

Лабораторная работа №10

Программирование в командном процессоре ОС UNIX. Командные
файлы

Бабенко Артём Сергеевич

Содержание

Цель работы	3
Теоретическое введение	4
Ход работы	5
Вывод	12
Контрольные вопросы	13

Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы.

Теоретическое введение

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:

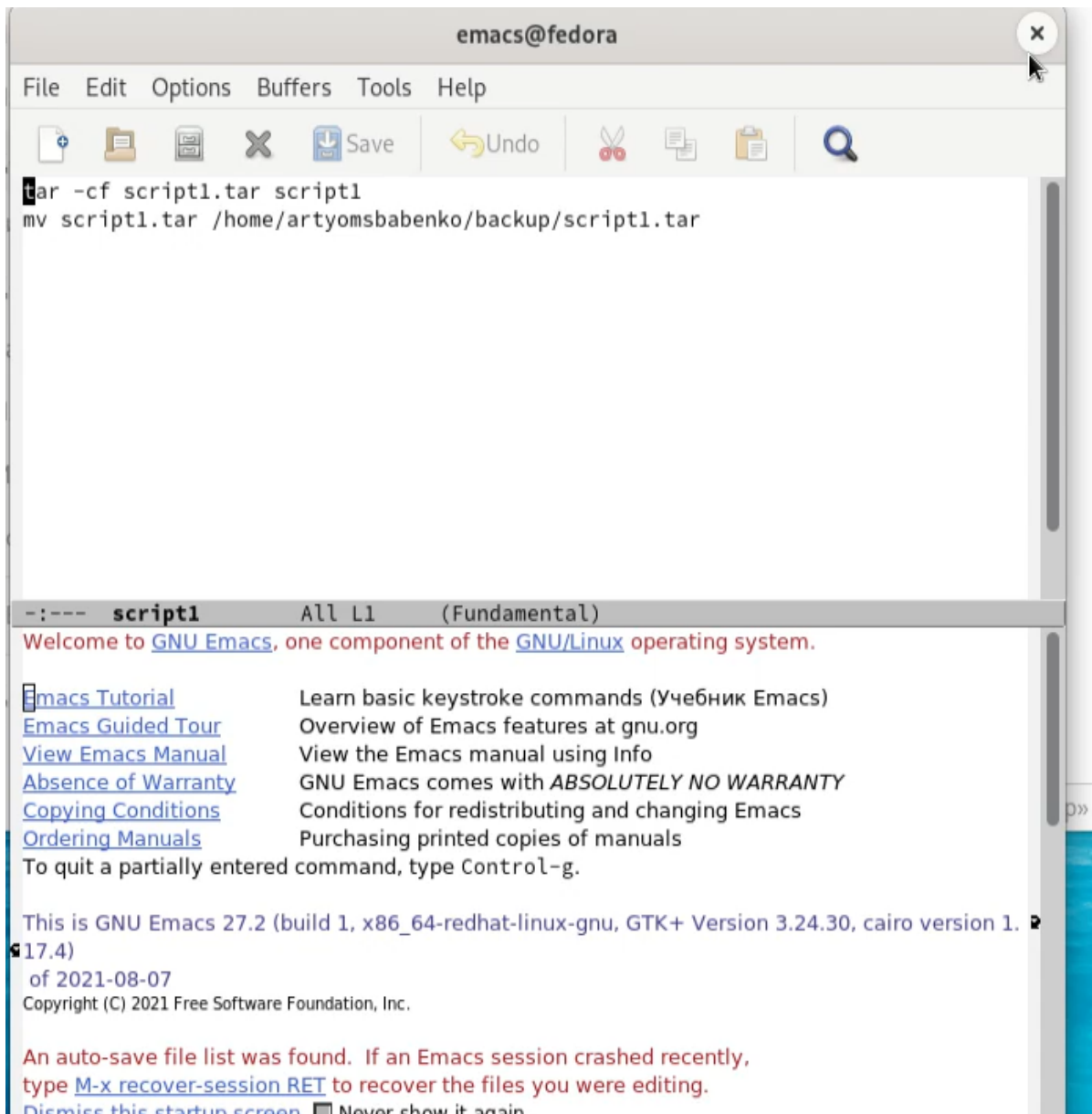
- оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;
- С-оболочка (или csh) — надстройка на оболочкой Борна, использующая С-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
- оболочка Корна (или ksh) — напоминает оболочку С, но операторы управления программой совместимы с операторами оболочки Борна;
- BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек С и Корна (разработка компании Free Software Foundation).

