



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«МИРЭА - Российский технологический университет»

**РТУ МИРЭА**

---

**Институт информационных технологий (ИИТ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**  
по дисциплине «Технология разработки программных приложений»

**Практическое задание № 2**

Студент группы

ИКБО-04-22 Арефьев А.М

\_\_\_\_\_  
(подпись)

Отчет представлен

«\_\_» \_\_\_\_\_ 2024г.

## ОГЛАВЛЕНИЕ

ЗАДАНИЕ.....	3
Цель работы:.....	3
Часть 1. Основные команды Git 1.....	3
Часть 2. Системы управления репозиториями.....	7
Часть 3. Работа с ветвлением и оформление кода.....	11
Ответы на контрольные вопросы.....	18
Вывод.....	20

# ЗАДАНИЕ

## Цель работы:

Получить навыки по работе с bash.

## Часть 1. Базовые Bash скрипты

1. Напишите сценарий, который выводит дату, время, список зарегистрировавшихся пользователей, и uptime системы и сохраняет эту информацию в файл.
2. Напишите сценарий, который выводит содержимое любого каталога \ или сообщение о том, что его не существует.
3. Напишите сценарий, который с помощью цикла прочитает файл и выведет его содержимое.
4. Напишите сценарий, который с помощью цикла выведет список файлов и директорий из текущего каталога, укажет, что есть файл, а что директория.
5. Напишите сценарий, который подсчитает объем диска, занимаемого директорией. В качестве директории можно выбрать любую директорию в системе.
6. Напишите сценарий, который выведет список всех исполняемых файлов в директории, для которых у текущего пользователя есть права на исполнение.

```

[merc@merc-modern14b5m bash]$ now_date=$(date)
[merc@merc-modern14b5m bash]$ users=$(cat /etc/passwd)
[merc@merc-modern14b5m bash]$ uptime=$(uptime)
[merc@merc-modern14b5m bash]$ echo "date: $now_date" > zadanie1.txt
[merc@merc-modern14b5m bash]$ echo "reg_users: $users" >> zadanie1.txt
[merc@merc-modern14b5m bash]$ echo "uptime: $uptime" >> zadanie1.txt
[merc@merc-modern14b5m bash]$ cat zadanie1.txt
date: Bc 03 map 2024 12:06:21 MSK
reg_users: root:x:0:0::/root:/bin/bash
nobody:x:65534:65534:Kernel Overflow User:/:usr/bin/nologin
dbus:x:81:81:System Message Bus:/:usr/bin/nologin
bin:x:1:1:/:usr/bin/nologin
daemon:x:2:2:/:usr/bin/nologin
mail:x:8:12:/:var/spool/mail:/usr/bin/nologin
ftp:x:14:11:/:srv/ftp:/usr/bin/nologin
http:x:33:33:/:srv/http:/usr/bin/nologin
systemd-coredump:x:980:980:systemd Core Dumper:/:usr/bin/nologin
systemd-network:x:979:979:systemd Network Management:/:usr/bin/nologin
systemd-oom:x:978:978:systemd Userspace OOM Killer:/:usr/bin/nologin
systemd-journal-remote:x:977:977:systemd Journal Remote:/:usr/bin/nologin
systemd-resolve:x:976:976:systemd Resolver:/:usr/bin/nologin
systemd-timesync:x:975:975:systemd Time Synchronization:/:usr/bin/nologin
tss:x:974:974:tss user for tpm2:/:usr/bin/nologin
uuid:x:68:68:/:usr/bin/nologin
dhcpcd:x:973:973:dhcpcd privilege separation:/:usr/bin/nologin
dnsmasq:x:972:972:dnsmasq daemon:/:usr/bin/nologin
_talkd:x:971:971:User for legacy talkd server:/:usr/bin/nologin
polkitd:x:102:102:PolicyKit daemon:/:usr/bin/nologin
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/usr/bin/nologin
rpcuser:x:34:34:RPC Service User:/var/lib/nfs:/usr/bin/nologin
avahi:x:970:970:Avahi mDNS/DNS-SD daemon:/:usr/bin/nologin
colord:x:969:969:Color management daemon:/var/lib/colord:/usr/bin/nologin
fwupd:x:968:968:Firmware update daemon:/var/lib/fwupd:/usr/bin/nologin
gdm:x:120:120:Gnome Display Manager:/var/lib/gdm:/usr/bin/nologin
geoclue:x:967:967:Geoinformation service:/var/lib/geoclue:/usr/bin/nologin
git:x:966:966:git daemon user:/:usr/bin/git-shell
passim:x:965:965:Local Caching Server:/usr/share/empty:/usr/bin/nologin
rtkit:x:133:133:RealtimeKit:/proc:/usr/bin/nologin
usbmux:x:140:140:usbmux user:/:usr/bin/nologin
merc:x:1000:1000:Merc:/home/merc:/bin/zsh
uptime: 12:07:12 up 1:10, 1 user, load average: 0,05, 0,40, 0,49
[merc@merc-modern14b5m bash]$

