно работает.

1. Запустила редактор emacs 11\_2. Здесь написала скрипт 2 задания (рис. 3)

**Скрипт 2 задания:**

#!/bin/bash

RES=result

SRC=main.cpp

if [ “$SRC" -nt "$RES” ]

then

echo "Creating $RES ..."  
  
g++ -o $RES $SRC

fi

./$RES $1

ec=$?

if [ “$ec” == “1” ]

then

echo "input > 0"

fi

if [ “$ec” == “2” ]

then

echo "input = 0"

fi

if [ “$ec” == “3” ]

then

echo "input < 0"

fi

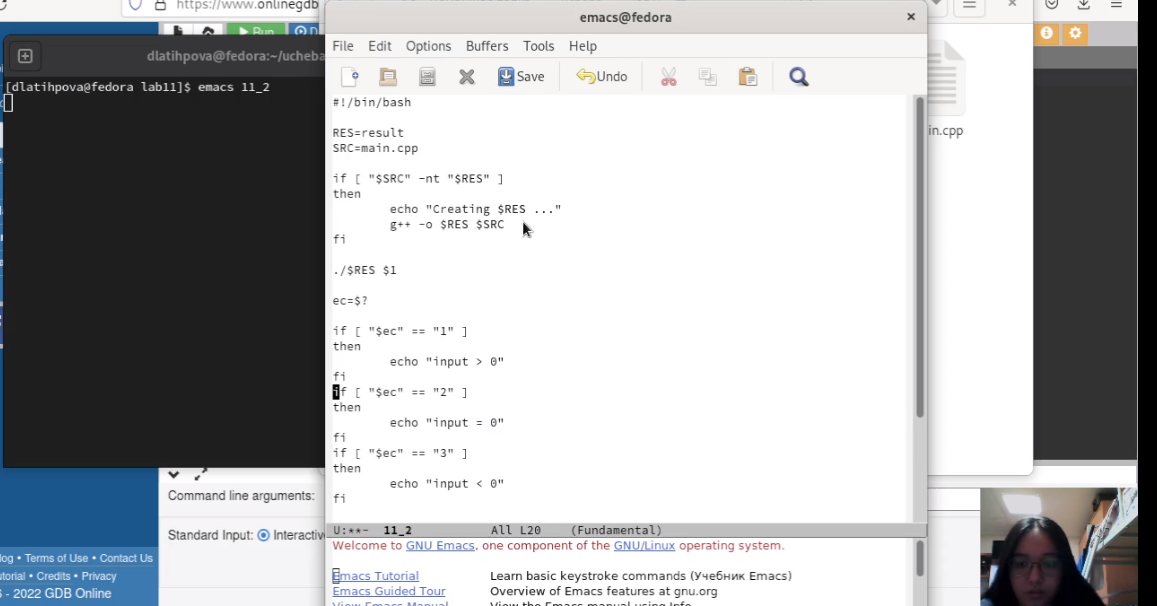


Рис. 3: Скрипт 2 задания

Код С++(рис. 4):

#include

using namespace std;

int main(int argc,char \*argc[])

{ if (atoi(argv[1]>0) exit(1);

else if (atoi(argv[1]==0) exit(2);  
  
else exit(3);  
  
return 0;

