

# Российский университет дружбы народов

Факультет физико-математических и естественных наук

## Лабораторная работа № 12. Программирование в командном процессоре ОС UNIX. Расширенное программирование

- Имя : исса гадир
- Студенческий билет : 1032218267
- Группа : нфибд-01-21

---

### Цель работы:

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

---

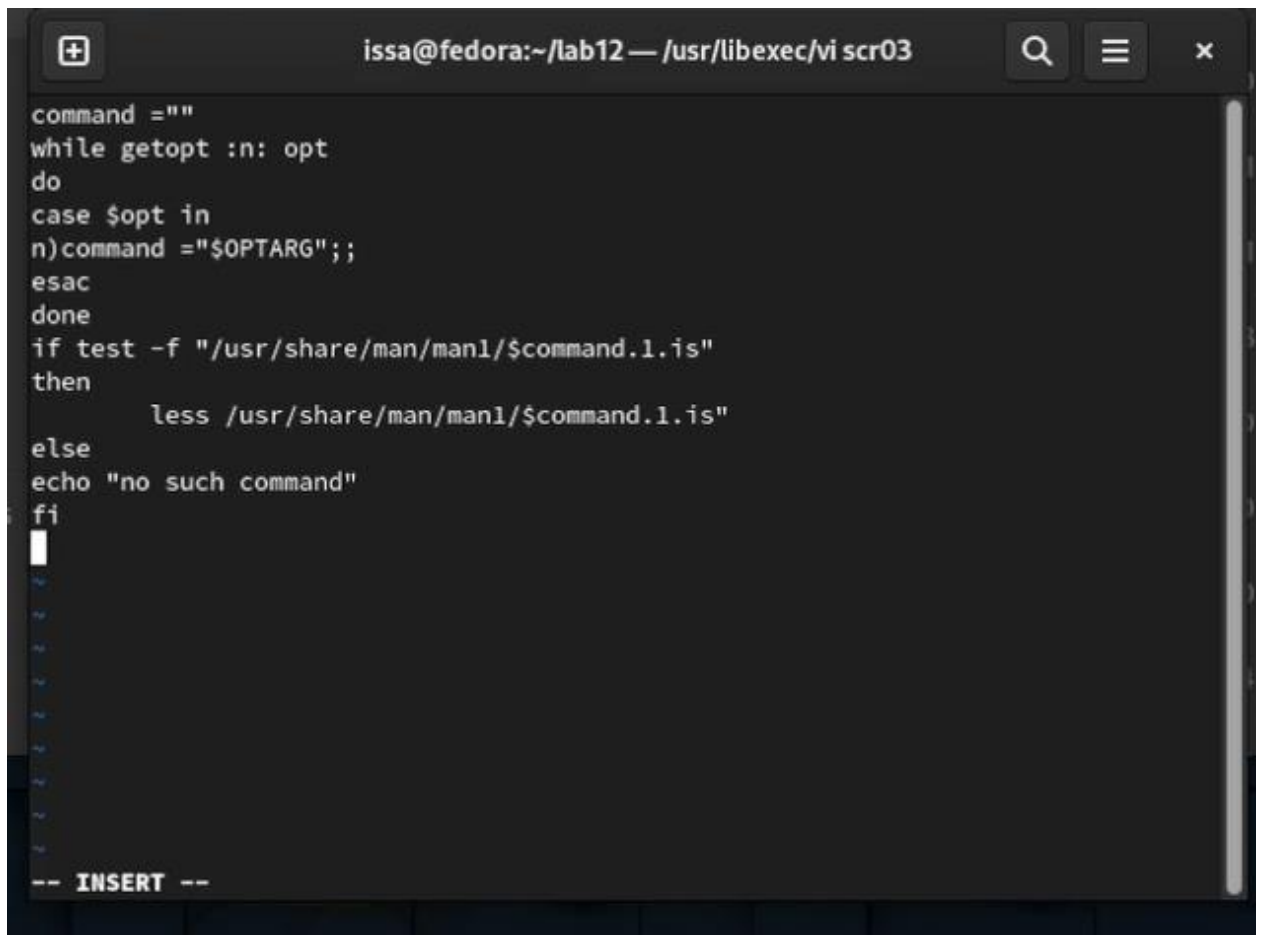
### выполнения работы

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.

