

Отчёта по лабораторной работе №12

Операционный Систем

Коне Сирики НФИБД-01-20

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	13

List of Tables

List of Figures

3.1	рисунок 1	7
3.2	рисунок 2	8
3.3	рисунок 3	8
3.4	рисунок 4	9
3.5	рисунок 5	9
3.6	рисунок 6	10
3.7	рисунок 7	10
3.8	рисунок 8	11
3.9	рисунок 9	11
3.10	рисунок 10	11

1 Цель работы

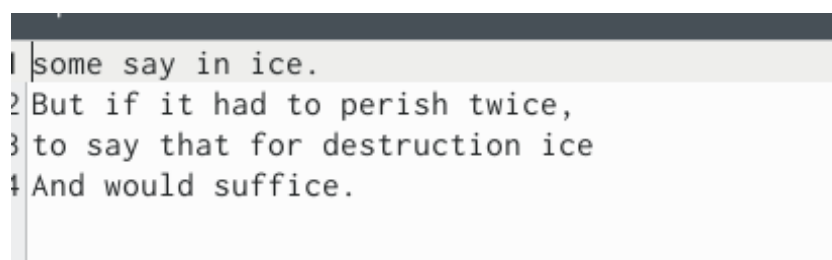
Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

2 Задание

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

3 Выполнение лабораторной работы

Ход работы: 1. Используя команды `getopts` `grep`, написала командный файл, который анализирует командную строку с ключами: `-i inputfile` — прочитать данные из указанного файла; `-o outputfile` — вывести данные в указанный файл; `-p` шаблон — 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`. (рис. 3.1)



```
1 | some say in ice.
2 But if it had to perish twice,
3 to say that for destruction ice
4 And would suffice.
```

Figure 3.1: рисунок 1

(рис. 3.2)

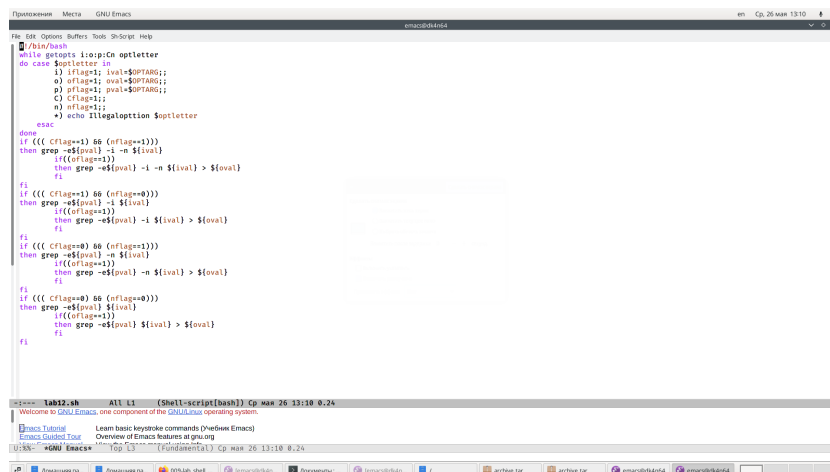


Figure 3.2: рисунок 2

(рис. 3.3)

