



**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**  
**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ**  
**ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Факультет автоматизации и информатики  
Кафедра автоматизированных систем управления

**ЛАБОРАТОРНАЯ РАБОТА №4**  
по курсу «ОС Linux»  
Вариант 5

Студент      ПИ-21-1

\_\_\_\_\_

(подпись, дата)

Морозов Д. С.

Руководитель

\_\_\_\_\_

(подпись, дата)

Кургасов В.В.

Липецк 2023

## **Цели работы:**

- изучить основные возможности языка программирования высокого уровня Shell;
- получить навыки написания и использования скриптов.

## **Порядок выполнения работы**

1. Используя команды ECHO, PRINTF, вывести информационные сообщения на экран.
2. Присвоить переменной A целочисленное значение. Просмотреть значение переменной A.
3. Присвоить переменной B значение переменной A. Просмотреть значение переменной B.
4. Присвоить переменной C значение “путь до своего каталога”. Перейти в этот каталог с использованием переменной.
5. Присвоить переменной D значение “имя команды”, а именно, команды PATE. Выполнить эту команду, используя значение переменной.
6. Присвоить переменной E значение “имя команды”, а именно, команды просмотра содержимого файла, просмотреть содержимое переменной. Выполнить эту команду, используя значение переменной.
7. Присвоить переменной F значение “имя команды”, а именно, сортировки содержимого текстового файла. Выполнить эту команду, используя значение переменной.

Написать скрипты, при запуске которых выполняются следующие действия:

1. Программа запрашивает значение переменной, а затем выводит значение этой переменной.
2. Программа запрашивает имя пользователя, затем здоровается с ним, используя значение введенной переменной.
3. Программа запрашивает значения двух переменных, вычисляет сумму (разность, произведение, деление) этих переменных. Результат выводится на экран (использовать команды: а) EXPR; б) BC).
4. Вычислить объем цилиндра. Исходные данные запрашиваются программой. Результат выводится на экран.

5. Используя позиционные параметры, отобразить имя программы, количество аргументов командной строки, значение каждого аргумента командной строки.
6. Используя позиционный параметр, отобразить содержимое текстового файла, указанного в качестве аргумента командной строки. После паузы экран очищается.
7. Используя оператор FOR, отобразить содержимое текстовых файлов текущего каталога поэкранно.
8. Программой запрашивается ввод числа, значение которого затем сравнивается с допустимым значением. В результате этого сравнения на экран выдаются соответствующие сообщения.
9. Программой запрашивается год, определяется, високосный ли он. Результат выдается на экран.
10. Вводятся целочисленные значения двух переменных. Вводится диапазон данных. Пока значения переменных находятся в указанном диапазоне, их значения инкрементируются.
11. В качестве аргумента командной строки указывается пароль. Если пароль введен верно, постранично отображается в длинном формате с указанием скрытых файлов содержимое каталога /etc.
12. Проверить, существует ли файл. Если да, выводится на экран его содержимое, если нет - выдается соответствующее сообщение.
13. Если файл есть каталог и этот каталог можно читать, просматривается содержимое этого каталога. Если каталог отсутствует, он создается. Если файл не есть каталог, просматривается содержимое файла.
14. Анализируются атрибуты файла. Если первый файл существует и используется для чтения, а второй файл существует и используется для записи, то содержимое первого файла перенаправляется во второй файл. В случае несовпадений указанных атрибутов или отсутствия файлов на экран выдаются соответствующие сообщения (использовать имена файлов и/или позиционные параметры).
15. Если файл запуска программы найден, программа запускается (по выбору).
16. В качестве позиционного параметра задается файл, анализируется его размер. Если размер файла больше нуля, содержимое файла сортируется по

первому столбцу по возрастанию, отсортированная информация помещается в другой файл, содержимое которого затем отображается на экране.

Для сравнения с другими языками программирования сделайте аналогичные действия на Java, Си и Python.

## Ход работы:

По заданию пишем код на Shell:

### Задание 1

```
daniil@ubuntuserver:~$ nano file1_
```

```
GNU nano 6.2
echo "Morozov"
printf "Daniil\n"
_
```

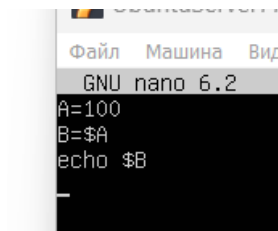
```
daniil@ubuntuserver:~$ sh file1
Morozov
Daniil
daniil@ubuntuserver:~$
```

### Задание 2

```
GNU nano 6.2
A=100
echo $A_
```

```
daniil@ubuntuserver:~$ sh file2
100
daniil@ubuntuserver:~$
```

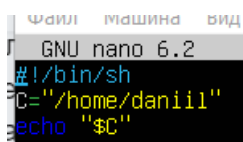
### Задание 3



```
GNU nano 6.2
A=100
B=$A
echo $B
_
```

```
daniil@ubuntuserver:~$ sh file2
100
daniil@ubuntuserver:~$
```

### Задание 4



```
GNU nano 6.2
#!/bin/sh
C="/home/daniil"
echo "$C"
```

```
daniil@ubuntuserver:~$ cd ../../..  
daniil@ubuntuserver:/$ pwd  
/  
daniil@ubuntuserver:/$ cd $(./home/daniil/file3.sh)  
daniil@ubuntuserver:~$ pwd  
/home/daniil  
daniil@ubuntuserver:~$ _
```

## Задание 5

```
daniil@ubuntuserver:~$ D="pwd"  
daniil@ubuntuserver:~$ $D  
/home/daniil  
daniil@ubuntuserver:~$
```

## Задание 6

```
daniil@ubuntuserver:~$ E="cat file2 ; echo $D"  
daniil@ubuntuserver:~$ $E  
100  
100  
daniil@ubuntuserver:~$ eval "$E"  
A=100  
B=$A  
echo $B  
pwd
```

## Задание 7

```
daniil@ubuntuserver:~$ F="sort"_  
daniil@ubuntuserver:~$ cat file7.txt  
Morozov  
Krasikov  
Berezin  
daniil@ubuntuserver:~$ $F file7.txt  
Berezin  
Krasikov  
Morozov  
daniil@ubuntuserver:~$ _
```

## Часть 2

### Задание 1

Код:

```
echo "Vvedite chislo"  
read a  
echo "$a"
```

Результат:

```
daniil@ubuntuserver:~$ sh file11
Vvedite chislo
45
45
daniil@ubuntuserver:~$
```

## Задание 2

Код:

```
GNU nano 6.2
echo "Vvedite name"
read a
echo "Hello $a"
```

Результат:

```
daniil@ubuntuserver:~$ sh file11
Vvedite name
Daniil
Hello Daniil
daniil@ubuntuserver:~$
```

