

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 №16**  
**дисциплины «Основы программной инженерии»**

Выполнила:  
Ламская Ксения Вячеславовна  
2 курс, группа ПИЖ-б-о-22-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка и  
сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Доцент кафедры инфокоммуникаций  
Воронкин Роман Александрович

---

(подпись)

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

Ставрополь, 2024 г.

**Тема:** Лабораторная работа 2.13. Модули и пакеты.

**Цель работы:** приобретение навыков по работе с модулями и пакетами с помощью языка программирования Python версии 3.x.



Порядок выполнения работы

## 1. Создание репозитория GitHub.

### 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 *
 ksenia-lamskaya ▾	/ 16laba
 16laba is available.	

Great repository names are short and memorable. Need inspiration? How about **congenial-giggle** ?

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 ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  main as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

Рисунок 1 – Создание репозитория

2. 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.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import first_folder.func_first as func_first

if __name__ == "__main__":
    num = [int(a) for a in input("Вводите числа: ").split()]
    test_max = func_first.fun1()
    result = test_max(num)
    print("Результат выполнения программы: ", result)
```

Рисунок 2.1 – Main

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

def fun1(type='max'):
    def fun2(list):
        if type == 'max':
            return max(list)
        else:
            return min(list)
    return fun2
```

Рисунок 2.2 – Func

```
Вводите числа: 4455 55 44 5554 77 3
Результат выполнения программы: 5554
```

Рисунок 2.3 – Результат выполнения программы

3. 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.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys
from funcs import *

if __name__ == '__main__':
    # Список работников.
    workers = []
    print("Список команд:\n")
    print("add - добавить работника;")
    print("list - вывести список работников;")
    print("select <стаж> - запросить работников со стажем;")
    print("help - отобразить справку;")
    print("exit - завершить работу с программой.")
    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о работнике.
            worker = get_worker.get_worker()

            # Добавить словарь в список.
            workers.append(worker)

            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))
```

```

elif command == 'list':
    # Отобразить всех работников.
    display_workers.display_workers(workers)

elif command.startswith('select '):
    # Разбить команду на части для выделения стажа.
    parts = command.split(' ', maxsplit=1)
    # Получить требуемый стаж.
    period = int(parts[1])

    # Выбрать работников с заданным стажем.
    selected = select_workers.select_workers(workers, period)
    # Отобразить выбранных работников.
    display_workers.display_workers(selected)

elif command == 'help':
    # Вывести справку о работе с программой.
    print("Список команд:\n")
    print("add - добавить работника;")
    print("list - вывести список работников;")
    print("select <стаж> - запросить работников со стажем;")
    print("help - отобразить справку;")
    print("exit - завершить работу с программой.")
else:
    print(f"Неизвестная команда {command}", file=sys.stderr)

```

Рисунок 3.1 – Main

```

__all__ = ["select_workers", "display_workers", "get_worker"]

```

Рисунок 3.2 - \_\_init\_\_

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from datetime import date

def select_workers(staff, period):
    """
    Выбрать работников с заданным стажем.
    """
    # Получить текущую дату.
    today = date.today()

    # Сформировать список работников.
    result = []
    for employee in staff:
        if today.year - employee.get('year', today.year) >= period:
            result.append(employee)

    # Возвратить список выбранных работников.
    return result

```

Рисунок 3.3 – select\_workers

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

def display_workers(staff):
    """
    Отобразить список работников.
    """
    # Проверить, что список работников не пуст.
    if staff:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )

        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)

        # Вывести данные о всех сотрудниках.
        for idx, worker in enumerate(staff, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
```

Рисунок 3.4 – display\_workers

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

def get_worker():
    """
    Запросить данные о работнике.
    """
    name = input("Фамилия и инициалы? ")
    post = input("Должность? ")
    year = int(input("Год поступления? "))

    # Создать словарь.
    return {
        'name': name,
        'post': post,
        'year': year,
    }
```

Рисунок 3.5 – get\_worker

## 1. Модуль в Python:

- Модуль в Python - это файл с расширением `.py`, который содержит код на Python. Модуль может содержать функции, классы и другие объекты, которые могут быть использованы в других частях программы.

## 2. Способы подключения модулей в Python:

- `import module_name`: Импортирует весь модуль целиком.  
- `from module_name import function_name`: Импортирует только конкретную функцию из модуля.

- `from module_name import`: Импортирует все объекты из модуля. Использование этого способа не рекомендуется из-за возможности конфликтов имен.

- `import module_name as alias`: Импортирует модуль с псевдонимом.

## 3. Пакет в Python:

Пакет в Python - это папка, которая содержит модули и другие пакеты. Пакет используется для организации и структурирования больших проектов на Python.

## 4. Назначение файла `__init__.py`:

- Файл `__init__.py` внутри пакета Python используется для указания интерпретатору, что данная папка является пакетом. Он может также содержать инициализационный код, который выполняется при импорте пакета.

## 5. Назначение переменной `__all__` в файле `__init__.py`:

- Переменная `__all__` в файле `__init__.py` используется для определения списка модулей, которые будут импортированы при использовании конструкции `from package_name import`. Она задает явный список модулей для экспорта и может помочь в избегании конфликтов имен при импорте всех модулей из пакета.