

Лабораторная работа №11

Беличева Д.М.; НКНбд-01-21

¹RUDN University, Moscow, Russian Federation

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

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;
 - `-р`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 `-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).

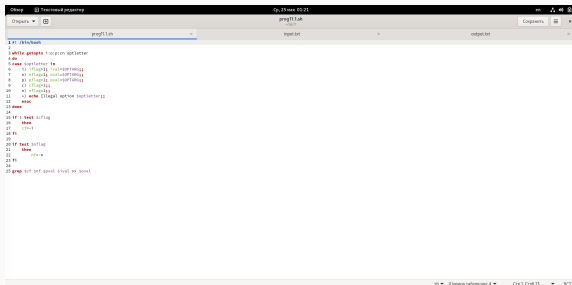
Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек: - оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;

- С-оболочка (или csh) — надстройка на оболочкой Борна, использующая С-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
- оболочка Корна (или ksh) — напоминает оболочку С, но операторы управления программой совместимы с операторами оболочки Борна;
- BASH — сокращение от Bourne Again Shell (опять оболочка Борна).

POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ. Стандарты POSIX разработаны комитетом IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости различных UNIX/Linux-подобных операционных систем и переносимости прикладных программ на уровне исходного кода. POSIX-совместимые оболочки разработаны на базе оболочки Корна. Рассмотрим основные элементы программирования в оболочке `bash`. В других оболочках большинство команд будет совпадать с описанными ниже.

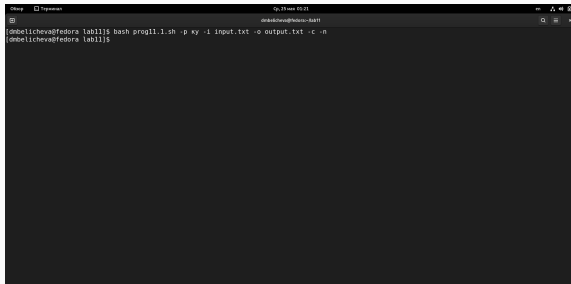
1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами:
 - `-iinputfile` — прочитать данные из указанного файла;
 - `-ooutputfile` — вывести данные в указанный файл;
 - `-р`шаблон — указать шаблон для поиска;
 - `-C` — различать большие и малые буквы;
 - `-n` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-р`. (рис. 1, 2, 3, 4)

Выполнение лабораторной работы



```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     char c;
7     cout << "Введите символ: ";
8     c = getChar();
9     cout << c;
10    return 0;
11 }
12
13 // Вывод: Введите символ: a
14 // Вывод: a
```

Figure 1: Первая программа

A terminal window with a dark background. The title bar shows 'Окно' and 'Терминал'. The terminal content shows a user prompt 'dmbelicheva@fedora lab11\$' followed by the command 'bash prog11.1.sh -p ky -i input.txt -o output.txt -c -n'. The command is executed, and the prompt returns. The window has standard Linux window controls (minimize, maximize, close) in the top right corner.

```
Окно  Терминал  09.08.2024 15:21
dmbelicheva@fedora lab11$ bash prog11.1.sh -p ky -i input.txt -o output.txt -c -n
dmbelicheva@fedora lab11$
```

Figure 2: Вызов программы в терминале

Выполнение лабораторной работы

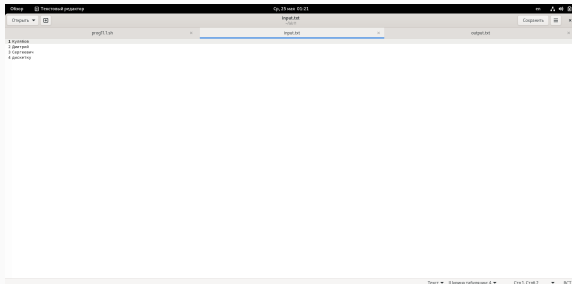


Figure 3: Результат

Выполнение лабораторной работы

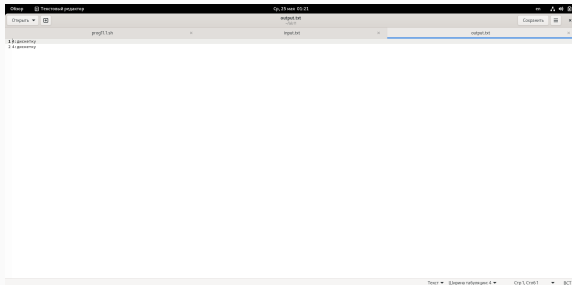
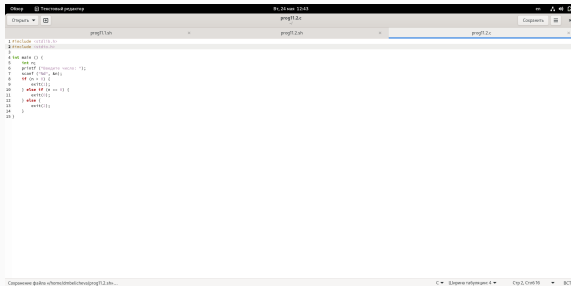