## Задание 3

Код:

```
GNU nano 6.2
echo "Vvedite #1 chislo"
read a
echo "Vvedite #2 chislo"
read b
echo "sum $(expr $a + $b)"
echo "difference $(expr $a - $b)"
echo "product $(expr $a \* $b)"
echo "scale=2; $a/$b" | bc
```

Результат:

```
daniil@ubuntuserver:~$ sh file11
Vvedite #1 chislo
27
Vvedite #2 chislo
15
sum 42
difference 12
product 405
1.80
daniil@ubuntuserver:~$
```

## Задание 4

Код:

```
GNU nano 6.2
echo "vvedite radius"
read a
echo "vvedite height"
read b
pi=3.14159
echo "Obem"
echo "scale=2; $pi*$a*$b" | bc
```

Результат:

```
daniil@ubuntuserver:~$ sh file11
vvedite radius
9
vvedite height
12
Obem
339.29172
daniil@ubuntuserver:~$
```

Задание 5

Код:

```
GNU nano 6.2
#!/bin/bash
args=("$@")
echo "Имя программы $0 Количество аргументов $#"
```

```
for ((i=0; i<"$#"; i++)); do
echo "Аргумент $((i+1)): ${args[i]}"
done
```

Результат:

```
daniil@ubuntuserver:~$ ./file11.sh 2 3 4
Имя программы ./file11.sh Количество аргументов 3
Аргумент 1: 2
Аргумент 2: 3
Аргумент 3: 4
daniil@ubuntuserver:~$
```

Задание 6

Код:



```
#!/bin/bash

# Проверяем, был ли передан аргумент командной строки (путь к файлу)
if [ $# -eq 0 ]; then
    echo "Ошибка: Укажите имя файла в аргументах командной строки."
    exit 1
fi

file="$1"

# Проверяем, существует ли файл
if [ ! -f "$file" ]; then
    echo "Ошибка: Файл не существует."
    exit 1
fi

cat "$file"
```

```
read -p "Нажмите Enter для продолжения..."

clear
```

Результат:

```
daniil@ubuntuuserver:~$ ./file12.sh file7.txt
Morozov
Krasikov
Berezin
Нажмите Enter для продолжения...
```

```
daniil@ubuntuuserver:~$
```

Задание 7

Код:

```

GNU nano 6.2                                file13.sh
#!/bin/bash

# Используем цикл for для перебора всех файлов с расширением .txt в текущем каталоге
for file in *.txt; do
    # Проверяем, существует ли файл
    if [ -f "$file" ]; then
        # Отображаем имя файла
        echo "Содержимое файла: $file"

        # Отображаем содержимое файла постранично с помощью команды less
        less "$file"

        read -p "Нажмите Enter для продолжения..."

        clear
    fi
done

```

Результат:

```

daniil@ubuntuuserver:~$ ./file13.sh
Содержимое файла: file123.txt
Нажмите Enter для продолжения...

```

```

File
123 dsgagagd
file123.txt (END)

```

```

file 1256 dgszgdgsgsdg
file1256.txt (END)

```

```

Содержимое файла: file1256.txt
Нажмите Enter для продолжения...

```

```

Morozov
Krasikov
Berezin
file7.txt (END)

```

```

Содержимое файла: file7.txt
Нажмите Enter для продолжения...

```

```

daniil@ubuntuuserver:~$

```

Задание 8

Код:

```
GNU nano 6.2 file14.sh
#!/bin/bash

# Запрашиваем ввод числа у пользователя
read -p "Введите число: " number

# Допустимое значение
valid_number=10

# Сравниваем введенное число с допустимым значением
if [ "$number" -eq "$valid_number" ]; then
    echo "Вы ввели допустимое число: $valid_number"
elif [ "$number" -lt "$valid_number" ]; then
    echo "Введенное число меньше допустимого значения: $valid_number"
else
    echo "Введенное число больше допустимого значения: $valid_number"
fi
```

Результат:

```
daniil@ubuntuserver:~$ chmod +x file14.sh
daniil@ubuntuserver:~$ ./file14.sh
Введите число: 78
Введенное число больше допустимого значения: 10
daniil@ubuntuserver:~$
```

Задание 9

Код:

```
GNU nano 6.2 file15.sh
#!/bin/bash

# Запрашиваем ввод года у пользователя
read -p "Введите год: " year

# Проверяем, является ли год високосным
if [ $((year % 4)) -eq 0 ] && [ $((year % 100)) -ne 0 ] || [ $((year % 400)) -eq 0 ]; then
    echo "Год $year - високосный."
else
    echo "Год $year - не високосный."
fi
```

Результат:

```
daniil@ubuntuserver:~$ chmod +x file15.sh
daniil@ubuntuserver:~$ ./file15.sh
Введите год: 2017
Год 2017 – не високосный.
daniil@ubuntuserver:~$
```

[  $((year \% 4)) -eq 0$  ]: Это первая часть условия. Она проверяет, что год делится на 4 без остатка. Если условие выполняется, то это означает, что год является кратным 4.

[  $((year \% 100)) -ne 0$  ]: Вторая часть условия проверяет, что год не делится на 100 без остатка. Это исключает некоторые високосные годы, такие как 1900.

[  $((year \% 400)) -eq 0$  ]: Третья часть условия проверяет, что год делится на 400 без остатка. Это исключает исключения и снова включает високосные годы, такие как 2000.

## Zadanie 10

Код:

```
GNU nano 6.2                                file16.sh
#!/bin/bash

# Запрашиваем ввод двух целых чисел
read -p "Введите первое целое число: " num1
read -p "Введите второе целое число: " num2

# Запрашиваем диапазон данных
read -p "Введите начало диапазона: " start
read -p "Введите конец диапазона: " end

# Проверяем, что начало диапазона меньше или равно концу диапазона
if [ "$start" -le "$end" ]; then
    # Входим в цикл, пока значения переменных находятся в диапазоне
    while [ "$num1" -ge "$start" ] && [ "$num1" -le "$end" ] && [ "$num2" -ge "$start" ] && [ "$num2" -le "$end" ]; do
        echo "Значение первой переменной: $num1, Значение второй переменной: $num2"
        num1=$((num1 + 1))
        num2=$((num2 + 1))
    done
else
    echo "Ошибка: Начало диапазона больше конца диапазона."
fi
```

[ "\$num1" -ge "\$start" ]: Это первая часть условия. Она проверяет, что значение переменной num1 больше или равно значению start. Пока это условие выполняется, цикл будет продолжаться.