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.

Ход работы

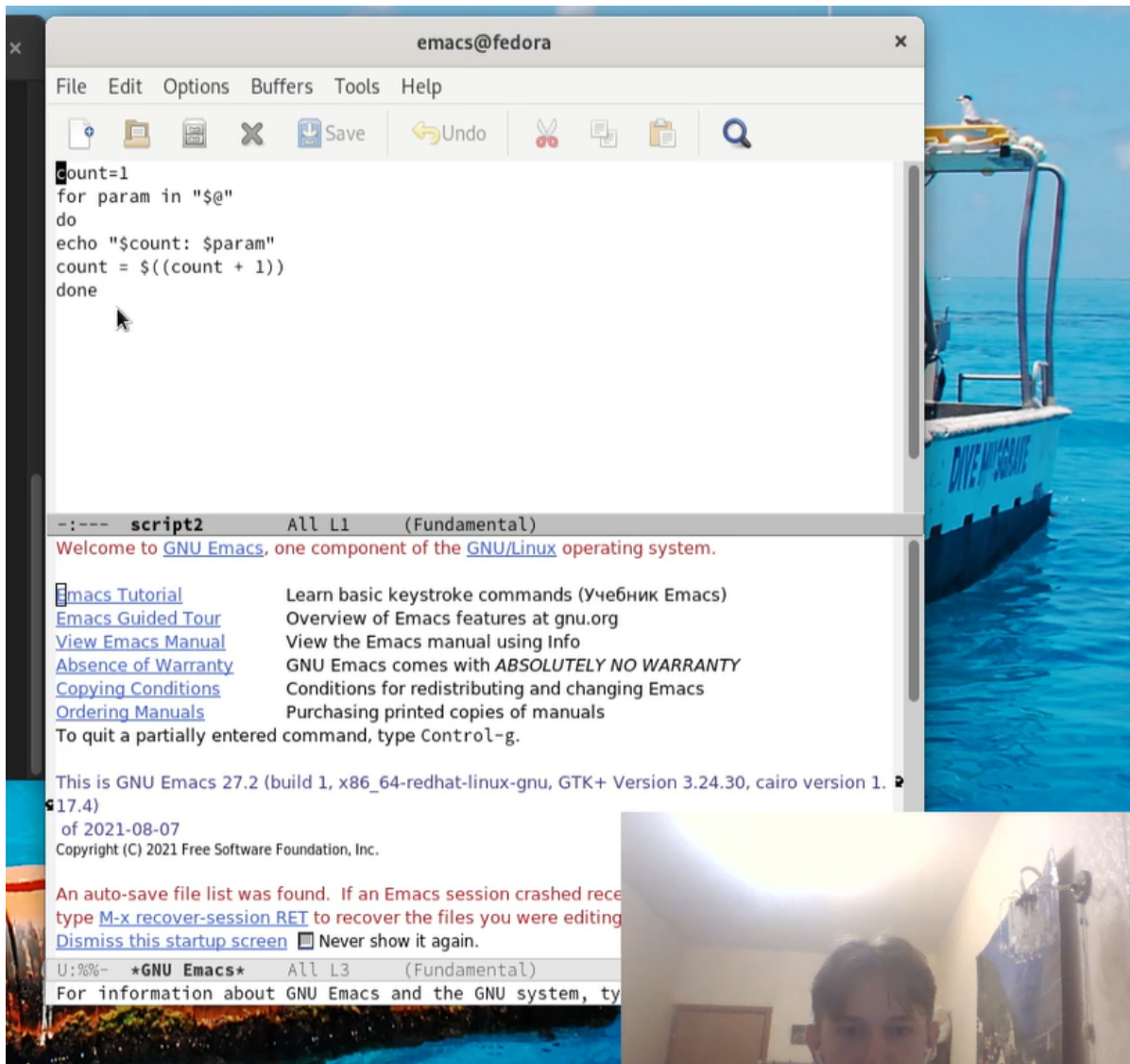
1. Написал скрипт, который при запуске будет делать резервную копию самого себя в другую директорию backup в моём домашнем каталоге. Файл архивировал архиватором tar (рис.1,2).



```
[artyomsbabenko@fedora ~]$ emacs script1
[artyomsbabenko@fedora ~]$ rmdir backup
[artyomsbabenko@fedora ~]$ mkdir backup
[artyomsbabenko@fedora ~]$ chmod +x script1
[artyomsbabenko@fedora ~]$ ./script1
[artyomsbabenko@fedora ~]$ cd backup/
[artyomsbabenko@fedora backup]$ ls
script1.tar
[artyomsbabenko@fedora backup]$ tar -xf script1.tar
[artyomsbabenko@fedora backup]$ ls
script1 script1.tar
[artyomsbabenko@fedora backup]$ emacs script1
```