```
lockfile="locking.file"
exec {fn}"$lockfile"
if test -f "$lockfile"
then
    while [ 1!=0 ]
    do
        if flock -n ${fn}
        echo "file was locked"
        sleep 4
        echo "unlocking"
        flock -u ${fn}
        else
        echo "file already locked"
        sleep 3
        fi
    done
fi

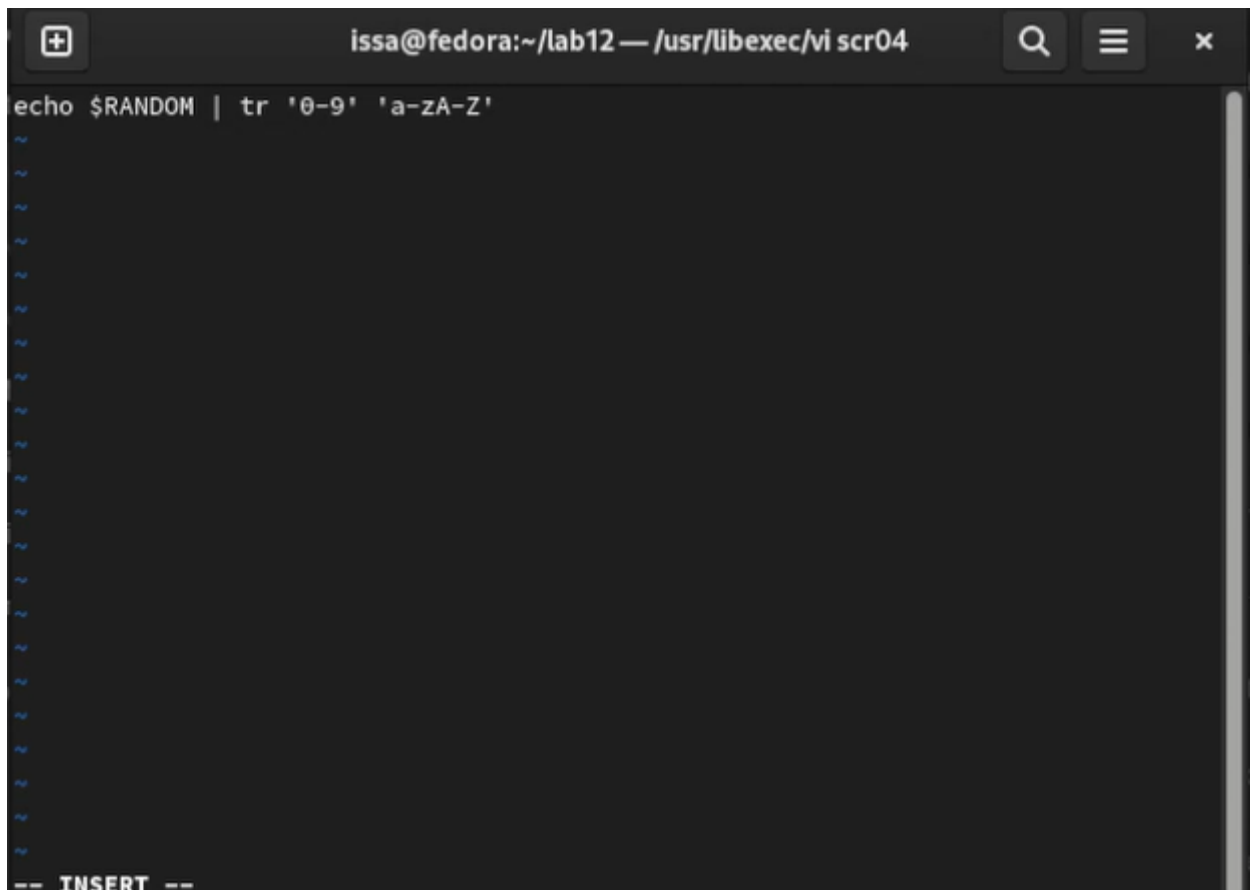
-- INSERT --
```

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.