[ "\$num1" -le "\$end" ]: Вторая часть условия проверяет, что значение переменной num1 меньше или равно значению end. Таким образом, это условие обеспечивает, что num1 находится в заданном диапазоне [start, end]

[ "\$num2" -ge "\$start" ]: Третья часть условия проверяет, что значение переменной num2 больше или равно значению start. Это аналогично первой части, но для переменной num2.

[ "\$num2" -le "\$end" ]: Четвертая часть условия проверяет, что значение переменной num2 меньше или равно значению end. Это аналогично второй части, но для переменной num2.

Результат:

```
daniil@ubuntuserver:~$ nano file16.sh
daniil@ubuntuserver:~$ chmod +x file16.sh
daniil@ubuntuserver:~$ ./file16.sh
Введите первое целое число: 7
Введите второе целое число: 17
Введите начало диапазона: 1
Введите конец диапазона: 20
Значение первой переменной: 7, Значение второй переменной: 17
Значение первой переменной: 8, Значение второй переменной: 18
Значение первой переменной: 9, Значение второй переменной: 19
Значение первой переменной: 10, Значение второй переменной: 20
```

Задание 11

Код:

```
GNU nano 6.2 file17.sh
#!/bin/bash

# Проверяем, был ли передан пароль в качестве аргумента командной строки
if [ "$#" -ne 1 ]; then
    echo "Использование: $0 <пароль>"
    exit 1
fi

# Указываем верный пароль
correct_password="0000"

# Сравниваем введенный пароль с верным
if [ "$1" = "$correct_password" ]; then
    # Если пароль верен, отображаем содержимое каталога /etc в длинном формате
    ls -la /etc
else
    # Если пароль неверен, выводим сообщение об ошибке
    echo "Неверный пароль."
fi
```

Результат:

```

daniil@ubuntuserver:~$ ./file17.sh 7777
Неверный пароль.
daniil@ubuntuserver:~$ ./file17.sh 0000
total 840
drwxr-xr-x 96 root root      4096 ноя  6 15:34 .
drwxr-xr-x 19 root root      4096 окт 24 06:29 ..
-rw-r--r--  1 root root      3028 авг 10 00:17 adduser.conf
drwxr-xr-x  2 root root      4096 авг 10 00:22 alternatives
drwxr-xr-x  3 root root      4096 авг 10 00:21 apparmor
drwxr-xr-x  8 root root      4096 авг 10 00:21 apparmor.d
drwxr-xr-x  3 root root      4096 авг 10 00:21 apport
drwxr-xr-x  8 root root      4096 окт 24 06:20 apt

```

## Задание 12

Код:

```

GNU nano 6.2 file18.sh
#!/bin/bash

# Задаем путь к каталогу, в котором находятся файлы
directory_path="/home/daniil"

# Переходим в указанный каталог
cd "$directory_path" || exit

# Перебираем все файлы в каталоге
for file in *; do
    if [ -f "$file" ]; then
        echo "Содержимое файла: $file"
        cat "$file"
        echo "-----"
    fi
done

```

Результат:

```

daniil@ubuntuuserver:~$ ./file18.sh
Введите имя файла: file17.sh
#!/bin/bash

# Проверяем, был ли передан пароль в качестве аргумента командной строки
if [ "$#" -ne 1 ]; then
    echo "Использование: $0 <пароль>"
    exit 1
fi

# Указываем верный пароль
correct_password="0000"

# Сравниваем введенный пароль с верным
if [ "$1" = "$correct_password" ]; then
    # Если пароль верен, отображаем содержимое каталога /etc в длинном формате
    ls -la /etc
else
    # Если пароль неверен, выводим сообщение об ошибке
    echo "Неверный пароль."
fi
daniil@ubuntuuserver:~$

```

### Задание 13

Код:

```

#!/bin/bash

# Запрашиваем у пользователя ввести путь к файлу или каталогу
read -p "Введите путь к файлу или каталогу: " path

# Проверяем, существует ли указанный путь
if [ -e "$path" ]; then
    # Проверяем, является ли указанный путь каталогом
    if [ -d "$path" ]; then
        # Если это каталог и его можно читать, просматриваем его содержимое
        if [ -r "$path" ]; then
            echo "Содержимое каталога '$path':"
            ls "$path"
        else
            echo "Нет прав на чтение каталога: '$path'"
        fi
    else
        # Если это не каталог, просматриваем содержимое файла
        echo "Содержимое файла '$path':"
        cat "$path"
    fi
else
    # Если указанный путь не существует, создаем каталог
    echo "Каталог не существует. Создаем каталог: '$path'"
    mkdir -p "$path"
fi

```

Результат:

```
daniil@ubuntuserver:~$ chmod +x file19.sh
daniil@ubuntuserver:~$ ./file19.sh
Введите путь к файлу или каталогу: file18.sh
Содержимое файла 'file18.sh':
#!/bin/bash

# Запрашиваем у пользователя ввести имя файла
read -p "Введите имя файла: " file_name

# Проверяем, существует ли введенный файл
if [ -e "$file_name" ]; then
    # Если файл существует, выводим его содержимое
    cat "$file_name"
else
    # Если файла нет, выводим сообщение об ошибке
    echo "Файл не существует: $file_name"
fi
```

```
daniil@ubuntuserver:~$ ./file19.sh
Введите путь к файлу или каталогу: proba
Каталог не существует. Создаем каталог: 'proba'
daniil@ubuntuserver:~$ ls -la
total 112
drwxrwxr-x 3 daniil daniil 4096 ноя  6 17:31 .local
drwxrwxr-x 2 daniil daniil 4096 ноя  7 04:04 proba
```

Задание 14

Код:



```

GNU nano 6.2 file20.sh
#!/bin/bash

# Запрашиваем у пользователя ввести имена файлов
read -p "Введите имя файла для чтения: " read_file
read -p "Введите имя файла для записи: " write_file

# Проверяем, существуют ли указанные файлы
if [ ! -e "$read_file" ]; then
    echo "Ошибка: Файл для чтения не существует: $read_file"
    exit 1
fi

if [ ! -e "$write_file" ]; then
    echo "Ошибка: Файл для записи не существует: $write_file"
    exit 1
fi

# Проверяем, можно ли читать из файла для чтения
if [ ! -r "$read_file" ]; then
    echo "Ошибка: Нет прав на чтение из файла: $read_file"
    exit 1
fi

# Проверяем, можно ли писать в файл для записи
if [ ! -w "$write_file" ]; then
    echo "Ошибка: Нет прав на запись в файл: $write_file"
    exit 1
fi

# Если атрибуты файлов верны, перенаправляем содержимое
cat "$read_file" > "$write_file"

echo "Содержимое файла '$read_file' перенаправлено в файл '$write_file'."

