Отчёт по лабораторной работе №13

Петлин Артём Дмитриевич

Содержание

Список иллюстраций

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;
- -pшаблон — указать шаблон для поиска;
- -C — различать большие и малые буквы;
- -n — выдавать номера строк.

а затем ищет в указанном файле нужные строки, определяемые ключом -p.

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 (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 tar 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 Теоретическое введение

Командный процессор (командная оболочка, интерпретатор команд shell) — это про- грамма, 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;
- BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек C и Корна (разработка компании Free Software Foundation).

4 Выполнение лабораторной работы

```
#!bin/bash

SCRIPT_DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )"

# Инициализируем переменные
input_file=""
output_file=""
pattern=""
case_sensitive=0
line_numbers=0

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 ;;
  esac
done

grep_flags=""
if [ $case_sensitive -eq 0 ]; then
  grep_flags+=" -i"
fi
if [ $line_numbers -eq 1 ]; then
  grep_flags+=" -n"
fi

if [ -z "$output_file" ]; then
  grep $grep_flags "$pattern" "$input_file"
else
  grep $grep_flags "$pattern" "$input_file" > "$output_file"
fi
```

Рис. 4.1: код скрипта

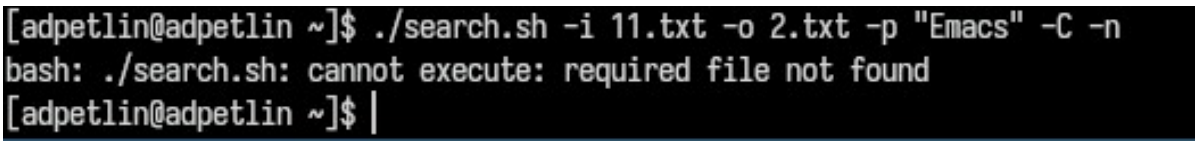

Рис. 4.2: работа скрипта

Пишем скрипт для поиска строк в файле с ключами.

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


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

  if (a == 0) {
    exit(0);
  }
  if (a > 0) {
    exit(1);
  }
  if (a < 0) {
    exit(2);
  }
}
```

Рис. 4.3: код программы на Си

```
#!/bin/bash

gcc check_number.c -o check_number
./check_number

case $? in
    0) echo "Число равно нулю";;
    1) echo "Число больше нуля";;
    2) echo "Число меньше нуля";;
esac
```

Рис. 4.4: код скрипта

```
[adpetlin@adpetlin ~]$ ./checkn_run.sh
Введите число: 10
Число больше нуля
[adpetlin@adpetlin ~]$ ./checkn_run.sh
Введите число: 0
Число равно нулю
[adpetlin@adpetlin ~]$ ./checkn_run.sh
Введите число: -1
Число меньше нуля
```

Рис. 4.5: работа скрипта

Пишем программу на Си для проверки числа и командный файл, который компилирует и запускает эту программу и анализирует код завершения ($?) и

выводит сообщение.

```
#!/bin/bash

if [ "$1" == "-d" ]; then
    rm -f *.tmp
    echo "Все .tmp файлы удалены"
else
    for ((i=1; i<=$1; i++)); do
        touch "$i.tmp"
    done
    echo "Создано $1 .tmp файлов"
fi
```

Рис. 4.6: код скрипта

```
[adpetlin@adpetlin backup]$ ls
backup_script_20250421_205537.tar.gz  tfiles.sh  tfiles.sh~
[adpetlin@adpetlin backup]$ ./tfiles.sh 7
Создано 7 .tmp файлов
[adpetlin@adpetlin backup]$ ls
1.tmp  2.tmp  3.tmp  4.tmp  5.tmp  6.tmp  7.tmp  backup_script_20250421_205537.tar.gz  tfiles.sh  tfiles.sh~
[adpetlin@adpetlin backup]$ ./tfiles.sh -d
Все .tmp файлы удалены
[adpetlin@adpetlin backup]$ ls
backup_script_20250421_205537.tar.gz  tfiles.sh  tfiles.sh~
[adpetlin@adpetlin backup]$
```

Рис. 4.7: работа скрипта

Пишем скрипт для создания и удаления временных файлов.

```
#!/bin/bash

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

find "$1" -type f -mtime -7 -print0 | tar -czvf backup.tar.gz --null -T -
echo "Архив backup.tar.gz создан."
```

Рис. 4.8: код скрипта

Рис. 4.9: работа скрипта

Пишем скрипт для архивации недавно измененных файлов.

5 Выводы

Мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Список литературы

1. Dash, P. Getting Started with Oracle VM VirtualBox / P. Dash. – Packt Publishing Ltd, 2013. – 86 cc.
2. Colvin, H. VirtualBox: An Ultimate Guide Book on Virtualization with VirtualBox. VirtualBox / H. Colvin. – CreateSpace Independent Publishing Platform, 2015. – 70 cc.
3. Vugt, S. van. Red Hat RHCSA/RHCE 7 cert guide : Red Hat Enterprise Linux 7 (EX200 and EX300) : Certification Guide. Red Hat RHCSA/RHCE 7 cert guide / S. van Vugt. – Pearson IT Certification, 2016. – 1008 cc.
4. Робачевский, А. Операционная система UNIX / А. Робачевский, С. Немнюгин, О. Стесик. – 2-е изд. – Санкт-Петербург : БХВ-Петербург, 2010. – 656 сс.
5. Немет, Э. Unix и Linux: руководство системного администратора. Unix и Linux / Э. Немет, Г. Снайдер, Т.Р. Хейн, Б. Уэйли. – 4-е изд. – Вильямс, 2014. – 1312 сс.
6. Колисниченко, Д.Н. Самоучитель системного администратора Linux : Системный администратор / Д.Н. Колисниченко. – Санкт-Петербург : БХВ-Петербург, 2011. – 544 сс.
7. Robbins, A. Bash Pocket Reference / A. Robbins. – O'Reilly Media, 2016. – 156 сс.