

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**  
**дисциплины «Основы программной инженерии»**

Выполнил:  
Магомедов Имран Борисович  
2 курс, группа ПИЖ-б-о-22-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р.А., кандидат технических  
наук, доцент кафедры  
инфокоммуникаций

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2023 г.

## ТЕМА: Модули и пакеты.

**ЦЕЛЬ РАБОТЫ:** приобретение навыков по работе с модулями и пакетами языка программирования Python версии 3.x.

## МЕТОДИКА ВЫПОЛНЕНИЯ


1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (\*).



Owner \*      Repository name \*

 m4g0med0v ▾ / BSE\_Python\_LR\_2.13

✓ BSE\_Python\_LR\_2.13 is available.

Great repository names are short and memorable. Need inspiration? How about [turbo-octo-memory](#) ?

Description (optional)

- ☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**  
You choose who can see and commit to this repository.

Initialize this repository with:

- ☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: Python ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: MIT License ▾

3. Выполните клонирование созданного репозитория.

```
student-09-523@S-4302020900139 MINGW64 ~/Desktop/Work
$ git clone https://github.com/m4g0med0v/BSE_Python_LR_2.13.git
Cloning into 'BSE_Python_LR_2.13'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

4. Организуйте свой репозиторий в соответствие с моделью ветвления git-flow.



```
~/w/B/BSE_Python_LR_2.13 main git checkout -b "develop" ✓
Переключились на новую ветку «develop»
```

5. Выполните индивидуальные задания. Приведите в отчете скриншоты работы программ решения индивидуального задания.

#### 5.1 Индивидуальное задание 1.

Выполнить индивидуальное задание лабораторной работы 2.11, оформив все функции программы в виде отдельного модуля. Разработанный модуль должен быть подключен в основную программу с помощью одного из вариантов команды `import`. Номер варианта уточнить у преподавателя.

Основной файл.

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  #Выполнить индивидуальное задание лабораторной работы 2.11, оформив все функции
5  #программы в виде отдельного модуля. Разработанный модуль должен быть подключен в
6  #основную программу с помощью одного из вариантов команды import . Номер варианта
7  #уточнить у преподавателя.
8
9  # Составить программу с использованием замыканий для решения задачи.
10 # Вариант 12. Используя замыкания функций, объявите внутреннюю функцию,
11 # которая заключает строку s (s – строка, параметр внутренней функции) в произвольный тег,
12 # содержащийся в переменной tag – параметре внешней функции. Далее,
13 # на вход программы поступает две строки: первая с тегом, вторая с некоторым содержимым.
14 # Вторую строку нужно поместить в тег из первой строки с помощью реализованного замыкания.
15 # Результат выведите на экран.
16
17 import module_for_individual_task_1 as md
18
19
20 if __name__=="__main__":
21     tag_input = input("Введите тег: ")
22     content_input = input("Введите содержимое: ")
23
24     tagged_string_closure = md.create_tagged_string(tag_input)
25     result = tagged_string_closure(content_input)
26
27     print("Результат:", result)

```

Модуль.

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  # Модуль для individual_task_1.py
5
6  def create_tagged_string(tag):
7      def tag_string(content):
8          return f"<{tag}>{content}</{tag}>"
9      return tag_string

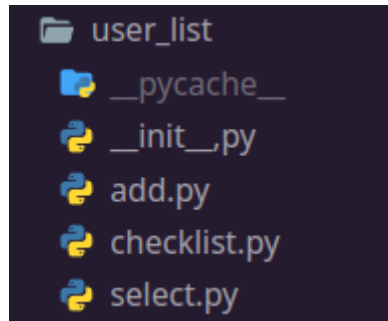
```

## 5.2 Индивидуальное задание 2.

Выполнить индивидуальное задание лабораторной работы 2.8, оформив все классы программы в виде отдельного пакета. Разработанный пакет должен быть подключен в основную программу с помощью одного из вариантов команды import . Настроить соответствующим образом

переменную `__all__` в файле `__init__.py` пакета. Номер варианта уточнить у преподавателя.

Реализовал пакет `user_list`.



Файл `__init__.py`

```
__init__.py ×  
1 #!/usr/bin/env python3  
2 # -*- coding: utf-8 -*-  
3  
4 __all__ = ["add", "checklist", "select"]  
5
```

Файл `add.py`

```
add.py ×  
1 #!/usr/bin/env python3  
2 # -*- coding: utf-8 -*-  
3  
4 def add_users(users):  
5     name = input('Введите Фамилию и Имя: ')  
6     phone_number = input('Введите Номер телефона: ')  
7     year = list(map(int, input('Введите дату рождения (пример: 05 07 2004): ').split()))  
8  
9     user = {  
10         'name': name,  
11         'phone_number': phone_number,  
12         'year': year  
13     }  
14  
15     users.append(user)  
16  
17     if len(users) > 1:  
18         users.sort(key=lambda item: item.get('name', ''))  
19
```

Файл `checklist.py`