A screenshot of a terminal window with a dark background. The window title bar shows the user 'issa' at 'fedora' in the directory '~/lab12', editing the file '/usr/libexec/vi scr03'. The terminal displays a shell script in vi editor mode. The script defines a 'command' variable, loops through command-line options using 'getopt', and checks if a corresponding manual page exists in '/usr/share/man/man1/'. If found, it uses 'less' to view it; otherwise, it prints 'no such command'. The editor is in 'INSERT' mode, indicated by the status line at the bottom.

```
command =""
while getopt :n: opt
do
case $opt in
n)command ="$OPTARG";;
esac
done
if test -f "/usr/share/man/man1/$command.1.is"
then
    less /usr/share/man/man1/$command.1.is"
else
echo "no such command"
fi
-- INSERT --
```

- 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.



The screenshot shows a terminal window with a dark background. The title bar at the top reads "issa@fedora:~/lab12 — /usr/libexec/vi scr04". The terminal content shows the command `echo $RANDOM | tr '0-9' 'a-zA-Z'` followed by a series of lines of random alphanumeric characters. At the bottom, the text `-- INSERT --` is visible.

---

Контрольные вопросы:

**1. Найдите синтаксическую ошибку в следующей строке : `while [ $ 1 ! – " exit " ]`**

\*Строковый тип не может быть использован \*

**2. Как объединить ( конкатенация ) несколько строк в одну ? `str3 = "\ (str1\ ) str2 "`**

**3. Найдите информацию об утилите `seq` . Какими иными способами можно реализовать её функционал при программировании на `bash` ?**

*используется для идентификации и аннотирования сайта вставки мутатора ,используемого при создании ресурсов VonpMu, Может быть реализован с помощью конвейера*

**4. Какой результат даст вычисление выражения `$ ((10/3))` ? 3**

**5. Укажите кратко основные отличия командной оболочки `zsh` от `bash`.**

*повторяющееся определение*

**6. Укажите кратко основные отличия командной оболочки zsh от bash.**

Баш	Зш
Bash – это оболочка по умолчанию для Linux, выпущенная взамен Bourne Shell.	Оболочка Z построена поверх оболочки bash и представляет собой расширенную версию bash с множеством новых функций.
Bash читает файл .bashrc в интерактивной оболочке без входа в систему и .bash_profile в оболочках входа.	Zsh читает .zshrc в интерактивной оболочке и .zprofile в оболочке входа.
Bash использует экранирование с обратной косой чертой.	Zsh использует процентное экранирование.
Bash не имеет встроенного подстановочного знака.	Zsh имеет встроенное расширение подстановочных знаков.
Не имеет параметров настройки.	Zsh имеет множество фреймворков, обеспечивающих настройку.

**7. Сравните язык bash с какими-либо языками программирования. Какие преимущества у bash по сравнению с ними? Какие недостатки?**

Оболочка — это язык программирования высокого уровня, что означает, что вам не нужно беспокоиться о сложных задачах, таких как управление памятью. Это облегчает изучение языка системного программирования, такого как C или C++. Программы оболочки, как правило, пишутся быстрее, чем соответствующие программы на языке C, и их часто легче отлаживать. Однако программы на C почти всегда работают быстрее и эффективнее. Таким образом, сценарии оболочки и программирование на C используются для очень разных задач. Для быстрого написания относительно коротких инструментов оболочка является гораздо лучшим выбором, но для крупных проектов системного программирования явно лучше подходит C.