```

Результат:

```

daniil@ubuntuserver:~$ chmod +x file20.sh
daniil@ubuntuserver:~$ ./file20.sh
Введите имя файла для чтения: file20.sh
Введите имя файла для записи: file20_2.sh
Содержимое файла 'file20.sh' перенаправлено в файл 'file20_2.sh'.
daniil@ubuntuserver:~$ nano file20_2.sh

```

```
GNU nano 6.2                                file20_2.sh
#!/bin/bash

# Запрашиваем у пользователя ввести имена файлов
read -p "Введите имя файла для чтения: " read_file
read -p "Введите имя файла для записи: " write_file

# Проверяем, существуют ли указанные файлы
if [ ! -e "$read_file" ]; then
    echo "Ошибка: Файл для чтения не существует: $read_file"
    exit 1
fi

if [ ! -e "$write_file" ]; then
    echo "Ошибка: Файл для записи не существует: $write_file"
    exit 1
fi

# Проверяем, можно ли читать из файла для чтения
if [ ! -r "$read_file" ]; then
    echo "Ошибка: Нет прав на чтение из файла: $read_file"
    exit 1
fi

# Проверяем, можно ли писать в файл для записи
if [ ! -w "$write_file" ]; then
    echo "Ошибка: Нет прав на запись в файл: $write_file"
    [ Read 33 lines ]
```

Задание 15

Код:

```

GNU nano 6.2                                     file21.sh *
#!/bin/bash

# Запрашиваем у пользователя ввести путь к файлу программы
read -p "Введите путь к файлу программы: " program_file

# Проверяем, существует ли указанный файл
if [ ! -e "$program_file" ]; then
    echo "Ошибка: Файл программы не существует: $program_file"
    exit 1
fi

# Проверяем, можно ли запустить файл программы
if [ ! -x "$program_file" ]; then
    echo "Ошибка: Нет прав на выполнение файла программы: $program_file"
    exit 1
fi

# Предлагаем пользователю выбор: запустить программу или нет
read -p "Хотите запустить программу? (y/n): " choice

# Проверяем выбор пользователя
if [ "$choice" == "y" ]; then
    # Если пользователь выбрал "y", запускаем программу
    "$program_file"
    echo "Программа выполнена."
else
    # Если пользователь выбрал что-то другое, выводим сообщение
    echo "Программа не выполнена."
fi

```

Результат:

```

daniil@ubuntuerver:~$ ./file21.sh
Введите путь к файлу программы: file20.sh
Хотите запустить программу? (y/n): y
Введите имя файла для чтения: file20_2.sh
Введите имя файла для записи: file20_3.sh
Ошибка: Файл для записи не существует: file20_3.sh
Программа выполнена.

```

Задание 16

Код:

```

GNU nano 6.2                                     file22.sh
#!/bin/bash

# Запрашиваем у пользователя ввести имя файла
read -p "Введите имя файла: " input_file

# Проверяем, существует ли указанный файл
if [ ! -e "$input_file" ]; then
    echo "Ошибка: Файл не существует: $input_file"
    exit 1
fi

# Получаем размер файла
file_size=$(wc -c < "$input_file")

# Проверяем, что размер файла больше нуля
if [ "$file_size" -gt 0 ]; then
    # Создаем временный файл для хранения отсортированной информации
    sorted_file=$(mktemp)

    # Сортируем содержимое файла по первому столбцу по возрастанию и сохраняем во временный файл
    sort -k1 "$input_file" > "$sorted_file"

    # Выводим отсортированное содержимое на экран
    cat "$sorted_file"

    # Удаляем временный файл после использования

```

```

    rm "$sorted_file"
else
    echo "Файл пустой. Нет данных для сортировки."
fi

```

```

GNU nano 6.2                                     file22_2.sh
Морозов Даниил Сергеевич
Денисова Елена
Березин Александр
Алексндрова Яна

```

Результат:

```

daniil@ubuntuserver:~$ ./file22.sh
Введите имя файла: file22_2.sh
Алексндрова Яна
Березин Александр
Денисова Елена
Морозов Даниил Сергеевич
daniil@ubuntuserver:~$

```

Те же задания на C++

Задание 1

Код:

```
GNU nano 6.2 file7.cpp
#include <iostream>

using namespace std;

int main() {
    // Объявление переменной
    int userValue;

    // Запрашиваем у пользователя значение переменной
    cout << "Введите значение переменной: ";
    cin >> userValue;

    // Выводим значение переменной на экран
    cout << "Введенное значение: " << userValue << endl;

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ nano file7.cpp
daniil@ubuntuserver:~$ g++ -o file7 file7.cpp
daniil@ubuntuserver:~$ ./file7
Введите значение переменной: 789123
Введенное значение: 789123
daniil@ubuntuserver:~$
```

Задание 2

Код:

```
GNU nano 6.2 file8.cpp
#include <iostream>
#include <string>

using namespace std;

int main() {
    string username;

    // Запрашиваем у пользователя имя
    cout << "Введите ваше имя: ";
    getline(cin, username);

    // Приветствуем пользователя
    cout << "Привет, " << username << "! Добро пожаловать!" << endl;

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ nano file8.cpp
daniil@ubuntuserver:~$ g++ -o file8 file8.cpp
daniil@ubuntuserver:~$ ./file8
Введите ваше имя: Daniil Morozov PI-21-1
Привет, Daniil Morozov PI-21-1! Добро пожаловать!
daniil@ubuntuserver:~$
```

Задание 3

Код:

```
GNU nano 6.2 file9.cpp
#include <iostream>

using namespace std;

int main() {
    double firstNumber, secondNumber;

    // Запрашиваем у пользователя значения переменных
    cout << "Введите значение первой переменной: ";
    cin >> firstNumber;

    cout << "Введите значение второй переменной: ";
    cin >> secondNumber;

    // Вычисляем и выводим сумму, разность, произведение и частное
    cout << "Сумма: " << firstNumber + secondNumber << endl;
    cout << "Разность: " << firstNumber - secondNumber << endl;
    cout << "Произведение: " << firstNumber * secondNumber << endl;

    // Проверяем деление на ноль
    if (secondNumber != 0) {
        cout << "Частное: " << firstNumber / secondNumber << endl;
    } else {
        cout << "Ошибка: деление на ноль!" << endl;
    }
}
```

```
    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ nano file9.cpp
daniil@ubuntuserver:~$ g++ -o file9 file9.cpp
daniil@ubuntuserver:~$ ./file9
Введите значение первой переменной: 45
Введите значение второй переменной: 78
Сумма: 123
Разность: -33
Произведение: 3510
Частное: 0.576923
daniil@ubuntuserver:~$
```