Figure 4: Результат

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. (рис. 5, 6, 7)

Выполнение лабораторной работы



The screenshot shows a C++ IDE window titled "Исходный редактор" (Source Editor) with the file "prog7.2.cpp" open. The code is as follows:

```
1 #include <iostream.h>
2 #include <stdlib.h>
3
4 int main () {
5     int x;
6     printf ("Введите число: ");
7     scanf ("%d", &x);
8     if (x > 0) {
9         printf ("
10     } else if (x == 0) {
11         printf ("
12     } else {
13         printf ("
14     }
15 }
```

The status bar at the bottom indicates the file path "С:\программы\src\prog7.2.cpp...", the current line and column "Стр. 2, Кол. 18", and the encoding "BCI".

Figure 5: Вторая программа

Выполнение лабораторной работы

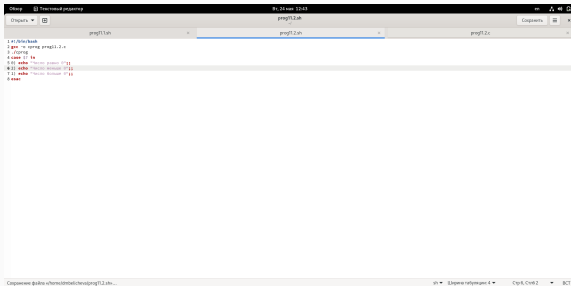
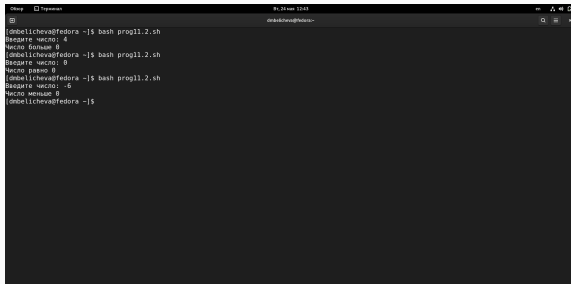


Figure 6: Вторая программа

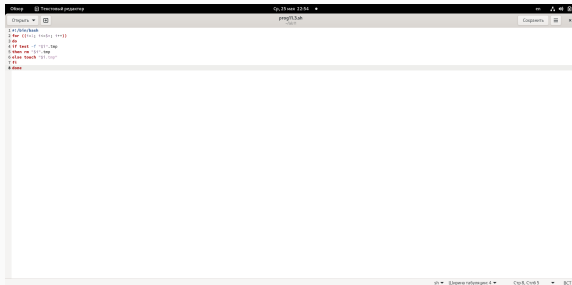


```
Обзор  Терминал  9:28 Nov 11:43
osbelicheva@fedora:~$
[osbelicheva@fedora ~]$ bash prog11.2.sh
Введите число: 4
Число больше 0
[osbelicheva@fedora ~]$ bash prog11.2.sh
Введите число: 0
Число равно 0
[osbelicheva@fedora ~]$ bash prog11.2.sh
Введите число: 5
Число меньше 0
[osbelicheva@fedora ~]$
```

Figure 7: Результат

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). (рис. 8, 9)

Выполнение лабораторной работы

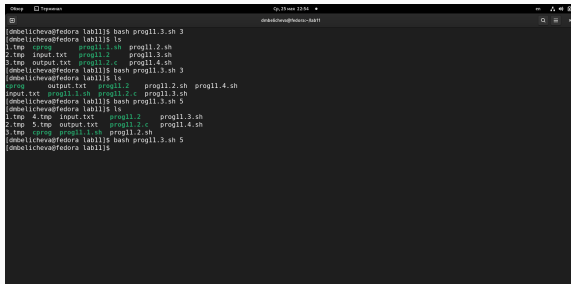


The screenshot shows the Qt Creator IDE interface. The top toolbar includes buttons for opening files, saving, and running. The menu bar shows 'Файл', 'Правка', 'Настройка', 'Справка'. The toolbar also has icons for 'Отладка' (Debug) and 'Запуск' (Run). The status bar at the bottom indicates '31', 'Среднее выражение: 4', 'C++8, C++9', and 'BCI'.

```
1 #include <iostream>
2 int main() {
3     int a, b;
4     if (a < b) {
5         std::cout << "a < b" << std::endl;
6     } else {
7         std::cout << "a >= b" << std::endl;
8     }
9     return 0;
10 }
```