Рис.1,2. Выполнение задания №1

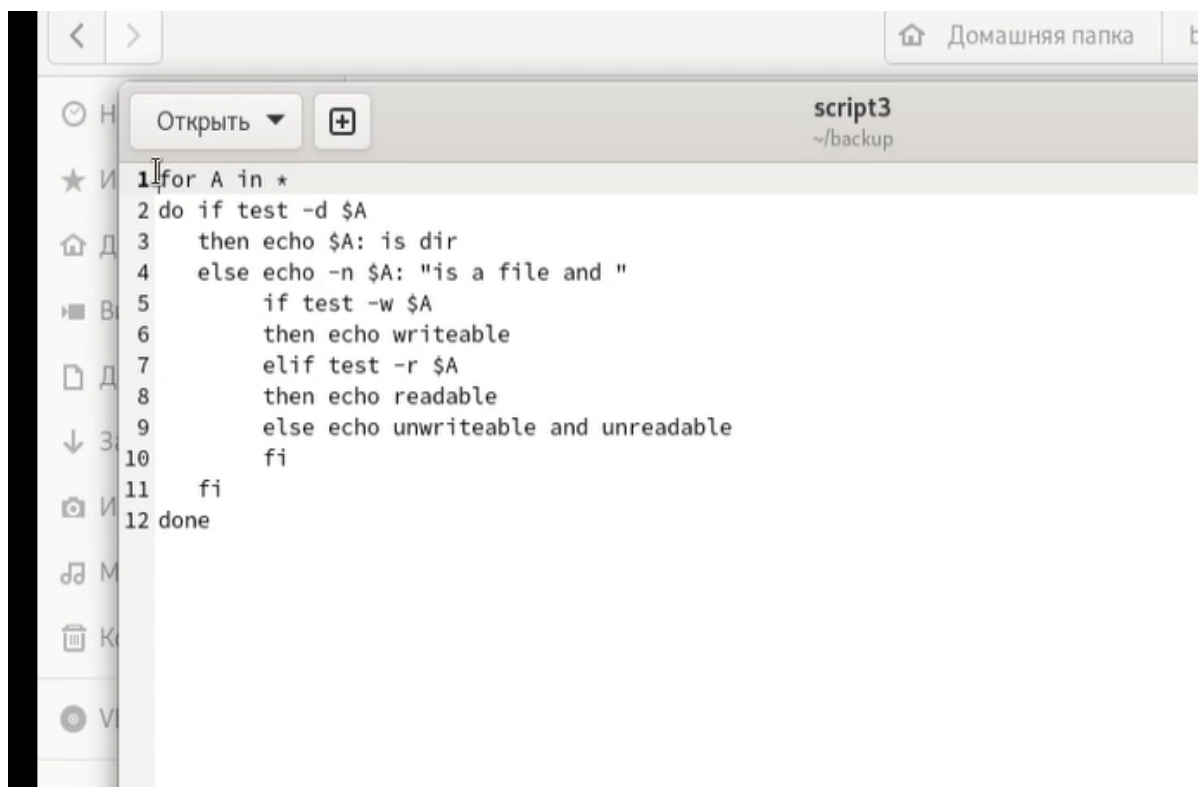
2. Написал пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять (рис.3,4).

