

Лабораторная работа №12

Беличева Д.М.;НКНбд-01-21

¹RUDN University, Moscow, Russian Federation

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

1. Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (`> /dev/tty#`, где `#` — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработать программу так, чтобы имела возможность взаимодействия трёх и более процессов.

2. Реализовать команду `man` с помощью командного файла. Изучите содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. Командный файл должен получать в виде аргумента командной строки название команды и в виде результата выдавать справку об этой команде или сообщение об отсутствии справки, если соответствующего файла нет в каталоге `man1`.
3. Используя встроенную переменную `$RANDOM`, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтите, что `$RANDOM` выдаёт псевдослучайные числа в диапазоне от 0 до 32767.

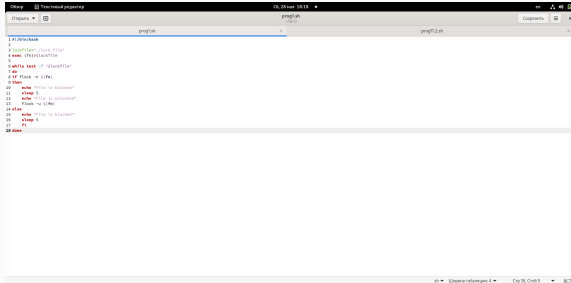
Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек:

- оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;
- C-оболочка (или csh) — надстройка на оболочкой Борна, использующая C-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
- оболочка Корна (или ksh) — напоминает оболочку C, но операторы управления программой совместимы с операторами оболочки 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 [`@Prog:bash`].

1. Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (`> /dev/tty#`, где `#` — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработать программу так, чтобы имелась возможность взаимодействия трёх и более процессов. (рис. 1, 2)

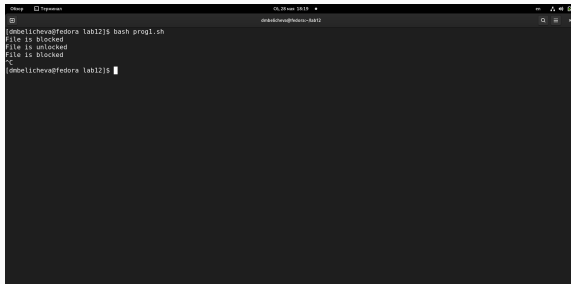
Выполнение лабораторной работы



The screenshot shows a code editor window titled "03.09.2018" with a file named "prog1.c". The code is written in C and includes comments in Russian. The code defines a function "blackoff" that takes a file name as an argument and opens it in append mode. It then writes the text "blackoff" to the file. The main function calls "blackoff" with the argument "prog1.c".

```
1 #include <stdio.h>
2
3 #define FILE_NAME "prog1.c"
4 #define FILE_MODE "a"
5
6 void blackoff(const char *file)
7 {
8     FILE *f;
9     if ((f = fopen(file, FILE_MODE)) == NULL)
10         return;
11     fprintf(f, "blackoff\n");
12     fclose(f);
13 }
14
15 int main()
16 {
17     blackoff(FILE_NAME);
18     return 0;
19 }
```

Figure 1: Текст первой программы



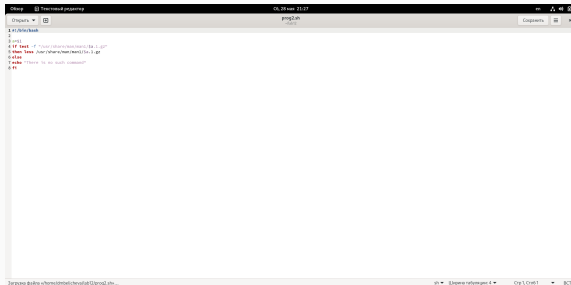
```
dbelicheva@fedora lab12]$ bash prog1.sh
File is blocked
File is unlocked
File is blocked
^C
dbelicheva@fedora lab12]$
```

The image shows a terminal window titled "dbelicheva@fedora lab12". The user has executed the command `bash prog1.sh`. The script's output is displayed on the following lines: `File is blocked`, `File is unlocked`, and `File is blocked`. The execution is then interrupted by the user pressing the `^C` (Ctrl-C) key, as indicated by the prompt `^C` on the next line. The terminal window's title bar includes standard Linux window controls and a system clock showing "03:20 Wed 18/10".

Figure 2: Результат

2. Реализовать команду `man` с помощью командного файла. Изучите содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. Командный файл должен получать в виде аргумента командной строки название команды и в виде результата выдавать справку об этой команде или сообщение об отсутствии справки, если соответствующего файла нет в каталоге `man1`. (рис. 3, 4, 5)

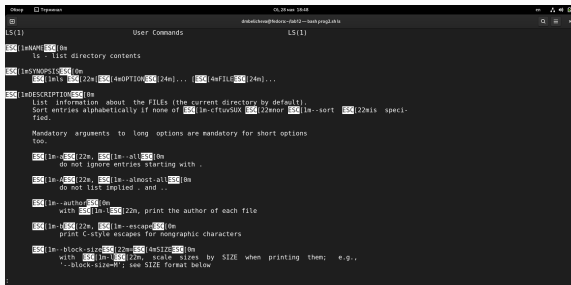
Выполнение лабораторной работы



```
1 #include <stdio.h>
2
3 int main()
4 {
5     if (test) {
6         printf("Test is true\n");
7     }
8     else {
9         printf("Test is false\n");
10    }
11 }
```