```

Рисунок 1 — Задание 1

Записываем данные в переменные затем сохраняем в файл при помощи echo.

```

[merc@merc-modern14b5m bash]$ dir=dir; if [ -d "$dir" ]; then echo "Содержимое:"; ls -l $dir; else echo "Каталога $dir нет."; fi
Содержимое:
итого 0
[merc@merc-modern14b5m bash]$ dir=dir1; if [ -d "$dir" ]; then echo "Содержимое:"; ls -l $dir; else echo "Каталога $dir нет."; fi
Каталога dir1 нет.
[merc@merc-modern14b5m bash]$

```

Рисунок 2 — Задание 2

Записываем путь к каталогу в переменную затем пытаемся считать, иначе выводится сообщение что каталога по пути нет.

```
[merc@merc-modern14b5m bash]$ while IFS= read -r line; do echo "$line"; done < "zadanie1.txt"
date: Bc 03 map 2024 12:06:21 MSK
reg_users: root:x:0:0::/root:/bin/bash
nobody:x:65534:65534:Kernel Overflow User:/:usr/bin/nologin
dbus:x:81:81:System Message Bus:/:usr/bin/nologin
bin:x:1:1:/:usr/bin/nologin
daemon:x:2:2:/:usr/bin/nologin
mail:x:8:12:/:var/spool/mail:usr/bin/nologin
ftp:x:14:11:/:srv/ftp:usr/bin/nologin
http:x:33:33:/:srv/http:usr/bin/nologin
systemd-coredump:x:980:980:systemd Core Dumper:/:usr/bin/nologin
systemd-network:x:979:979:systemd Network Management:/:usr/bin/nologin
systemd-oom:x:978:978:systemd Userspace OOM Killer:/:usr/bin/nologin
systemd-journal-remote:x:977:977:systemd Journal Remote:/:usr/bin/nologin
systemd-resolve:x:976:976:systemd Resolver:/:usr/bin/nologin
systemd-timesync:x:975:975:systemd Time Synchronization:/:usr/bin/nologin
tss:x:974:974:tss user for tpm2:/:usr/bin/nologin
uidd:x:68:68:/:usr/bin/nologin
dhcpcd:x:973:973:dhcpcd privilege separation:/:usr/bin/nologin
dnsmasq:x:972:972:dnsmasq daemon:/:usr/bin/nologin
_talkd:x:971:971:User for legacy talkd server:/:usr/bin/nologin
polkitd:x:102:102:PolicyKit daemon:/:usr/bin/nologin
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:usr/bin/nologin
rpcuser:x:34:34:RPC Service User:/var/lib/nfs:usr/bin/nologin
avahi:x:970:970:Avahi mDNS/DNS-SD daemon:/:usr/bin/nologin
colord:x:969:969:Color management daemon:/var/lib/colord:usr/bin/nologin
fwupd:x:968:968:Firmware update daemon:/var/lib/fwupd:usr/bin/nologin
gdm:x:120:120:Gnome Display Manager:/var/lib/gdm:usr/bin/nologin
geoclue:x:967:967:Geoinformation service:/var/lib/geoclue:usr/bin/nologin
git:x:966:966:git daemon user:/:usr/bin/git-shell
passim:x:965:965:Local Caching Server:/usr/share/empty:usr/bin/nologin
rtkit:x:133:133:RealtimeKit:/proc:usr/bin/nologin
usbmux:x:140:140:usbmux user:/:usr/bin/nologin
merc:x:1000:1000:Merc:/home/merc:/bin/zsh
uptime: 12:07:12 up 1:10, 1 user, load average: 0,05, 0,40, 0,49
[merc@merc-modern14b5m bash]$
```