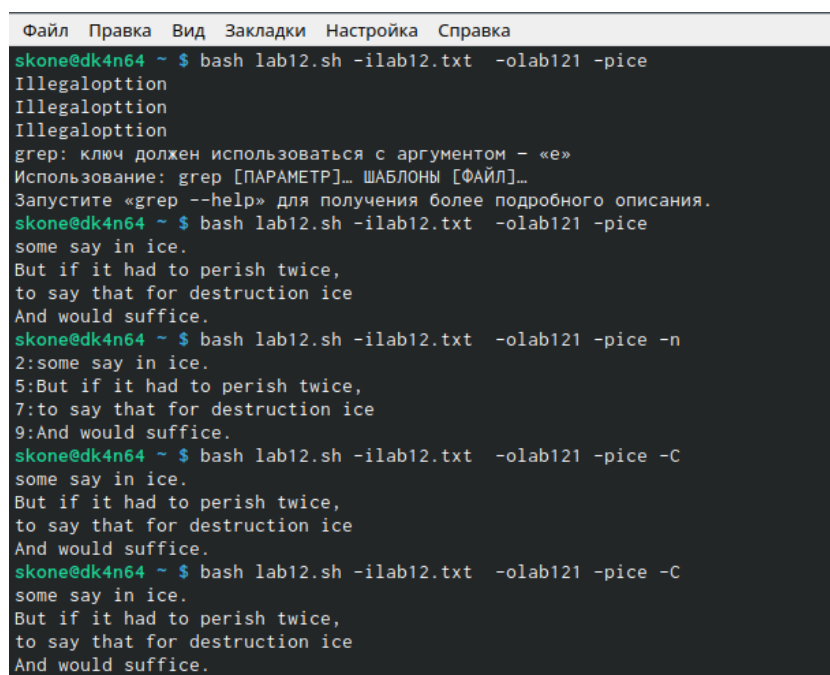
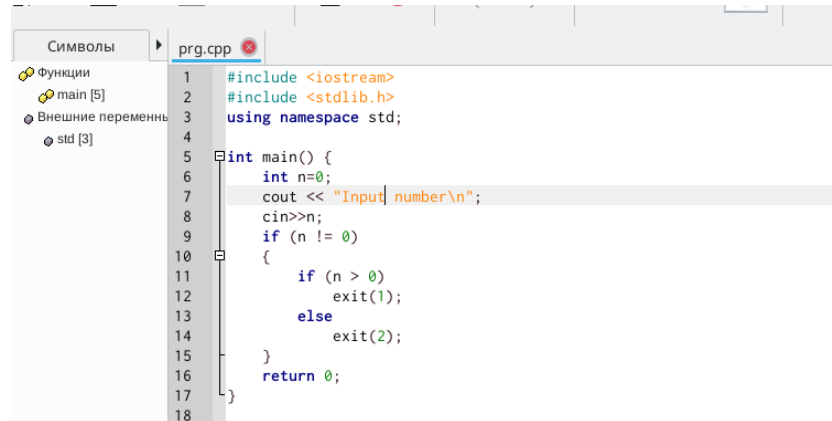


Figure 3.3: рисунок 3

2. Написал 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 о коде завершения в оболочку. Командный файл вызывает эту программу и, про-

анализировав с помощью команды \$?, выдаёт сообщение о том, какое число было введено. Код на C++:

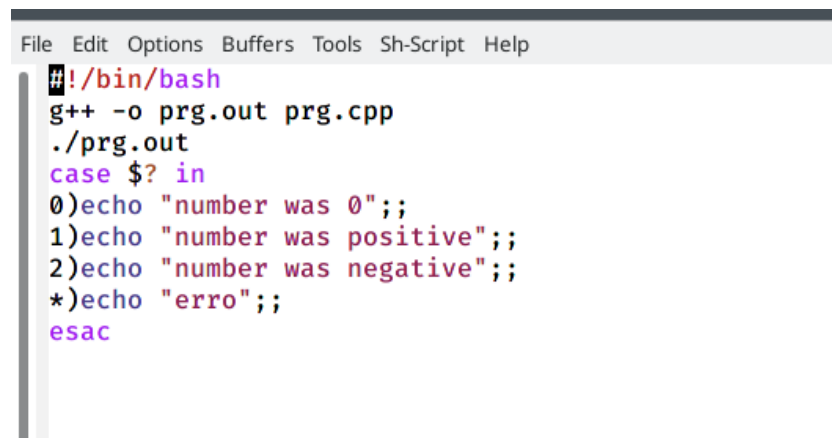
(рис. 3.4)



```
1 #include <iostream>
2 #include <stdlib.h>
3 using namespace std;
4
5 int main() {
6     int n=0;
7     cout << "Input number\n";
8     cin>>n;
9     if (n != 0)
10    {
11        if (n > 0)
12            exit(1);
13        else
14            exit(2);
15    }
16    return 0;
17 }
18
```

Figure 3.4: рисунок 4

(рис. 3.5)



```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
g++ -o prg.out prg.cpp
./prg.out
case $? in
0)echo "number was 0";;
1)echo "number was positive";;
2)echo "number was negative";;
*)echo "erro";;
esac
```

Figure 3.5: рисунок 5

(рис. 3.6)

```

collect2: ошибка: выполнение ld завершилось с кодом возврата 1
skone@dk4n64 ~/prg/prg $ g++ -o prg.out prg.cpp
skone@dk4n64 ~/prg/prg $ ./lab12.sh
number was 0
skone@dk4n64 ~/prg/prg $ ./lab12.sh
./lab12.sh: строка 3: ./out: Нет такого файла или каталога
erro
skone@dk4n64 ~/prg/prg $ ./lab12.sh
Input number
3
number was positive
skone@dk4n64 ~/prg/prg $ -5
bash: -5: команда не найдена
skone@dk4n64 ~/prg/prg $ ./lab12.sh
Input number
0
number was 0
skone@dk4n64 ~/prg/prg $ ./lab12.sh
Input number
-5
number was negative
skone@dk4n64 ~/prg/prg $ ./lab12.sh
Input number
10
number was positive
skone@dk4n64 ~/prg/prg $