Задание 4

Код:

```
GNU nano 6.2 file10.cpp
#include <iostream>
#include <cmath>

using namespace std;

int main() {
    const double pi = 3.14159;

    double radius, height;

    // Запрашиваем радиус у пользователя
    cout << "Введите радиус цилиндра: ";
    cin >> radius;

    // Запрашиваем высоту у пользователя
    cout << "Введите высоту цилиндра: ";
    cin >> height;

    // Вычисляем объем цилиндра
    double volume = pi * pow(radius, 2) * height;

    // Выводим результат
    cout << "Объем цилиндра: " << volume << endl;

    return 0;
}
```

Результат:

```
daniil@ubuntuuserver:~$ nano file10.cpp
daniil@ubuntuuserver:~$ g++ -o file10 file10.cpp
daniil@ubuntuuserver:~$ ./file10
Введите радиус цилиндра: 14
Введите высоту цилиндра: 89
Объем цилиндра: 54801.9
daniil@ubuntuuserver:~$
```

Задание 5

Код:



```
GNU nano 6.2 file11.cpp
#include <iostream>

using namespace std;

int main(int argc, char* argv[]) {
    // Отображаем имя программы (первый аргумент)
    cout << "Имя программы: " << argv[0] << endl;

    // Отображаем количество аргументов командной строки
    cout << "Количество аргументов командной строки: " << argc - 1 << endl;

    // Отображаем значения каждого аргумента командной строки
    for (int i = 1; i < argc; ++i) {
        cout << "Аргумент " << i << ": " << argv[i] << endl;
    }

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ nano file11.cpp
daniil@ubuntuserver:~$ g++ -o file11 file11.cpp
daniil@ubuntuserver:~$ ./file11
Имя программы: ./file11
Количество аргументов командной строки: 0
daniil@ubuntuserver:~$ ./file11 file12.sh
Имя программы: ./file11
Количество аргументов командной строки: 1
Аргумент 1: file12.sh
daniil@ubuntuserver:~$
```

Задание 6

Код:

```

GNU nano 6.2                                     file12.cpp
#include <iostream>
#include <fstream>
#include <cstdlib> // для функции system()

using namespace std;

void displayFileContent(const char* filename) {
    ifstream file(filename);

    if (file.is_open()) {
        cout << "Содержимое файла " << filename << ":" << endl;

        char ch;
        while (file.get(ch)) {
            cout << ch;
        }

        file.close();
    } else {
        cerr << "Ошибка: Не удалось открыть файл " << filename << endl;
    }
}

int main(int argc, char* argv[]) {
    // Проверяем, передано ли достаточное количество аргументов
    if (argc != 2) {

```

[ Wrote 44 lines ]

```

        cerr << "Использование: " << argv[0] << " <имя_файла>" << endl;
        return 1;
    }

    const char* filename = argv[1];

    displayFileContent(filename);

    // Ждем ввода пользователя перед очисткой экрана
    cout << "Нажмите Enter для очистки экрана...";
    cin.ignore(); // Игнорируем введенные символы
    cin.get();    // Ожидаем ввод Enter

    // Очищаем экран
    system("clear");

    return 0;
}

```

Результат:

```
daniil@ubuntu-server:~$ ./file12 file12.sh
Содержимое файла file12.sh:
#!/bin/bash

# Проверяем, был ли передан аргумент командной строки (путь к файлу)
if [ $# -eq 0 ]; then
    echo "Ошибка: Укажите имя файла в аргументах командной строки."
    exit 1
fi

file="$1"

# Проверяем, существует ли файл
if [ ! -f "$file" ]; then
    echo "Ошибка: Файл не существует."
    exit 1
fi

cat "$file"

read -p "Нажмите Enter для продолжения..."

clear
Нажмите Enter для очистки экрана...
```

## Задание 7

Код:

```
GNU nano 6.2 file13.cpp
#include <iostream>
#include <fstream>
#include <string>
#include <dirent.h>

using namespace std;

void displayFileContent(const char* filename) {
    ifstream file(filename);

    if (file.is_open()) {
        cout << "Содержимое файла " << filename << ":" << endl;

        string line;
        while (getline(file, line)) {
            cout << line << endl;
        }

        cout << endl;

        file.close();
    } else {
        cerr << "Ошибка: Не удалось открыть файл " << filename << endl;
    }
}
```

```
void displayDirectoryContents() {
    DIR* dir;
    struct dirent* entry;

    if ((dir = opendir(".")) != nullptr) {
        while ((entry = readdir(dir)) != nullptr) {
            // Проверяем, что это файл, а не каталог или другой тип
            if (entry->d_type == DT_REG) {
                displayFileContent(entry->d_name);
            }
        }

        closedir(dir);
    } else {
        cerr << "Ошибка: Не удалось открыть текущий каталог." << endl;
    }
}

int main() {
    displayDirectoryContents();

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ nano file13.cpp
daniil@ubuntuserver:~$ g++ -o file13 file13.cpp
daniil@ubuntuserver:~$ ./file13
```

```
Содержимое файла file16.sh:
#!/bin/bash

# Запрашиваем ввод двух целых чисел
read -p "Введите первое целое число: " num1
read -p "Введите второе целое число: " num2

# Запрашиваем диапазон данных
read -p "Введите начало диапазона: " start
read -p "Введите конец диапазона: " end

# Проверяем, что начало диапазона меньше или равно концу диапазона
if [ "$start" -le "$end" ]; then
    # Входим в цикл, пока значения переменных находятся в диапазоне
    while [ "$num1" -ge "$start" ] && [ "$num1" -le "$end" ] && [ "$num2" -ge "$start" ] && [ "$num2" -le "$end" ]; do
        echo "Значение первой переменной: $num1, Значение второй переменной: $num2"
        num1=$((num1 + 1))
        num2=$((num2 + 1))
    done
else
    echo "Ошибка: Начало диапазона больше конца диапазона."
fi

Содержимое файла file12.sh:
#!/bin/bash

# Проверяем, был ли передан аргумент командной строки (путь к файлу)
if [ $# -eq 0 ]; then
    echo "Ошибка: Укажите имя файла в аргументах командной строки."
```

## Задание 8

Код:

```
GNU nano 6.2 file14.cpp
#include <iostream>

using namespace std;

int main() {
    const int allowedValue = 42; // Допустимое значение

    int userInput;

    // Запрашиваем ввод числа от пользователя
    cout << "Введите число: ";
    cin >> userInput;

    // Сравниваем введенное число с допустимым значением
    if (userInput == allowedValue) {
        cout << "Введенное число совпадает с допустимым значением." << endl;
    } else if (userInput < allowedValue) {
        cout << "Введенное число меньше допустимого значения." << endl;
    } else {
        cout << "Введенное число больше допустимого значения." << endl;
    }

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ nano file14.cpp
daniil@ubuntuserver:~$ g++ -o file14 file14.cpp
daniil@ubuntuserver:~$ ./file14
Введите число: 75
Введенное число больше допустимого значения.
daniil@ubuntuserver:~$
```