```
checklist.py ×
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 def checklist_users(users):
5     line = '+-{}--{}--{}--{}--+'.format('-' * 4, '-' * 20, '-' * 18, '-' * 10)
6     print(line)
7     print('| {:^4} | {:^20} | {:^18} | {:^10} |'.format(
8         '№',
9         'Название',
10        'Номер телефона',
11        'Дата'
12    ))
13    print(line)
14
15    for idx, user in enumerate(users, 1):
16        print(
17            '| {:^4} | {:^20} | {:^18} | {:^10} |'.format(
18                idx,
19                user['name'],
20                user['phone_number'],
21                ' '.join(map(str, user['year']))
22            )
23        )
24    print(line)
25
```

Файл select.py

```

select.py x
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 def select_users(users):
5     print('Введите число месяца (1 - 12): ')
6     while True:
7         num = int(input())
8         if num < 1 or num > 12:
9             print('Значение введено неправильно! Попробуйте еще раз.')
10        else:
11            break
12
13    line = '+-{}-+-{}-+-{}-+-{}-+'.format('-' * 4, '-' * 20, '-' * 18, '-' * 10)
14    print(line)
15    print('| {:^4} | {:^20} | {:^18} | {:^10} |'.format(
16        '№',
17        'Название',
18        'Номер телефона',
19        'Дата'
20    ))
21    print(line)
22
23    for idx, user in enumerate(users, 1):
24        if user['year'][1] == num:
25            print(
26                '| {:^4} | {:^20} | {:^18} | {:^10} |'.format(
27                    idx,
28                    user['name'],
29                    user['phone_number'],
30                    ' '.join(map(str, user['year']))
31                )
32            )
33            print(line)

```

И сам файл individual\_task\_2.py

```

individual_task_2.py x
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 #Выполнить индивидуальное задание лабораторной работы 2.8, оформив все классы программы в виде отдельного пакета.
5 #Разработанный пакет должен быть подключен в основную программу с помощью одного из вариантов команды import .
6 #Настроить соответствующим образом переменную __all__ в файле __init__.py пакета.
7 #Номер варианта уточнить у преподавателя.
8
9
10 from user_list import add, checklist, select
11
12
13 if __name__ == "__main__":
14     users = []
15
16     while True:
17         command = input("$ ").lower()
18
19         if command == 'exit':
20             break
21
22         elif command == 'add':
23             add.add_users(users)
24
25         elif command == 'select':
26             checklist.checklist_users(users)
27
28         elif command == 'list':
29             select.select_users(users)

```

6. Зафиксируйте сделанные изменения в репозитории.

```
~ /w/B/BSE_Python_LR_2.13  git P develop +7  git commit -m 'added individual tasks'
[develop 5e2fe55] added individual tasks
7 files changed, 152 insertions(+)
create mode 100644 individual_task_1.py
create mode 100644 individual_task_2.py
create mode 100644 module_for_individual_task_1.py
create mode 100644 user_list/__init__.py
create mode 100644 user_list/add.py
create mode 100644 user_list/checklist.py
create mode 100644 user_list/select.py
```

7. Добавьте отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксируйте изменения.

8. Выполните слияние ветки для разработки с веткой master/main.

```
~ /work/BSE/BSE_Python_LR_2.13  git P develop  git checkout main
Переключились на ветку «main»
Эта ветка соответствует «origin/main».
~ /work/BSE/BSE_Python_LR_2.13  git P main  git merge develop
Обновление ab5da00..5e2fe55
Fast-forward
individual_task_1.py      | 29 ++++++
individual_task_2.py      | 30 ++++++
module_for_individual_task_1.py | 9 ++++++
user_list/__init__.py     | 5 +++++
user_list/add.py          | 20 ++++++
user_list/checklist.py    | 25 ++++++
user_list/select.py       | 34 ++++++
7 files changed, 152 insertions(+)
create mode 100644 individual_task_1.py
create mode 100644 individual_task_2.py
create mode 100644 module_for_individual_task_1.py
create mode 100644 user_list/__init__.py
create mode 100644 user_list/add.py
create mode 100644 user_list/checklist.py
create mode 100644 user_list/select.py
```

9. Отправьте сделанные изменения на сервер GitHub.

10. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

1 Что является модулем языка Python?

В языке программирования Python модуль - это файл с расширением .py, содержащий код, который можно использовать в других программах.



## 2 Какие существуют способы подключения модулей в языке Python?

Существует несколько способов подключения модулей в Python:

- Использование ключевого слова `import` для загрузки всего модуля.
- Использование ключевого слова `from` для импорта конкретных объектов из модуля.
- Использование `import` с ключевым словом `as` для создания псевдонима при импорте.
- Использование оператора `from module import *` для импорта всех объектов из модуля.

## 3 Что является пакетом языка Python?

Пакет в Python - это специальная директория, содержащая модули и/или другие пакеты. Пакеты позволяют организовать пространство имен и структурировать код.

## 4 Каково назначение файла `__init__.py` ?

Файл `__init__.py` внутри пакета используется для указания, что данная директория должна рассматриваться как пакет, а не просто как обычная директория. Он может также содержать инициализационный код, который выполняется при импорте пакета.

## 5 Каково назначение переменной `__all__` файла `__init__.py` ?

Переменная `__all__` в файле `__init__.py` используется для определения списка имен, которые будут доступны для импорта при использовании оператора `from package import *`. Это предоставляет контроль над тем, какие имена экспортируются из пакета.