}

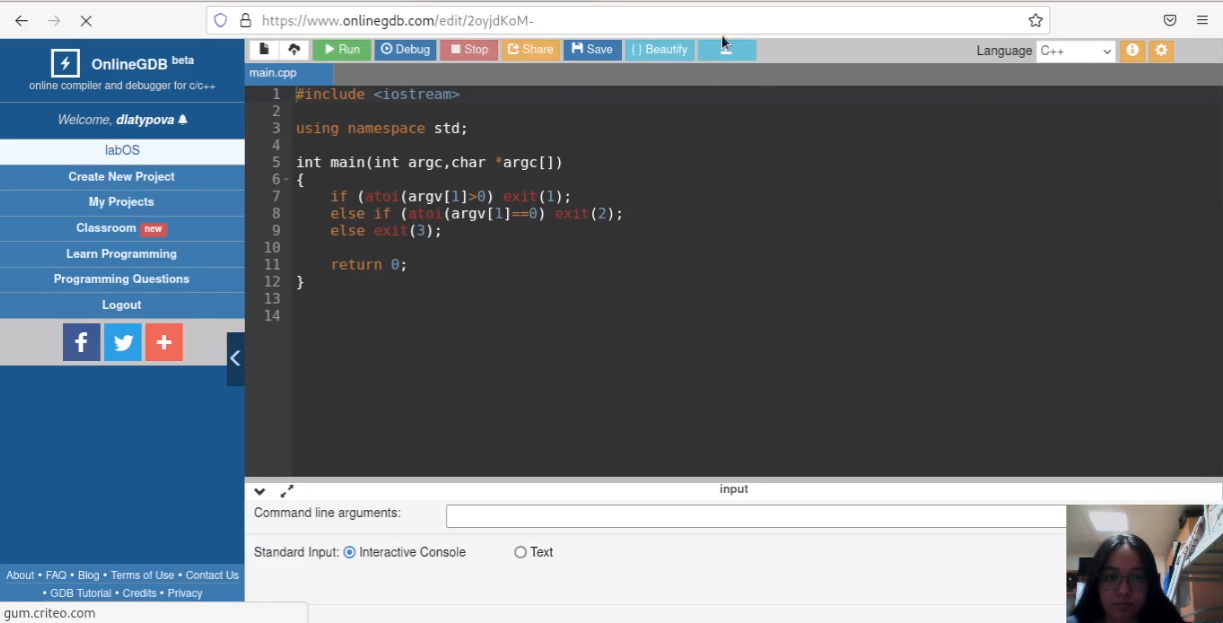


Рис. 4: Код на С++

Снова сначала предоставила права на выполнение файла 11\_2. И запустила файл(рис. 5):

1 **chmod +x 11\_2**

2 **./11\_2 1**

3 **./11\_2 0**

4 **./11\_2 -2**

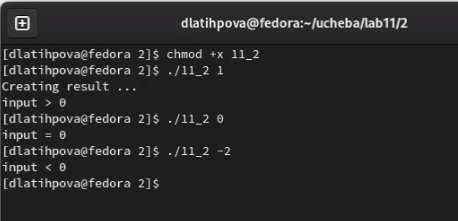


Рис. 5: Результат 2 скрипта

1. Приступила к написанию 3 скрипта(рис. 6):

**emacs 11\_3**

**Скрипт 3 задания:**

#!/bin/bash

while getopts c:r opt

do

case $opt in  
 c)n="$OPTARG"; for i in $(seq 1 $n); do touch "$i.tmp"; done;;  
 r)for i in $(find -name "\*.tmp"); do rm $i; done;;  
  
esac

done

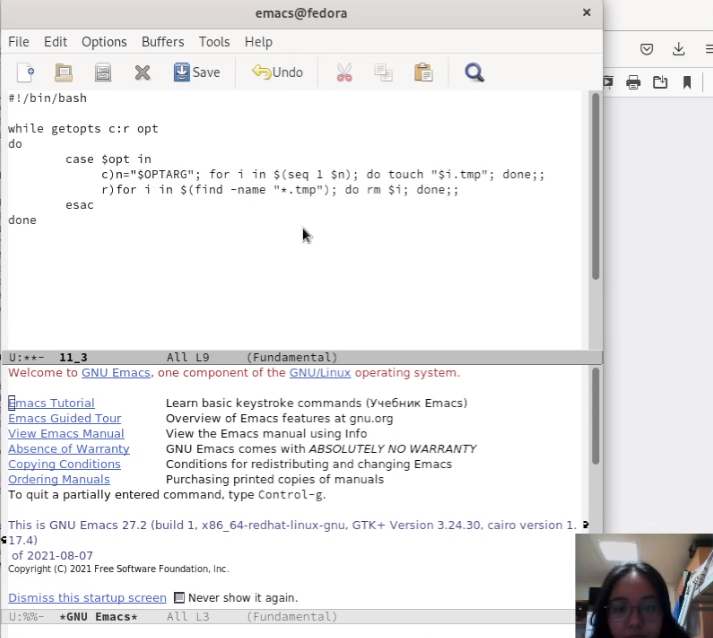


Рис. 6: Скрипт 3 задания

Снова предоставила права на выполнение файла 11\_3. И запустила файл(рис. 7):