## Задание 9

Код:

```
GNU nano 6.2 file15.cpp
#include <iostream>

using namespace std;

bool isLeapYear(int year) {
    // Год високосный, если он делится на 4, но не делится на 100,
    // или если он делится на 400.
    return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
}

int main() {
    int year;

    // Запрашиваем год от пользователя
    cout << "Введите год: ";
    cin >> year;

    // Проверяем, является ли год високосным
    if (isLeapYear(year)) {
        cout << year << " - високосный год." << endl;
    } else {
        cout << year << " - не високосный год." << endl;
    }

    return 0;
}
```

[ Wrote 26 lines ]

Результат:

```
daniil@ubuntuserver:~$ nano file15.cpp
daniil@ubuntuserver:~$ g++ -o file15 file15.cpp
daniil@ubuntuserver:~$ ./file15
Введите год: 2009
2009 - не високосный год.
daniil@ubuntuserver:~$
```

## Задание 10

Код:

```
GNU nano 6.2                                file16.cpp *
#include <iostream>

using namespace std;

int main() {
    // Вводим целочисленные значения двух переменных
    int var1, var2;
    cout << "Введите значение первой переменной: ";
    cin >> var1;
    cout << "Введите значение второй переменной: ";
    cin >> var2;

    // Вводим диапазон данных
    int minRange, maxRange;
    cout << "Введите минимальное значение диапазона: ";
    cin >> minRange;
    cout << "Введите максимальное значение диапазона: ";
    cin >> maxRange;

    // Пока значения переменных находятся в указанном диапазоне, инкрементируем их
    while (var1 >= minRange && var1 <= maxRange && var2 >= minRange && var2 <= maxRange) {
        cout << "Значение первой переменной: " << var1 << ", Значение второй переменной: " << var2 << endl;

        // Инкрементируем значения переменных
        var1++;
        var2++;
    }

    cout << "Конец программы." << endl;

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ nano file16.cpp
daniil@ubuntuserver:~$ g++ -o file16 file16.cpp
daniil@ubuntuserver:~$ ./file16
Введите значение первой переменной: 7
Введите значение второй переменной: 12
Введите минимальное значение диапазона: 6
Введите максимальное значение диапазона: 17
Значение первой переменной: 7, Значение второй переменной: 12
Значение первой переменной: 8, Значение второй переменной: 13
Значение первой переменной: 9, Значение второй переменной: 14
Значение первой переменной: 10, Значение второй переменной: 15
Значение первой переменной: 11, Значение второй переменной: 16
Значение первой переменной: 12, Значение второй переменной: 17
Конец программы.
daniil@ubuntuserver:~$
```

## Задание 11

Код:

```
GNU nano 6.2                                     file17.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <cstdlib> // для функции system()
#include <unistd.h> // для функции access()
#include <dirent.h> // для opendir(), readdir(), closedir()

using namespace std;

bool checkPassword(const string& enteredPassword, const string& correctPassword) {
    return enteredPassword == correctPassword;
}

void displayDirectoryContent(const char* path) {
    DIR* dir = opendir(path);

    if (dir != nullptr) {
        struct dirent* entry;
        while ((entry = readdir(dir)) != nullptr) {
            cout << entry->d_name << endl;
        }

        closedir(dir);
    } else {
        cerr << "Ошибка: Не удалось открыть каталог " << path << endl;
    }
}
```



```

    }
}

int main(int argc, char* argv[]) {
    // Проверяем, передано ли достаточное количество аргументов
    if (argc != 2) {
        cerr << "Использование: " << argv[0] << " <пароль>" << endl;
        return 1;
    }

    const string correctPassword = "0000"; // Установите свой пароль

    const string enteredPassword(argv[1]);

    // Проверяем введенный пароль
    if (!checkPassword(enteredPassword, correctPassword)) {
        cerr << "Ошибка: Введен неверный пароль." << endl;
        return 1;
    }

    cout << "Пароль верный. Содержимое каталога /etc:" << endl;

    displayDirectoryContent("/etc");

    return 0;
}

```

Результат:

```

daniil@ubuntuserver:~$ nano file17.cpp
daniil@ubuntuserver:~$ g++ -o file17 file17.cpp
daniil@ubuntuserver:~$ ./file17 0000
Пароль верный. Содержимое каталога /etc:
profile.d
subgid
rsyslog.d
bash_completion
selinux
netplan
terminfo
cron.daily
passwd
rc0.d
python3
mke2fs.conf

```

Задание 12

Код:

```
GNU nano 6.2                                     file18.cpp
#include <iostream>
#include <fstream>

using namespace std;

int main(int argc, char* argv[]) {
    // Проверяем, передано ли достаточное количество аргументов
    if (argc != 2) {
        cerr << "Использование: " << argv[0] << " <имя_файла>" << endl;
        return 1;
    }

    const char* filename = argv[1];

    // Создаем объект ifstream для чтения файла
    ifstream file(filename);

    // Проверяем, существует ли файл
    if (file.is_open()) {
        cout << "Содержимое файла " << filename << ":" << endl;

        // Читаем и выводим содержимое файла
        string line;
        while (getline(file, line)) {
            cout << line << endl;
        }

        // Закрываем файл
        file.close();
    } else {
        cerr << "Ошибка: Файл " << filename << " не существует." << endl;
        return 1;
    }

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ g++ -o file18 file18.cpp
daniil@ubuntuserver:~$ ./file18 file19.sh
Содержимое файла file19.sh:
#!/bin/bash

# Запрашиваем у пользователя ввести путь к файлу или каталогу
read -p "Введите путь к файлу или каталогу: " path

# Проверяем, существует ли указанный путь
if [ -e "$path" ]; then
    # Проверяем, является ли указанный путь каталогом
    if [ -d "$path" ]; then
        # Если это каталог и его можно читать, просматриваем его содержимое
        if [ -r "$path" ]; then
            echo "Содержимое каталога '$path':"
            ls "$path"
        else
            echo "Нет прав на чтение каталога: '$path'"
        fi
    else
        # Если это не каталог, просматриваем содержимое файла
        echo "Содержимое файла '$path':"
        cat "$path"
    fi
else
    # Если указанный путь не существует, создаем каталог
    echo "Каталог не существует. Создаем каталог: '$path'"
fi
```

