

Лабораторная работа номер 13

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Титков Ярослав Максимович

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	10
5	Контрольные вопросы	11

Список иллюстраций

3.1	Используя команды <code>getopts</code> <code>grep</code> , написал командный файл, который анализирует командную строку с ключами	7
3.2	Написал на языке Си программу,	7
3.3	Написал командный файл, создающий указанное число файлов . .	8
3.4	Написал командный файл, который работает с архивом	8
3.5	Командный файл	8
3.6	Компиляция всего	9

Список таблиц

1 Цель работы

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

2 Задание

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; `-r` — 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 `-r`.
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 `tag` 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`).

3 Выполнение лабораторной работы

```
#!/bin/bash

while getopts "i:op:Ca" opt; do
  case $opt in
    i) inputfile="$OPTARG" ;;
    o) outfile="$OPTARG" ;;
    p) pattern="$OPTARG" ;;
    C) case_sensitive=1 ;;
    n) with_line_numbers=1 ;;
    *) echo "Usage: $0 -i inputfile -o outfile -p pattern [-C] [-n]" ; exit 1 ;;
  esac
done

if [ -z "$inputfile" ] || [ -z "$outfile" ] || [ -z "$pattern" ]; then
  echo "Missing required options"
  exit 1
fi

grep_opts=""
if [ "$with_line_numbers" = 1 ] || [ "$case_sensitive" = 1 ]; then
  grep_opts="$grep_opts -n"
  if [ "$case_sensitive" = 1 ]; then
    grep_opts="$grep_opts -i"
  fi
fi

grep $grep_opts "$pattern" "$inputfile" "$outfile"
```

Рис. 3.1: Используя команды `getopts` `grep`, написал командный файл, который анализирует командную строку с ключами

```
#include <stdio.h>
#include <stdlib.h>

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

    if (number > 0)
        exit(1);
    else if (number < 0)
        exit(2);
    else
        exit(0);
}
```

Рис. 3.2: Написал на языке Си программу,

```
/bin/bash
./check_number
code $?

case $code in
0) echo "Число равно нулю" ;;
1) echo "Число больше нуля" ;;
2) echo "Число меньше нуля" ;;
*) echo "Неизвестный код: $code" ;;
esac
```

Рис. 3.3: Написал командный файл, создающий указанное число файлов

```
/bin/bash
if [ -x "$1" ]; then
echo "Использование: $0 N [--delete]"
exit 1
fi

N=$1

if [ "$2" == "--delete" ]; then
for ((i=1; i<=N; i++)); do
file="$i.tap"
if [ -f "$file" ]; then
rm "$file"
echo "Удален: $file"
else
echo "Файл не существует: $file"
fi
done
else
for ((i=1; i<=N; i++)); do
touch "$i.tap"
echo "Создан: $i.tap"
done
fi
```

Рис. 3.4: Написал командный файл, который работает с архивом

```
/bin/bash

if [ -x "$1" ]; then
echo "Использование: $0 <директория>"
exit 1
fi

dir="$1"
timestamp=$(date +%Y%m%d_%H%M%S)
archive_name="archive_${timestamp}.tar.gz"

find "$dir" -type f -mtime -7 -print0 | tar -czvf "$archive_name" --null -T -
echo "Архив создан: $archive_name"
```

Рис. 3.5: Командный файл


```
yitkov@yitkov:~/python/work1$ cd
yitkov@yitkov:~/work1$ $ cd work13/
yitkov@yitkov:~/work13$ nano grep_wrapper.sh
yitkov@yitkov:~/work13$ chmod +x grep_wrapper.sh
yitkov@yitkov:~/work13$ nano check_number.c
yitkov@yitkov:~/work13$ gcc check_number.c -o check_number
yitkov@yitkov:~/work13$ nano check_number.sh
yitkov@yitkov:~/work13$ chmod +x check_number.sh
yitkov@yitkov:~/work13$ nano tmp_files.sh
yitkov@yitkov:~/work13$ chmod +x tmp_files.sh
yitkov@yitkov:~/work13$ nano archive_recent.sh
yitkov@yitkov:~/work13$ chmod +x archive_recent.sh
yitkov@yitkov:~/work13$
```

Рис. 3.6: Компиляция всего

4 Выводы

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

5 Контрольные вопросы

1. `getopts` обрабатывает аргументы командной строки в скриптах, разбирая опции и их параметры.
2. Метасимволы (*, ?, []) используются в шаблонах имен файлов для их генерации по заданному паттерну.
3. Операторы управления действиями: `if`, `case`, `for`, `while`, `until`, `select`.
4. Для прерывания цикла: `break` (выход из цикла), `continue` (переход к следующей итерации).
5. `false` возвращает код ошибки (1), `true` — успешный код (0); используются для управления потоком выполнения.
6. `if test -f mans/i.s, mans/i.s` и является ли он обычным файлом.
7. `while` выполняет цикл, пока условие истинно, `until` — пока условие ложно.