1 **chmod +x 11\_3**

2 **./11\_3 -c 5**

3 **ls**

4 **./11\_3 -r**

3 **ls**

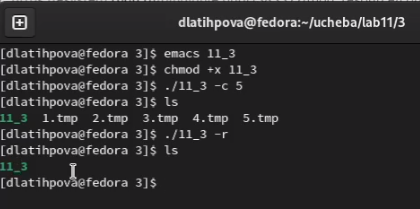


Рис. 7: Результат 3 скрипта

Скрипт корректно работает.

1. Запустила редактор emacs 11\_4. Здесь написала скрипт 4 задания (рис. 8)

**Скрипт 4 задания:**

#!/bin/bash

while getopts :p: opt

do

case $opt in  
 p)dir="$OPTARG";;  
esac

done

find $dir -mtime -7 -mtime +0 -type f > archiv.txt

tar -cf result.tar -T archiv.txt

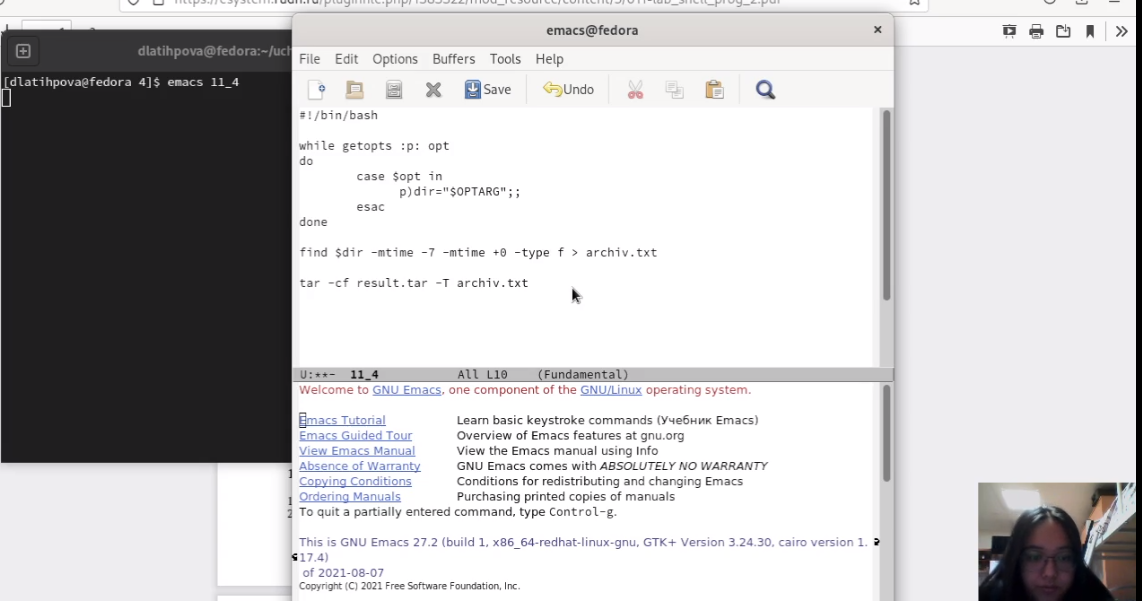


Рис. 8: Скрипт 4 задания

Снова предоставила права на выполнение файла 11\_3. И запустила файл(рис. 9)(рис. 10):

1 **chmod +x 11\_4**

2 **./11\_4 -p /home/dlatihpova/**

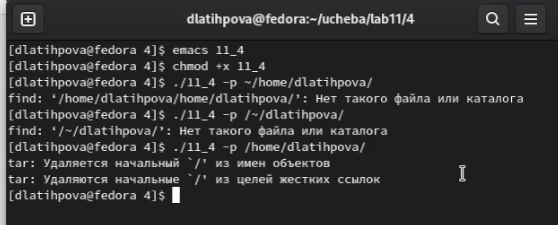


Рис. 9: Результат 4 скрипта (1)

