

Лабораторная работа №13

Отчёт

Андрюшин Никита Сергеевич

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	12
	Список литературы	13

Список иллюстраций

3.1	Код первой программы	7
3.2	Проверка работы первой программы	8
3.3	Код второй программы на С	8
3.4	Код второй программы	9
3.5	Проверка работы второй программы	9
3.6	Код третьей программы	10
3.7	Проверка работы третьей программы	10
3.8	Код четвёртой программы	11
3.9	Проверка работы четвёртой программы	11

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. [1]

2 Задание

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами, 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 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 `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 Выполнение лабораторной работы

Напишем код первой программы (рис. 3.1).

```
1      echo "Неверный параметр: $OPTARG" 1>&2
2      exit 1
3      ;;
4  : )
5      echo "Отсутствует значение для параметра: $OPTARG" 1>&2
6      exit 1
7      ;;
8  esac
9  done
10
11 if [ -z "$pattern" ]; then
12     echo "Не указан шаблон для поиска." 1>&2
13     exit 1
14 fi
15
16 if [ -z "$inputfile" ]; then
17     echo "Не указан входной файл." 1>&2
18     exit 1
19 fi
20
21 if [ "$case_sensitive" = true ]; then
22     grep_options+=" -i"
23 fi
24
25 if [ "$line_numbers" = true ]; then
26     grep_options+=" -n"
27 fi
28
29 grep $grep_options "$pattern" "$inputfile"
30
31 if [ ! -z "$outputfile" ]; then
32     grep $grep_options "$pattern" "$inputfile" > "$outputfile"
33 fi
```

Рис. 3.1: Код первой программы

И проверим работу (рис. 3.2).

```

[nsandryushin@nsandryushin lab13]$ ./1.sh -p "сушки"
Не указан входной файл.
[nsandryushin@nsandryushin lab13]$ ./1.sh -p "сушки" -i text
И сосала сушку
[nsandryushin@nsandryushin lab13]$ ./1.sh -p "сушки" -i text -n
5:И сосала сушку
[nsandryushin@nsandryushin lab13]$ ./1.sh -p "Сушки" -i text -n
-C
[nsandryushin@nsandryushin lab13]$ ./1.sh -p "Сушки" -i text -n
-C
5:И сосала сушку
[nsandryushin@nsandryushin lab13]$ 

```

Рис. 3.2: Проверка работы первой программы

Напишем код второй программы на С (рис. 3.3).

```

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

int main() {
    int number;

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

    if (number > 0) {
        printf("Число больше нуля\n");
        exit(1);
    } else if (number < 0) {
        printf("Число меньше нуля\n");
        exit(2);
    } else {
        printf("Число равно нулю\n");
        exit(0);
    }
}

```

Рис. 3.3: Код второй программы на С

И напишем код второй программы (рис. 3.4).

```
#!/bin/bash

./2|

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

Рис. 3.4: Код второй программы

И проверим работу (рис. 3.5).

```
[nsandryushin@nsandryushin lab13]$ ./2.sh
Введите число: 7
Число больше нуля
Число больше нуля
[nsandryushin@nsandryushin lab13]$ ./2.sh
Введите число: -5
Число меньше нуля
Число меньше нуля
[nsandryushin@nsandryushin lab13]$ ./2.sh 0
Введите число: 0
Число равно нулю
Число равно нулю
```

Рис. 3.5: Проверка работы второй программы

Напишем код третьей программы (рис. 3.6).

```

    for ((i=1; i<=$count; i++)); do
        touch "$i.tmp"
        echo "Создан файл $i.tmp"
    done
}

delete_files() {
    local count=$1
    for ((i=1; i<=$count; i++)); do
        if [ -e "$i.tmp" ]; then
            rm "$i.tmp"
            echo "Удален файл $i.tmp"
        fi
    done
}

if [ $# -eq 0 ]; then
    echo "Не указано количество файлов для создания"
    exit 1
fi

action=$1

case $action in
    create)
        create_files $2
        ;;
    delete)
        delete_files $2
        ;;
    *)
        echo "Неверное действие. Используйте 'create' для создания файлов или 'delete' для удаления файлов."
        exit 1
        ;;
esac

```

Рис. 3.6: Код третьей программы

И проверим работу (рис. 3.7).

```

[nsandryushin@nsandryushin lab13]$ ./3.sh create 5
Создан файл 1.tmp
Создан файл 2.tmp
Создан файл 3.tmp
Создан файл 4.tmp
Создан файл 5.tmp
[nsandryushin@nsandryushin lab13]$ ./3.sh delete 5
Удален файл 1.tmp
Удален файл 2.tmp
Удален файл 3.tmp
Удален файл 4.tmp
Удален файл 5.tmp

```

Рис. 3.7: Проверка работы третьей программы

Напишем код четвёртой программы (рис. 3.8).

```
#!/bin/bash

directory=$1
output_archive="archive.tar.gz"
threshold_days=7

if [ -z "$directory" ]; then
    echo "Укажите директорию в качестве аргумента"
    exit 1
fi

if [ ! -d "$directory" ]; then
    echo "Указанная директория не существует"
    exit 1
fi

# Используем find для поиска файлов, измененных менее чем threshold_days назад,
# и передаем их в tar
find "$directory" -type f -mtime -$threshold_days -print0 | tar --null -czf "$output_archive" --files-from -
echo "Архивация завершена. Архив создан: $output_archive"
```

Рис. 3.8: Код четвёртой программы

И проверим работу (рис. 3.9).

```
[nsandryushin@nsandryushin lab13]$ ./4.sh .
Архивация завершена. Архив создан: archive.tar.gz
```

Рис. 3.9: Проверка работы четвёртой программы

4 Выводы

В результате лабораторной работы появились навыки обработки аргументов командной строки и написаны 4 программы

Список литературы

1. Kulyabov. Лабораторная работа № 13. Программирование в командном процессоре ОС UNIX. Ветвления и циклы. RUDN.