```

Figure 3.6: рисунок 6

3. Написал 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 удаляет все созданные им файлы (если они существуют). Код скрипта:

(рис. 3.7)



```

emacs@dk4n64
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
typeset -i count=0;
for prov in *.tmp
do rm $prov
done
for ((count=1; count<$1+1; count++))
do
    touch $count.tmp
done

```

Figure 3.7: рисунок 7

(рис. 3.8)

```
skone@dk4n64 ~$ touch lab122.sh
skone@dk4n64 ~$ chmod +x lab122.sh
skone@dk4n64 ~$ ./lab122.sh
rm: невозможно удалить '*.*tmp': Нет такого файла или каталога
skone@dk4n64 ~$ ./lab122.sh 13
rm: невозможно удалить '*.*tmp': Нет такого файла или каталога
skone@dk4n64 ~$ ls|grep .tmp
1. tmp
10. tmp
11. tmp
12. tmp
13. tmp
1. tmp
2. tmp
3. tmp
4. tmp
5. tmp
6. tmp
7. tmp
8. tmp
9. tmp
skone@dk4n64 ~$ ./lab122.sh 6
skone@dk4n64 ~$ ls|grep .tmp
1. tmp
2. tmp
3. tmp
4. tmp
5. tmp
6. tmp
skone@dk4n64 ~$
```

Figure 3.8: рисунок 8

4. Написал командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировала его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовав команду find). Код скрипта:

(рис. 3.9)

```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
tar -cf 8.tar $@
tar -cf 8l.tar
find $@ -mtime -7 -exec tar -rf 8l.tar '{}' ';'

----- lab123.sh All L1 (Shell-script[bash]) Ср мая 26 13:41 0.43
```

Figure 3.9: рисунок 9

(рис. 3.10)

```

$ ./lib123.sh /lib121.txt
[usr] Удаляете каталог /usr из всех объектов
[usr] /lib121.txt: Функция stat неопределена в опциях. Нет такого файла или каталога
[usr] Завершение lib123 с уведомлением о несоответствии из-за возможной ошибки.
[usr] Робот отказ от создания пустого архива
[usr]Подготовка star --lib123 --info star --source для
[usr] загрузки более подробных сведений.
[usr] $ ./lib121.txt: Нет такого файла или каталога
$ ./lib121.sh -p 1
1.tsp 5.tsp 5.tsp 5.tsp 8.tsp
hostname test Hostname test
acemid=/lib123/report/template-master archive test
acemid=/presentation/markdown/template-master host test

```

Figure 3.10: рисунок 10

Контрольные вопросы:

1. Каково предназначение команды `getopts`? Команда `getopts` осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных.
2. Какое отношение метасимволы имеют к генерации имён файлов? Метасимволы позволяют обращаться к файлам, не зная их точных имён. Например, `*` соответствует произвольной, в том числе и пустой строке, `?` соответствует любому одинарному символу, `[c1-c1]` соответствует любому символу, лексикографически находящемуся между символами `c1` и `c2`.
3. Какие операторы управления действиями вы знаете? К операторам управления относятся `if`, `while`, `until`, `for`, `case`.
4. Какие операторы используются для прерывания цикла? Для прерывания цикла используются команды `break` и `continue`. `Break` завершает выполнение цикла, а `continue` завершает текущую итерацию цикла.
5. Для чего нужны команды `false` и `true`? Эти команды используются совместно с операторами управления. Команда `true` всегда возвращает код завершения, равный нулю (т.е. истина), команда `false` всегда возвращает код завершения, не равный нулю (т. е. ложь).
6. Что означает строка `if test -f mans/i.s, ?, mans/Si.Ss`
7. Объясните различия между конструкциями `while` и `until`. Обе команды имеют схожий синтаксис: `while` и `until`. Цикл с `While` выполняется до тех пор, пока не вернёт ненулевой код завершения, а `until`, пока не вернёт код завершения, равный нулю.

4 Выводы

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