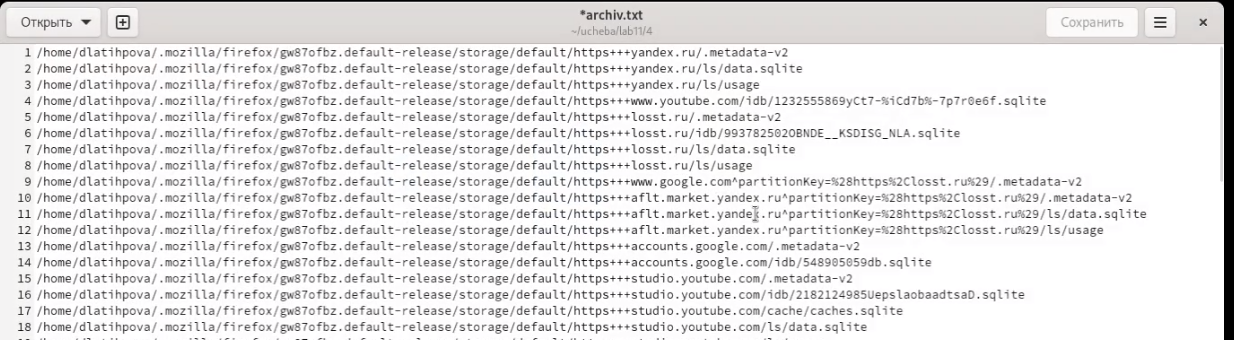


Рис. 10: Результат 4 задания (2)

# 4 Контрольные вопросы

1. Каково предназначение команды getopts?

Команда getopts является встроенной командой командной оболочки bash, предназначенной для разбора параметров сценариев. Она обрабатывает исключительно однобуквенные параметры как с аргументами, так и без них и этого вполне достаточно для передачи сценариям любых входных данных.

1. Какое отношение метасимволы имеют к генерации имён файлов?

При перечислении имен файлов текущего каталога можно использовать следующие символы: · \* — соответствует произвольной, в том числе и пустой строке; · ? — соответствует любому одному символу; · [c1-c1] — соответствует любому символу, лексикографически на ходящемуся между символами c1 и с2. · echo \* — выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды ls; · ls .c — выведет все файлы с последними двумя символами, равными .c. · echo prog.? — выдаст все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются prog. . · [a-z] — соответствует произвольному имени файла в текущем каталоге, начинающемуся с любой строчной буквы латинского алфавита.

1. Какие операторы управления действиями вы знаете?

Часто бывает необходимо обеспечить проведение каких-либо действий циклически и управление дальнейшими действиями в зависимости от результатов проверки некоторого условия. Для решения подобных задач язык программирования bash предоставляет Вам возможность использовать такие управляющие конструкции, как for, case, if и while. С точки зрения командного процессора эти управляющие конструкции являются обычными командами и могут использоваться как при создании командных файлов, так и при работе в интерактивном режиме. Команды, реализующие подобные конструкции, по сути дела являются операторами языка программирования bash. Поэтому при описании языка программирования bash термин оператор будет использоваться наравне с термином команда.

1. Какие операторы используются для прерывания цикла?

Прерывание цикла: операторы break, Continue, return и функция Abort.

1. Для чего нужны команды false и true?

true,: - всегда возвращает 0 в качестве кода выхода. false - всегда возвращает 1 в качестве кода выхода.

1. Что означает строка if test -f mani.$s, встреченная в командном файле?

Введенная строка означает условие существования файла mani.$s

1. Объясните различия между конструкциями while и until.

Если речь идет о 2-х параллельных действиях, то это while. когда мы показываем, что сначала делается 1-е действие. потом оно заканчивается при наступлении 2-го действия, применяем until.

# 5 Выводы

Я изучила основы программирования в оболочке ОС UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.