### Задание 13

Код:

```
#include <iostream>
#include <fstream>
#include <dirent.h> // для opendir(), readdir(), closedir()
#include <sys/stat.h> // для stat()
#include <sys/types.h>
#include <cstring>

using namespace std;

void listDirectory(const char* path) {
    DIR* dir = opendir(path);

    // Если открытие каталога не удалось
    if (dir == nullptr) {
        // Пытаемся создать каталог
        if (mkdir(path, S_IRWXU | S_IRWXG | S_IROTH | S_IXOTH) == 0) {
            cout << "Каталог создан: " << path << endl;
        } else {
            cerr << "Ошибка: Не удалось открыть или создать каталог: " << path << endl;
            return;
        }
    } else {
        cout << "Содержимое каталога " << path << ":" << endl;

        struct dirent* entry;
        while ((entry = readdir(dir)) != nullptr) {
```

[ Read 75 lines ]

```
        cout << entry->d_name << endl;
    }

    closedir(dir);
}

void readFile(const char* filename) {
    ifstream file(filename);

    // Если открытие файла удалось
    if (file.is_open()) {
        cout << "Содержимое файла " << filename << ":" << endl;

        string line;
        while (getline(file, line)) {
            cout << line << endl;
        }

        file.close();
    } else {
        cerr << "Ошибка: Не удалось открыть файл: " << filename << endl;
    }
}

int main(int argc, char* argv[]) {
```

```

// Проверяем, передано ли достаточное количество аргументов
if (argc != 2) {
    cerr << "Использование: " << argv[0] << " <путь_к_каталогу_или_файлу>" << endl;
    return 1;
}

const char* path = argv[1];

struct stat fileInfo;
if (stat(path, &fileInfo) != 0) {
    cerr << "Ошибка: Не удалось получить информацию о файле или каталоге: " << path << endl;
    return 1;
}

// Если это каталог
if (S_ISDIR(fileInfo.st_mode)) {
    listDirectory(path);
} else {
    readFile(path);
}

return 0;
}

```

Результат:

```

daniil@ubuntuserver:~$ nano file19.cpp
daniil@ubuntuserver:~$ g++ -o file19 file19.cpp

```

```

daniil@ubuntuserver:~$ ./file19 /home/daniil
Содержимое каталога /home/daniil:
.lessshst
file16.sh
file12.sh
file5
file11.sh
.local
file20.cpp
..
file21.cpp
file22_2.sh
file123.txt
.sudo_as_admin_successful
file19
.
file20_2.sh
file20
file14.sh
file7.txt
file1256.txt
file21
file2
file11

```

Задание 14

Код:

```
GNU nano 6.2                                file20.cpp
#include <iostream>
#include <cstdlib> // для функции system()
#include <fcntl.h> // для системных вызовов open(), read(), write()
#include <unistd.h>

using namespace std;

int main(int argc, char* argv[]) {
    // Проверяем, передано ли достаточное количество аргументов
    if (argc != 3) {
        cerr << "Использование: " << argv[0] << " <входной_файл> <выходной_файл>" << endl;
        return 1;
    }

    // Открываем первый файл для чтения
    int input_fd = open(argv[1], O_RDONLY);

    // Проверяем успешность открытия файла
    if (input_fd == -1) {
        perror("Ошибка открытия входного файла");
        return 1;
    }

    // Открываем второй файл для записи
    int output_fd = open(argv[2], O_WRONLY | O_CREAT | O_TRUNC, S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH);

    // Проверяем успешность открытия файла
    if (output_fd == -1) {
        perror("Ошибка открытия выходного файла");
        // Закрываем открытый файл
        close(input_fd);
        return 1;
    }

    // Читаем из первого файла и записываем во второй файл
    char buffer[4096]; // Буфер для чтения и записи данных
    ssize_t bytesRead;

    while ((bytesRead = read(input_fd, buffer, sizeof(buffer))) > 0) {
        if (write(output_fd, buffer, bytesRead) != bytesRead) {
            perror("Ошибка записи в выходной файл");
            // Закрываем открытые файлы
            close(input_fd);
            close(output_fd);
            return 1;
        }
    }

    // Проверяем, были ли ошибки при чтении из файла
    if (bytesRead == -1) {
```

```

        perror("Ошибка чтения из входного файла");
        // Закрываем открытые файлы
        close(input_fd);
        close(output_fd);
        return 1;
    }

    // Закрываем открытые файлы
    close(input_fd);
    close(output_fd);

    cout << "Содержимое из входного файла успешно записано в выходной файл." << endl;

    return 0;
}

```

Результат:

```
daniil@ubuntuserver:~$ g++ -o file20 file20.cpp
```

```
daniil@ubuntuserver:~$ nano file20_3.sh
daniil@ubuntuserver:~$ ./file20 file20.sh file20_3.sh
```

```

GNU nano 6.2                                file20_3.sh
#!/bin/bash

# Запрашиваем у пользователя ввести имена файлов
read -p "Введите имя файла для чтения: " read_file
read -p "Введите имя файла для записи: " write_file

# Проверяем, существуют ли указанные файлы
if [ ! -e "$read_file" ]; then
    echo "Ошибка: Файл для чтения не существует: $read_file"
    exit 1
fi

if [ ! -e "$write_file" ]; then
    echo "Ошибка: Файл для записи не существует: $write_file"
    exit 1
fi

# Проверяем, можно ли читать из файла для чтения
if [ ! -r "$read_file" ]; then
    echo "Ошибка: Нет прав на чтение из файла: $read_file"
    exit 1
fi

# Проверяем, можно ли писать в файл для записи
if [ ! -w "$write_file" ]; then
    echo "Ошибка: Нет прав на запись в файл: $write_file"

```

[ Read 33 lines ]

Задание 15

Код:

```
GNU nano 6.2 file21.cpp
#include <iostream>
#include <fstream>
#include <cstdlib> // Для функции system()

using namespace std;

int main() {
    // Запрашиваем у пользователя ввести имя файла
    cout << "Введите имя файла: ";
    string filename;
    cin >> filename;

    // Пытаемся открыть файл
    ifstream file(filename.c_str());

    // Проверяем, успешно ли открыт файл
    if (!file.is_open()) {
        cerr << "Ошибка: Не удалось открыть файл: " << filename << endl;
        return 1;
    }

    // Закрываем файл
    file.close();
```

```
    // Выводим сообщение о том, что файл найден
    cout << "Файл найден. Хотите запустить программу? (y/n): ";
    char choice;
    cin >> choice;

    if (choice == 'y' || choice == 'Y') {
        // Запускаем программу с использованием system()
        if (system(filename.c_str()) == 0) {
            cout << "Программа успешно запущена." << endl;
        } else {
            cerr << "Ошибка при запуске программы." << endl;
            return 1;
        }
    } else {
        cout << "Программа не была запущена." << endl;
    }

    return 0;
}
```