Figure 3: Текст второй программы

Выполнение лабораторной работы



```
ls(1)                                User Commands                                ls(1)

[55] [1mNAME[55] 0m
ls - list directory contents

[55] [1mSYNOPSIS[55] 0m
[55] [1m[55] [1m[55] [22m[55] [40PTION[55] [24m]... [55] [40FILE[55] [24m]...

[55] [1mDESCRIPTION[55] 0m
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of [55] [1m-ctvuSUX[55] [22mnor [55] [1m--sort[55] [22mis specified.

Mandatory arguments to long options are mandatory for short options too.

[55] [1m- [55] [22m, [55] [1m--all[55] 0m
do not ignore entries starting with .

[55] [1m- [55] [22m, [55] [1m--almost-all[55] 0m
do not list implied . and ..

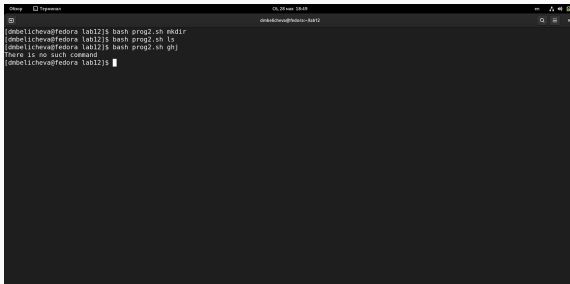
[55] [1m--author[55] 0m
with [55] [1m-[55] [22m, print the author of each file

[55] [1m- [55] [22m, [55] [1m--escape[55] 0m
print C-style escapes for nongraphical characters

[55] [1m--block-size[55] [55] [22m[55] [40SIZE[55] 0m
with [55] [1m-[55] [22m, scale sizes by SIZE when printing them; e.g.,
'--block-size=M'; see SIZE format below
```

Figure 4: Результат

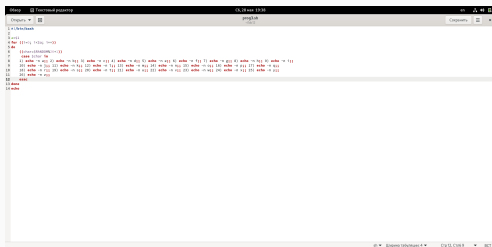
Выполнение лабораторной работы

A terminal window titled 'dbelicheva@fedora lab12' with a dark background. The window shows a series of commands and their outputs. The commands are: 'bash prog2.sh mkdir', 'bash prog2.sh ls', and 'bash prog2.sh ghj'. The outputs are: an empty directory listing for 'mkdir', a single file 'ls' for 'ls', and an error message 'There is no such command' for 'ghj'. The prompt '[dbelicheva@fedora lab12]\$' is visible at the end of each line.

```
[dbelicheva@fedora lab12]$ bash prog2.sh mkdir
[dbelicheva@fedora lab12]$ bash prog2.sh ls
[dbelicheva@fedora lab12]$ bash prog2.sh ghj
There is no such command
[dbelicheva@fedora lab12]$
```

Figure 5: Результат

3. Используя встроенную переменную \$RANDOM, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтите, что \$RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767. (рис. 6, 7)



```
#!/bin/bash
# Генерация случайной последовательности букв латинского алфавита

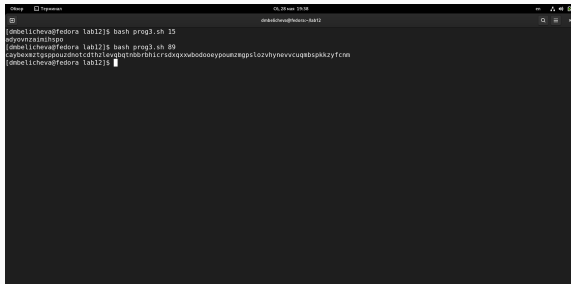
# Инициализация массива для хранения букв
letters="abcdefghijklmnopqrstuvwxyz"

# Генерация случайной последовательности букв
for i in $(seq 1 10); do
    # Генерация случайного индекса от 0 до 25
    index=$((RANDOM % 26))

    # Вывод буквы из массива
    echo -n "${letters:index:1}"
done
```

Figure 6: Текст третьей программы

Выполнение лабораторной работы



A terminal window titled 'Ch_20 lab 15' with a dark background. The prompt is '[dbelicheva@fedora lab12]'. The first command is 'bash prog3.sh 15', which outputs 'edyovnzelnihspo'. The second command is 'bash prog3.sh 89', which outputs a long alphanumeric string: 'Caybexaztqspouzdnotcdth2levabqtnb0rth1crsdxqxvbdooeypounzaggpslozhynevvcuqabspkhzyfcm'. The prompt returns to '[dbelicheva@fedora lab12]`.

```
[dbelicheva@fedora lab12]$ bash prog3.sh 15
edyovnzelnihspo
[dbelicheva@fedora lab12]$ bash prog3.sh 89
Caybexaztqspouzdnotcdth2levabqtnb0rth1crsdxqxvbdooeypounzaggpslozhynevvcuqabspkhzyfcm
[dbelicheva@fedora lab12]$
```

Figure 7: Результат

В процессе выполнения этой лабораторной работы я продолжила осваивать программирование на `bash`.

Спасибо за внимание!