Рисунок 3 — Задание 3

При помощи while обходим все строки и выводим их.

```
[merc@merc-modern14b5m bash]$ for item in $(ls "."); do if [ -d "$item" ]; then echo "$item - директория"; elif [ -f "$item" ]; then echo "$item - файл"; fi; done
dir - директория
zadanie1.txt - файл
[merc@merc-modern14b5m bash]$
```

Рисунок 4 — Задание 4

При помощи for обходим элеметы в текущей папки, и в зависимости от типа выводим файл это или директория.

```
[merc@merc-modern14b5m ~]$ path="repos/4sem"
[merc@merc-modern14b5m ~]$ size=$(du -sh $path | cut -f1)
[merc@merc-modern14b5m ~]$ echo $size
12M
```

Рисунок 5 — Задание 5

При помощи du вычисляем размер, а -sh суммируем в удобном

формате. А при помощи `cut -f1` берем только 12М.

```
[merc@merc-modern14b5m ~]$ for file in $(find $path -type f); do if  
[ -x "$file" ]; then echo "$file"; fi; done;  
repos/4sem/ТПР/1  
repos/4sem/.git/hooks/README.sample  
[merc@merc-modern14b5m ~]$
```

Рисунок 6 — Задание 6

При помощи `-x` проверяем есть ли доступ к файлу если да, выводим.

## Часть 2. Развертка и запуск проекта при помощи Bash Script

### 1. Определение зависимостей проекта

Любой проект зависит от ряда библиотек, которые предоставляют тот или иной функционал. Для развертывания приложения необходимо, чтобы данные библиотеки были установлены в том окружении, где предполагается это самое развертывание. На основании этого для начала необходимо определить, какие зависимости имеет проект. По ссылке <https://www.dropbox.com/s/ija7ax3sj6ysb0p/blocknote-master.tar.gz> расположен проект для скачивания. Будет скачан архив с непонятным названием, распаковать его можно при помощи команды `tar -xvf имя_архива` имя\_директории\_для\_распаковки. Проект написан на языке программирования Python. Необходимо составить список зависимостей проекта в виде `requirements.txt` файла. Данный файл содержит в себе список библиотек, которые необходимо установить в окружение для запуска приложения. Подробнее про составление данного файла можно почитать по ссылке [https://semakin.dev/2020/04/requirements\\_txt/](https://semakin.dev/2020/04/requirements_txt/). Зависимости в Python можно определить по `import`'ам в файлах, однако некоторые библиотеки

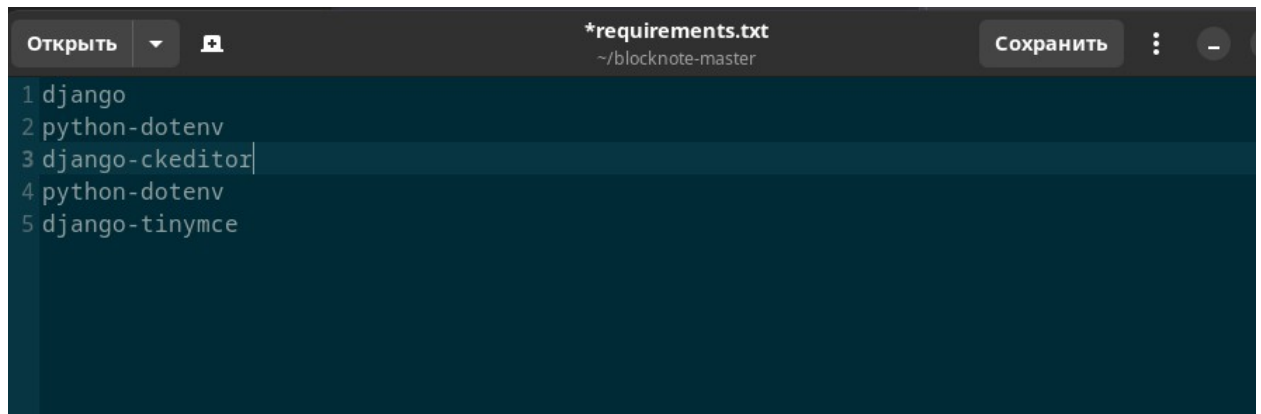
включены в стандартную библиотеку языка, поэтому также необходимо будет определить, является ли библиотека внешней или же встроенной в язык.

