

Программирование в командном процессоре ОС  
UNIX.

Работу выполнил:  
Рыбалко Тимофей Александрович  
НБИбд-02-24

Российский университет дружбы народов, Москва,  
Россия

# Введение

- Освоить работу с файловой системой Linux необходимо.
- Умение работать с git и GitHub важно для IT-специалистов.

# Цель работы

- Освоить применение управляющих конструкций (ветвлений и циклов) в командных файлах UNIX.
- Изучить практическое использование команд `getopts`, `grep`, `find`, `tar` и механизма обработки кодов возврата (\$?).

# Техническое обеспечение

- Виртуальная машина: VirtualBox или QEMU.
- Операционная система: Linux (дистрибутив Fedora).
- Требования: Intel Core i3-550, 4 ГБ ОЗУ, 80 ГБ свободного места.

# Поиск по шаблону

- Демонстрирует работу скрипта для поиска строк, содержащих "hello" (без учёта регистра), с выводом номеров строк (-n):

```
foot
GNU nano 8.3      search.sh      Изменё
^[[I/bin/bash

# Инициализация переменных
input_file=""
output_file=""
pattern=""
case_sensitive=0
line_numbers=0

# Обработка аргументов командной строки
while getopts "i:o:p:Cn" opt; do
  case $opt in
    i) input_file="$OPTARG" ;;
    o) output_file="$OPTARG" ;;
    p) pattern="$OPTARG" ;;
    C) case_sensitive=1 ;;
    n) line_numbers=1 ;;
    *) echo "Использование: $0 [-i inputfile] [-o outfile] [-p pattern] [-C]"
       exit 1 ;;
  esac
done

# Проверка обязательных параметров
if [ -z "$input_file" ] || [ -z "$pattern" ]; then
  echo "Необходимо указать входной файл (-i) и шаблон (-p)"
  exit 1
fi

# Проверка существования входного файла
if [ ! -f "$input_file" ]; then
  echo "Файл $input_file не существует"
  exit 1
fi

# Построение команды grep
grep_cmd="grep"
if [ $case_sensitive -eq 0 ]; then
  grep_cmd="$grep_cmd -i"
fi

if [ $line_numbers -eq 1 ]; then
```

# Проверка числа

- Показывает взаимодействие С-программы (check\_number) и shell-скрипта. Программа определяет знак числа, а скрипт анализирует код возврата (\$?)

```
foot
GNU nano 8.3 check_number.c
#include <stdio.h>
#include <stdlib.h>

int main() {
    int number;
    printf("Введите число: ");
    scanf("%d", &number);

    if (number > 0) {
        printf("Число положительное\n");
        exit(1);
    } else if (number < 0) {
        printf("Число отрицательное\n");
        exit(2);
    } else {
        printf("Число равно нулю\n");
        exit(0);
    }
}
```

# Управление нумерованными файлами

- Скрипт создаёт 5 файлов (1.tmp–5.tmp) и удаляет их по команде clean

```
foot
GNU nano 8.3 file_creator.sh
#!/bin/bash

if [ "$1" = "clean" ]; then
    echo "Удаление всех .tmp файлов..."
    rm -f *.tmp
    echo "Файлы удалены"
    exit 0
fi

if [ -z "$1" ] || [ "$1" -le 0 ]; then
    echo "Использование: $0 N (где N - количество файлов для создания)"
    echo "Или: $0 clean (для удаления всех .tmp файлов)"
    exit 1
fi

for i in $(seq 1 "$1"); do
    touch "${i}.tmp"
done

echo "Создано $1 файлов с расширением .tmp"
```

# Архивирование новых файлов

- Скрипт запаковывает в архив (recent\_files.tar.gz) только файлы из test\_dir, изменённые за последние 7 дней

```
foot
GNU nano 8.3          archive_recent.sh          Изменё
#!/bin/bash

if [ -z "$1" ]; then
    echo "Использование: $0 директория [имя_архива]"
    exit 1
fi

dir="$1"
archive_name="${2:-recent_files.tar.gz}"

if [ ! -d "$dir" ]; then
    echo "Директория $dir не существует"
    exit 1
fi

# Находим файлы, измененные за последние 7 дней, и архивируем их
find "$dir" -type f -mtime -7 -print0 | tar -czvf "$archive_name" --null -T -

echo "Архив $archive_name создан с файлами, измененными за последние 7 дней"
```



# Ответы на вопросы:

- 1.getopts — разбирает аргументы командной строки (ключи и их значения) в shell-скриптах.
- 2.Метасимволы (\*, ?, []) — используются для шаблонного поиска и генерации имён файлов (например, \*.txt).
- 3.Операторы управления — if, case, for, while, until, &&, ||, break, continue.
- 4.Прерывание цикла — break (выход), continue (следующая итерация), exit (завершение скрипта).
- 5.false/true — возвращают код завершения 1 и 0, используются для управления условиями.
- 6.if test -f man\$s/\$i.\$s — проверяет, существует ли файл с именем вида manX/Y.X, где X и Y — значения переменных.
- 7.while vs until — while выполняется пока условие истинно, until — пока оно ложно (т.е. до первого совпадения).

# Заключение

- Освоены ключевые конструкции shell-программирования: ветвления (if/case), циклы (for/while), обработка аргументов (getopts).
- Приобретены практические навыки автоматизации задач: поиск файлов, обработка кодов возврата, пакетное создание/удаление файлов, архивирование с фильтрацией.