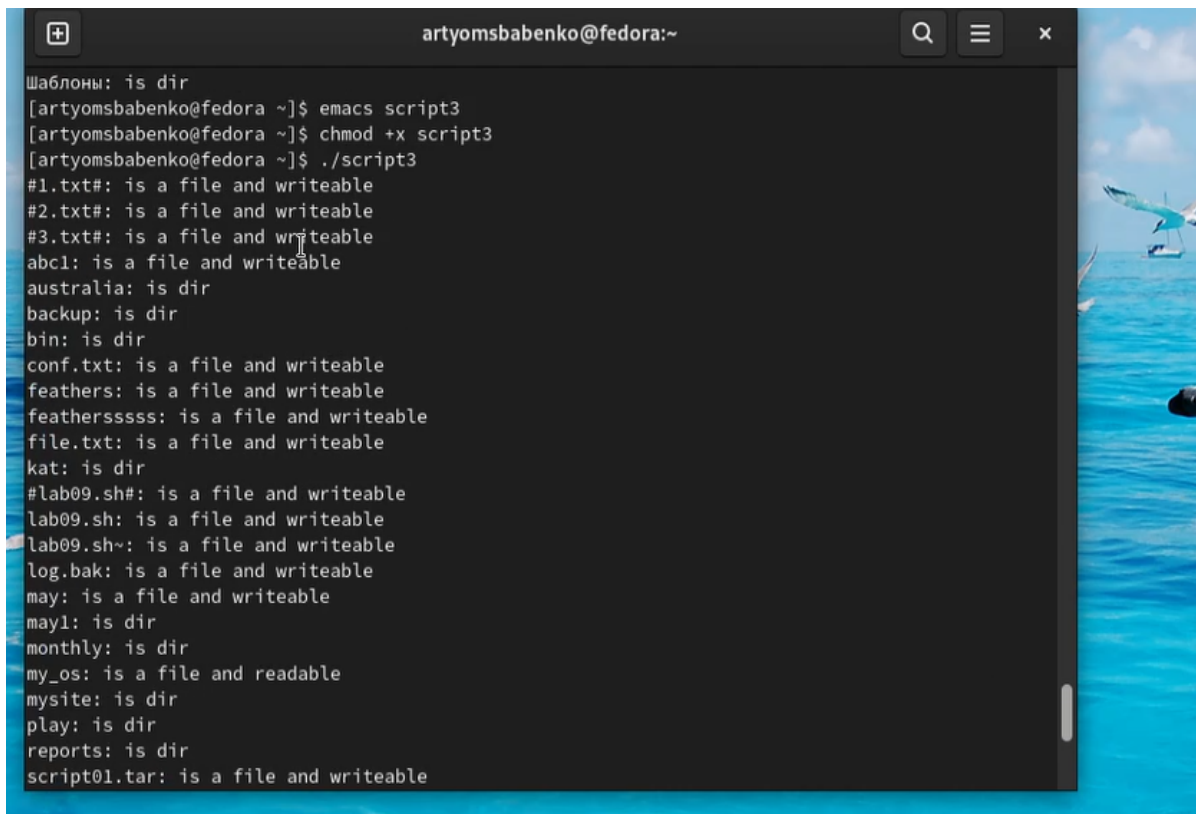



```
[artyomsbabenko@fedora backup]$ emacs script2
[artyomsbabenko@fedora backup]$ chmod +x script2
[artyomsbabenko@fedora backup]$ ./script2 1 2 3 4 5 6 7 8 9 10 11 12 13
1: 1
2: 2
3: 3
4: 4
5: 5
6: 6
7: 7
8: 8
9: 9
10: 10
11: 11
12: 12
13: 13
[artyomsbabenko@fedora backup]$
```

Рис.3,4. Выполнение задания №2

3. Написал командный файл — аналог команды `ls` (без использования самой этой команды и команды `dir`), чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога (рис.5,6).