2. Создание виртуального окружения Python позволяет создавать так называемое виртуальное окружение. Данное окружение представляет из себя отдельную копию Python с собственным набором библиотек. Оно позволяет работать с проектами, не загрязняя основной интерпретатор ненужными глобально, то есть для всей системы, библиотеками. Подробнее про создание такого рода окружений можно прочитать по ссылке <https://ru.hexlet.io/courses/python-setupenvironment>. Необходимо на основании составленного в прошлом шаге списка команд написать скрипт скачивания указанного в прошлом шаге проекта с последующим созданием виртуального окружения и настройкой его под проект, то есть установкой всех необходимых библиотек.

3. Написание скрипта запуска приложения на новой системе Bash-скрипты позволяют создать с нуля всё необходимое окружение в системе, начиная с установки самого python-а и всего необходимого ПО для запуска приложения и заканчивая запуском самого приложения. Для начала необходимо установить python 3. Сделать это можно при помощи команды `sudo apt install python3`. Далее необходимо загрузить к себе на машину собственный проект. После этого необходимо воссоздать полученное на прошлом этапе виртуальное окружение со всеми зависимостями. 12 Затем необходимо запустить проект из виртуального окружения при помощи следующих команд: `python manage.py makemigrations` `python manage.py migrate` `python manage.py runserver`.

```
~/blocknote-master bash
[merc@merc-modern14b5m blocknote-master]$ chmod +x create_req.sh
[merc@merc-modern14b5m blocknote-master]$ ./create_req.sh
./create_req.sh: строка 14: special_cases: неправильный индекс массива
is part of the standard library, skipping...
django does not exist in PyPI
main does not exist in PyPI
os does not exist in PyPI
pathlib is part of the standard library, skipping...
simple_history does not exist in PyPI
sys does not exist in PyPI
time is part of the standard library, skipping...
requirements.txt has been created.
[merc@merc-modern14b5m blocknote-master]$
```

Рисунок 7 — Запуск bash скрипта для поиска зависимостей



```
Открыть *requirements.txt Сохранить
~/blocknote-master
1 django
2 python-dotenv
3 django-ckeditor
4 python-dotenv
5 django-tinymce
```

Рисунок 8 — Сгенерированный файл с зависимостями



```

~ ➤ cd blocknote-master
~/blocknote-master ➤ python -m venv venv
^[[A^[[[
~/blocknote-master
~/blocknote-master
~/blocknote-master ➤ source venv/bin/activate
~/blocknote-master ➤ python manage.py runserver
Traceback (most recent call last):
  File "/home/merc/blocknote-master/manage.py", line 11, in main
    from django.core.management import execute_from_command_line
ModuleNotFoundError: No module named 'django'

The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "/home/merc/blocknote-master/manage.py", line 22, in <module>
    main()
  File "/home/merc/blocknote-master/manage.py", line 13, in main
    raise ImportError(
ImportError: Couldn't import Django. Are you sure it's installed and available on your PY
~/blocknote-master ➤ pip install -r requirements.txt
Collecting django (from -r requirements.txt (line 1))
  Obtaining dependency information for django from https://files.pythonhosted.org/package
  Using cached Django-5.0.2-py3-none-any.whl.metadata (4.1 kB)
Collecting python-dotenv (from -r requirements.txt (line 2))
  Obtaining dependency information for python-dotenv from https://files.pythonhosted.org/
  Using cached python_dotenv-1.0.1-py3-none-any.whl.metadata (23 kB)
Collecting asgiref<4,>=3.7.0 (from django->-r requirements.txt (line 1))
  Obtaining dependency information for asgiref<4,>=3.7.0 from https://files.pythonhosted.
  Using cached asgiref-3.7.2-py3-none-any.whl.metadata (9.2 kB)
Collecting sqlparse>=0.3.1 (from django->-r requirements.txt (line 1))
  Obtaining dependency information for sqlparse>=0.3.1 from https://files.pythonhosted.or
  Using cached sqlparse-0.4.4-py3-none-any.whl.metadata (4.0 kB)
Using cached Django-5.0.2-py3-none-any.whl (8.2 MB)
Using cached python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Using cached asgiref-3.7.2-py3-none-any.whl (24 kB)
Using cached sqlparse-0.4.4-py3-none-any.whl (41 kB)
Installing collected packages: sqlparse, python-dotenv, asgiref, django
Successfully installed asgiref-3.7.2 django-5.0.2 python-dotenv-1.0.1 sqlparse-0.4.4

[notice] A new release of pip is available: 23.2.1 -> 24.0
[notice] To update, run: pip install --upgrade pip
~/blocknote-master ➤ python manage.py runserver
Watching for file changes with StatReloader