Результат:

```
daniil@ubuntuserver:~$ g++ -o file21 file21.cpp
```



```
daniil@ubuntuserver:~$ ./file21
Введите имя файла: file20.sh
Файл найден. Хотите запустить программу? (y/n): n
Программа не была запущена.
daniil@ubuntuserver:~$
```

## Задание 16

Код:

```
GNU nano 6.2 file22.cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <algorithm>

using namespace std;

int main(int argc, char* argv[]) {
    // Проверяем, передан ли файл в качестве аргумента
    if (argc != 2) {
        cerr << "Использование: " << argv[0] << " <имя_файла>" << endl;
        return 1;
    }

    const char* inputFileNames = argv[1];
    const char* outputFileNames = "sorted_output.txt";

    // Открываем входной файл
    ifstream inputFile(inputFileNames);

    if (!inputFile.is_open()) {
        cerr << "Ошибка: Не удалось открыть файл " << inputFileNames << endl;
        return 1;
    }

    // Проверяем размер файла
    [ Read 83 lines ]
```

```

inputFile.seekg(0, ios::end);
int fileSize = inputFile.tellg();
inputFile.seekg(0, ios::beg);

if (fileSize <= 0) {
    cerr << "Ошибка: Размер файла " << inputFileName << " равен нулю." << endl;
    inputFile.close();
    return 1;
}

// Читаем содержимое файла в вектор строк
vector<string> lines;
string line;

while (getline(inputFile, line)) {
    lines.push_back(line);
}

inputFile.close();

// Сортируем вектор строк по первому столбцу
sort(lines.begin(), lines.end());

// Открываем выходной файл для записи отсортированных данных
ofstream outputFile(outputFileName);

```

```

if (!outputFile.is_open()) {
    cerr << "Ошибка: Не удалось открыть файл " << outputFileName << " для записи." << endl;
    return 1;
}

// Записываем отсортированные данные в выходной файл
for (const string& sortedLine : lines) {
    outputFile << sortedLine << endl;
}

outputFile.close();

// Открываем выходной файл для чтения и выводим его содержимое на экран
ifstream sortedOutputFile(outputFileName);

if (!sortedOutputFile.is_open()) {
    cerr << "Ошибка: Не удалось открыть файл " << outputFileName << " для чтения." << endl;
    return 1;
}

cout << "Отсортированное содержимое файла " << inputFileName << " по первому столбцу:" << endl;

while (getline(sortedOutputFile, line)) {
    cout << line << endl;
}

```

```

sortedOutputFile.close();

return 0;
}

```

Результат:

```

daniil@ubuntuserver:~$ ./file22 file17.sh
Отсортированное содержимое файла file17.sh по первому столбцу:

# Если пароль верен, отображаем содержимое каталога /etc в длинном формате
# Если пароль неверен, выводим сообщение об ошибке
echo "Использование: $0 <пароль>"
echo "Неверный пароль."
exit 1
ls -la /etc
# Проверяем, был ли передан пароль в качестве аргумента командной строки
# Сравниваем введенный пароль с верным
# Указываем верный пароль
#!/bin/bash
correct_password="0000"
else
fi
fi
if [ "$#" -ne 1 ]; then
if [ "$1" = "$correct_password" ]; then
daniil@ubuntuserver:~$

```

Контрольный вопросы:

- 1) Отличие пользовательских переменных от переменных среды:

Пользовательские переменные - это переменные, определенные и используемые внутри конкретного скрипта или сеанса оболочки. Они имеют локальную область видимости и доступны только в пределах текущего процесса.

Переменные среды - это переменные, доступные для всех процессов в системе. Они определяются на уровне операционной системы и используются для передачи информации между различными программами.

- 2) Математические операции в SHELL:

В оболочке SHELL для выполнения математических операций часто используется команда `expr` или встроенная конструкция `$(( ))`. Пример: `result=$((2 + 2))`.

- 3) Условные операторы в SHELL:

В SHELL используется конструкция if-then-else-fi для выполнения условных операций. Пример:

```
if [ условие ]; then
```

```
    # блок кода, выполняемый при истинном условии
```

```
else
```

```
    # блок кода, выполняемый при ложном условии
```

```
fi
```

#### 4) Принципы построения простых и составных условий:

Простые условия строятся с использованием операторов сравнения (например, -eq, -lt, -gt), а составные условия формируются с использованием логических операторов (&& - логическое "и", || - логическое "или").

#### 5) Циклы в SHELL:

В SHELL используются циклы for, while, и until для выполнения повторяющихся задач.

#### 6) Массивы и модули в SHELL:

В SHELL можно использовать массивы для хранения набора значений. Модули не являются стандартными для оболочек вроде Bash.

#### 7) Чтение параметров командной строки:

Параметры командной строки доступны в скрипте через переменные \$1, \$2, и так далее, где \$1 - это первый параметр, \$2 - второй, и так далее.

#### 8) Как различать ключи и параметры:

Ключи обычно представляют собой опции, передаваемые скрипту с использованием символа -, например, -f. Параметры - это значения, передаваемые скрипту, которые не начинаются с символа -.

#### 9) Чтение данных из файлов:

Для чтения данных из файлов в SHELL используются команды cat, grep, awk, sed и другие.

10) Стандартные дескрипторы файлов:

В SHELL стандартные дескрипторы файлов включают 0 (стандартный ввод), 1 (стандартный вывод) и 2 (стандартный вывод ошибок).

11) Перенаправление вывода:

В SHELL вывод можно перенаправлять с помощью операторов >, <, >> для стандартного ввода, вывода и вывода ошибок.

12) Подавление вывода:

Вывод можно подавить, добавив >/dev/null после команды.

13) Отправка сигналов скриптам:

Команда kill может использоваться для отправки сигналов процессам, включая скрипты.

14) Использование функций:

В SHELL функции определяются и вызываются, как и в других языках программирования.

15) Обработка текстов (чтение, выбор, вставка, замена данных):

Для обработки текста в SHELL часто используются команды sed, awk, grep и различные текстовые потоковые операторы.

16) Отправка сообщений в терминал пользователя:

Для отправки сообщений в терминал пользователя можно использовать команду echo или другие команды вывода.

17) BASH и SHELL – синонимы?

Нет, BASH (Bourne Again SHell) - это одна из оболочек командной строки для Unix-подобных операционных систем. SHELL - более общий термин, описывающий интерфейс командной строки.

18) PowerShell в операционных системах семейства Windows:

PowerShell представляет собой мощный язык сценариев и оболочку для автоматизации задач в операционных системах Windows. Он предоставляет доступ к командам и функциям, аналогичным тем, которые доступны в командной строке, но также имеет расширенные возможности.