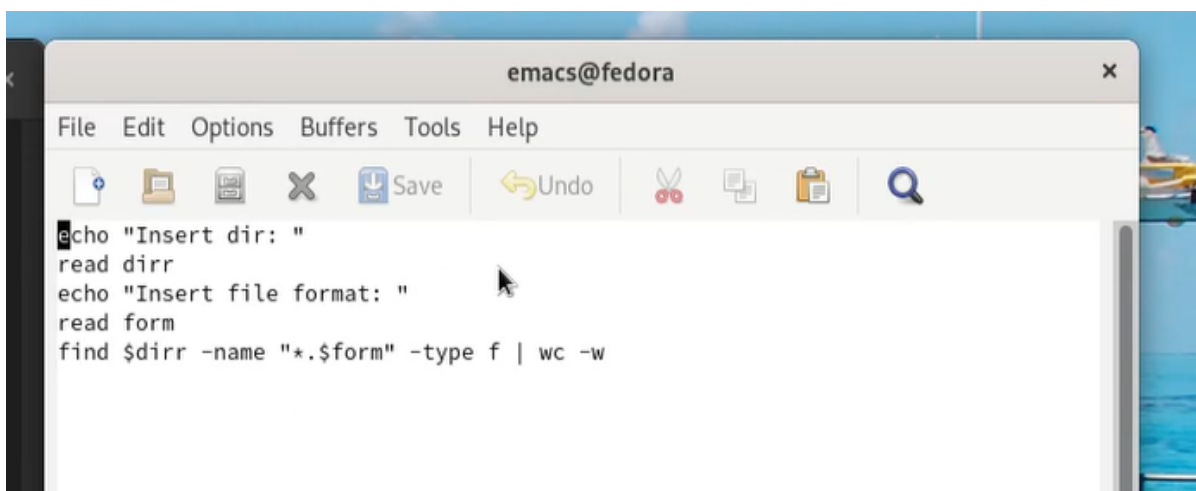




```
Шаблоны: is dir
[artyomsbabenko@fedora ~]$ emacs script3
[artyomsbabenko@fedora ~]$ chmod +x script3
[artyomsbabenko@fedora ~]$ ./script3
#1.txt#: is a file and writeable
#2.txt#: is a file and writeable
#3.txt#: is a file and writeable
abc1: is a file and writeable
australia: is dir
backup: is dir
bin: is dir
conf.txt: is a file and writeable
feathers: is a file and writeable
feathersssss: is a file and writeable
file.txt: is a file and writeable
kat: is dir
#lab09.sh#: is a file and writeable
lab09.sh: is a file and writeable
lab09.sh~: is a file and writeable
log.bak: is a file and writeable
may: is a file and writeable
may1: is dir
monthly: is dir
my_os: is a file and readable
mysite: is dir
play: is dir
reports: is dir
script01.tar: is a file and writeable
```

Рис.5,6. Выполнение задания №3

4. Написал командный файл, который получает в качестве аргумента командной строки формат файла и вычисляет количество таких файлов в указанной директории (рис.7,8).



```
emacs@fedora
File Edit Options Buffers Tools Help
[Icons: Save, Undo, Cut, Copy, Paste, Find]
echo "Insert dir: "
read dirr
echo "Insert file format: "
read form
find $dirr -name "*.$form" -type f | wc -w
```

```
[artyomsbabenko@fedora ~]$ chmod +x script4
[artyomsbabenko@fedora ~]$ ./script4
Insert dir:
/home/artyomsbabenko/
Insert file format:
txt
9
[artyomsbabenko@fedora ~]$ emacs script4
```

Рис.7,8. Выполнение задания №4

Вывод

Я изучил основы программирования в оболочке ОС UNIX/Linux. Научился писать небольшие командные файлы.

Контрольные вопросы

1. Командный процессор (командная оболочка, интерпретатор команд `shell`) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера.
2. POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ.
3. Команда `let` является показателем того, что последующие аргументы представляют собой выражение, подлежащее вычислению. Команда `read` позволяет читать значения переменных со стандартного ввода.
4. Для облегчения программирования можно записывать условия оболочки `bash` в двойные скобки — `(())`.
5. Такие символы, как `' < > * ? | " &`, являются метасимволами и имеют для командного процессора специальный смысл. Снятие специального смысла с метасимвола называется экранированием метасимвола.
6. Экранирование может быть осуществлено с помощью предшествующего метасимволу символа `\`, который, в свою очередь, является метасимволом.
7. Если использовать `typeset -i` для объявления и присвоения переменной, то при последующем её применении она станет целой. Изъять переменную из программы можно с помощью команды `unset`. Для создания массива используется команда `set` с флагом `-A`.

8. Команда `shift` позволяет удалять первый параметр и сдвигает все остальные на места предыдущих. При использовании в командном файле комбинации символов `$#` вместо неё будет осуществлена подстановка числа параметров, указанных в командной строке при вызове данного командного файла на выполнение.

Вот ещё несколько специальных переменных, используемых в командных файлах:

- `$*` — отображается вся командная строка или параметры оболочки;
- `$?` — код завершения последней выполненной команды;
- `$$` — уникальный идентификатор процесса, в рамках которого выполняется командный процесс.