```

Рисунок 9 — Запуск проект и установка зависимостей

```

*requirements.txt
*create_req.sh

1 SEARCH_DIR="."
2
3 TEMP_FILE="temp_imports.txt"
4
5 declare -A special_cases=( ["dotenv"]="python-dotenv" )
6
7 standard_libs=" pathlib time "
8
9 grep -Irho --include \*.py -E "from [^ ]+ import|import [^ ]+" $SEARCH_DIR | \
10 awk '{print $2}' | cut -d. -f1 | \
11 sort | uniq > $TEMP_FILE
12
13 while read -r line; do
14     if [[ ${special_cases[$line]} ]]; then
15         line=${special_cases[$line]}
16     fi
17
18     if [[ " $standard_libs " =~ " $line " ]]; then
19         echo "$line is part of the standard library, skipping..."
20         continue
21     fi
22
23     # Проверка существования модуля на PyPI
24     response=$(curl -s -o /dev/null -w "%{http_code}" https://pypi.org/project/$line/)
25     if [ "$response" == "200" ]; then
26         echo "$line" >> requirements.txt
27     else
28         # Попытка проверить модуль с префиксом "django-"
29         response_django=$(curl -s -o /dev/null -w "%{http_code}" https://pypi.org/project/django-$line/)
30         if [ "$response_django" == "200" ]; then
31             echo "django-$line" >> requirements.txt
32         else
33             echo "$line does not exist in PyPI"
34         fi
35     fi
36 done < $TEMP_FILE
37
38 rm $TEMP_FILE

```

**Рисунок 10 — Скрипт для поиска зависимостей**

Самое сложное было понять какие зависимости добавить в файл, потому что при импорте они выглядят иначе, чем название при установке, для этого применил дополнительные условия и замену по типу dotenv=python-dotenv.

*Листинг 1. Задача 3 — общий bash скрипт*

```

wget https://www.dropbox.com/s/ija7ax3sj6ysb0p/blocknote-master.tar.gz
tar -xvf blocknote-master.tar.gz -C /path/to/unpack/

```

```

cd /path/to/unpack/blocknote-master
sudo pacman -Ss python3
python3 -m venv venv
source venv/bin/activate
SEARCH_DIR="."

TEMP_FILE="temp_imports.txt"

declare -A special_cases=( ["dotenv"]="python-dotenv" )

standard_libs=" pathlib time "

grep -Irho --include \*.py -E "from [^ ]+ import|import [^ ]+" $SEARCH_DIR | \
awk '{print $2}' | cut -d. -f1 | \
sort | uniq > $TEMP_FILE

while read -r line; do
    if [[ ${special_cases[$line]} ]]; then
        line=${special_cases[$line]}
    fi

    if [[ " $standard_libs " =~ " $line " ]]; then
        echo "$line is part of the standard library, skipping..."
        continue
    fi

    # Проверка существования модуля на PyPI
    response=$(curl -s -o /dev/null -w "%{http_code}")

```

```

https://pypi.org/project/$line/)
  if [ "$response" == "200" ]; then
    echo "$line" >> requirements.txt
  else
    # Попытка проверить модуль с префиксом "django-"
    response_django=$(curl -s -o /dev/null -w "%{http_code}")
https://pypi.org/project/django-$line/)
    if [ "$response_django" == "200" ]; then
      echo "django-$line" >> requirements.txt
    else
      echo "$line does not exist in PyPI"
    fi
  fi
done < $TEMP_FILE

rm $TEMP_FILE

pip install -r requirements.txt

python manage.py makemigrations
python manage.py migrate
python manage.py runserver

```

Общий скрипт для полной подготовки и запуска проекта.

## **ВЫВОД**

Получил навыки по работе с bash.