Figure 8: Третья программа

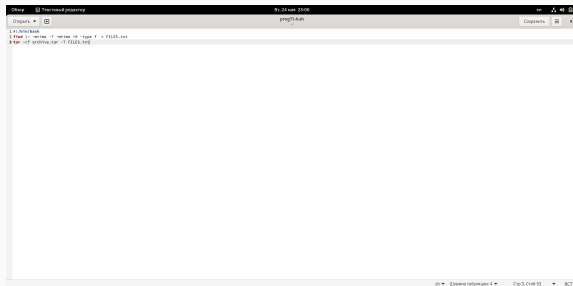
Выполнение лабораторной работы



```
dbelicheva@fedora lab11$ bash prog11.3.sh 3
dbelicheva@fedora lab11$ ls
1.tmp  cprog  prog11.1.sh  prog11.2.sh
2.tmp  input.txt  prog11.2    prog11.3.sh
3.tmp  output.txt  prog11.2.c  prog11.4.sh
dbelicheva@fedora lab11$ bash prog11.3.sh 3
dbelicheva@fedora lab11$ ls
prog11.1.sh  prog11.2    prog11.2.sh  prog11.4.sh
input.txt  prog11.1.sh  prog11.2.c  prog11.3.sh
dbelicheva@fedora lab11$ bash prog11.3.sh 5
dbelicheva@fedora lab11$ ls
1.tmp  4.tmp  input.txt  prog11.2    prog11.3.sh
2.tmp  5.tmp  output.txt  prog11.2.c  prog11.4.sh
3.tmp  cprog  prog11.1.sh  prog11.2.sh
dbelicheva@fedora lab11$ bash prog11.3.sh 5
dbelicheva@fedora lab11$
```

Figure 9: Результат

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`). (рис. 10, 11, 12)

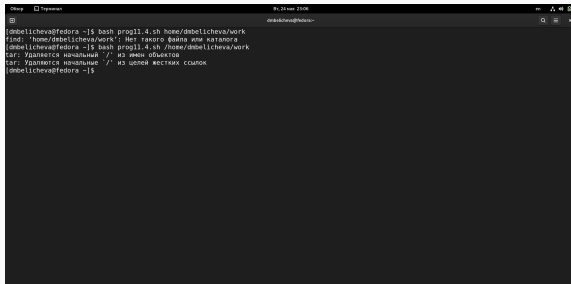


The screenshot shows a terminal window titled "Terminal" with a dark theme. The window contains the following C code:

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Enter n: ");
6     scanf("%d", &n);
7     int sum = 0;
8     for (int i = 1; i <= n; i++) {
9         sum += i * i;
10    }
11    printf("Sum of squares: %d\n", sum);
12    return 0;
13 }
```

The terminal output shows the program running successfully, with the prompt "Enter n:" and the output "Sum of squares: 10" (assuming n=3). The status bar at the bottom indicates the file path "C:\Program Files\Git\bin\cmd.exe" and the current directory "C:\Program Files\Git\bin\".

Figure 10: Четвертая программа



```
dbelicheva@fedora ~$ bash prog11.4.sh /home/dbelicheva/work
find: 'home/dbelicheva/work': Нет такого файла или каталога
dbelicheva@fedora ~$ bash prog11.4.sh /home/dbelicheva/work
tar: Удаляется начальный '/' из имен объектов
tar: Удаляется начальный '/' из имен жестких ссылок
dbelicheva@fedora ~$
```

Figure 11: Вызов программы в терминале

Выполнение лабораторной работы

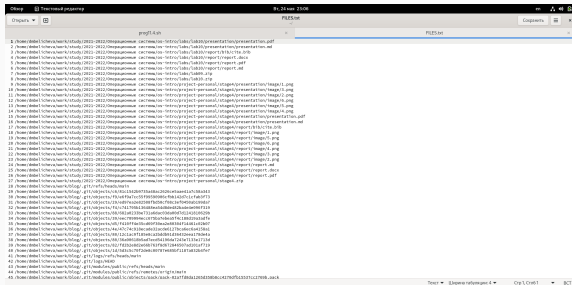


Figure 12: Результат

В процессе выполнения данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

1. Лабораторная работа № 10. Программирование в командном процессоре ОС UNIX. Командные файлы [Электронный ресурс]. URL: <https://esystem.rudn.ru/>.

спасибо за внимание!