

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 №27
дисциплины «Основы программной инженерии»

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

(подпись)

Проверил Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: Работа с исключениями в языке Python.

Цель работы: приобретение навыков по работе с исключениями при написании программ с помощью языка программирования Python версии 3.x.

Ход работы.

1. Создание нового репозитория с лицензией MIT.

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 * SashkaHacker / **Repository name *** laba10
✔ laba10 is available.

Great repository names are short and memorable. Need inspiration? How about [fluffy-guide](#) ?

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.](#)

ⓘ You are creating a public repository in your personal account.

Create repository

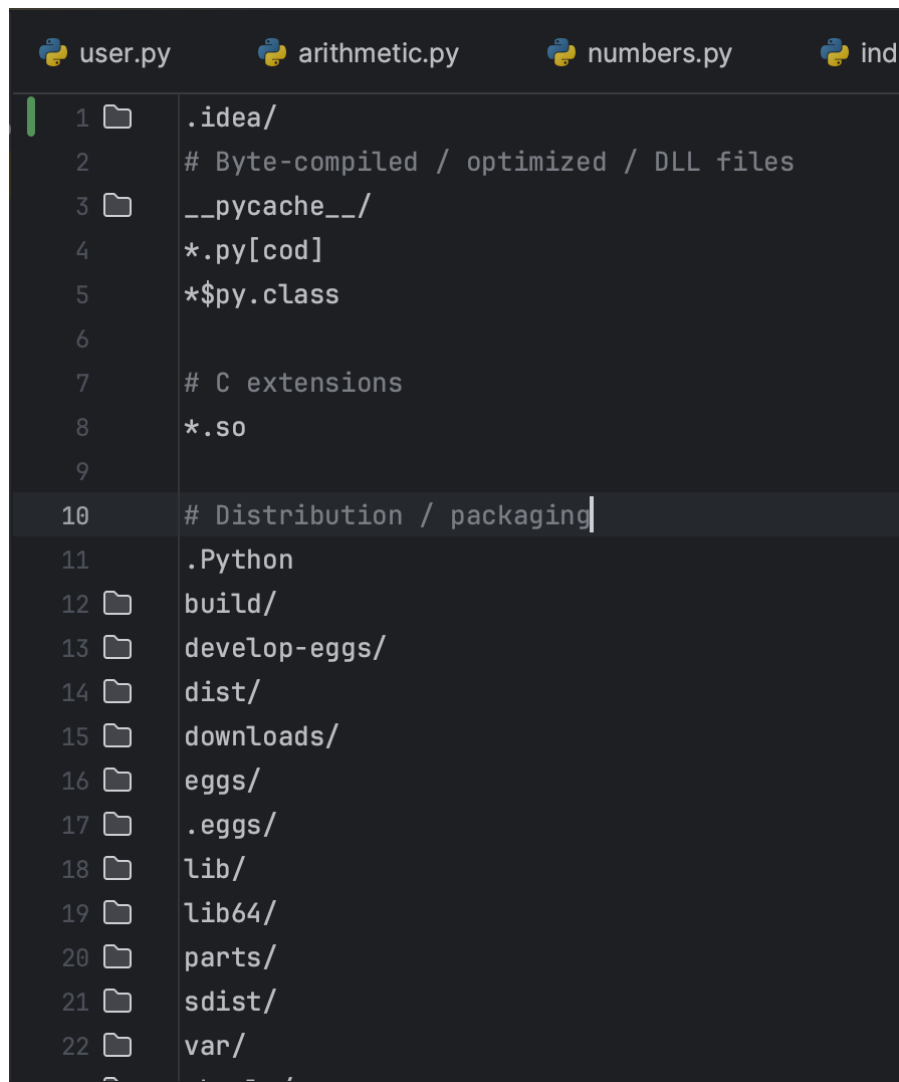
Рисунок 1 – Создание репозитория

2. Клонировал репозиторий на рабочий ПК.

```
[→ GitHub git clone https://github.com/SashkaHacker/laba16.git
Cloning into 'laba16'...
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.
```

Рисунок 2 – Клонирование репозитория

3. Дополнил файл .gitignore необходимыми инструкциями.

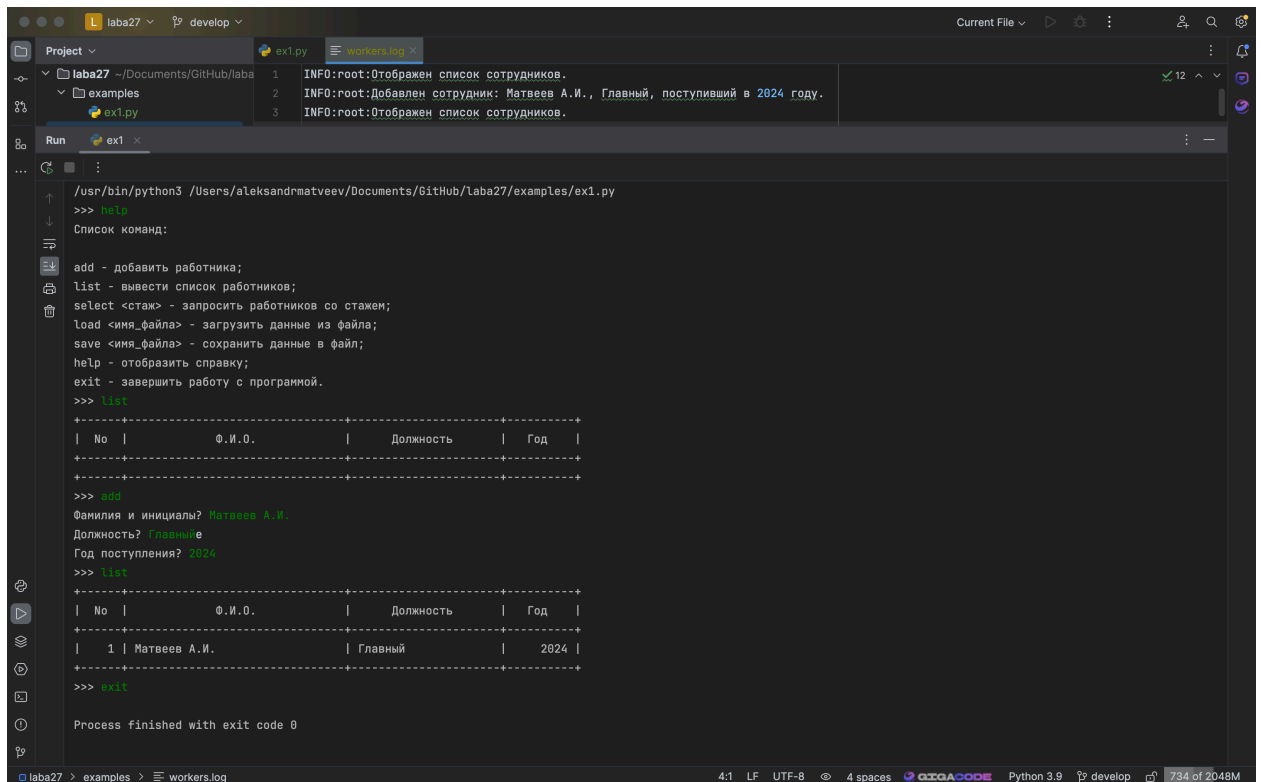


The image shows a code editor with a dark theme. At the top, there are four tabs: 'user.py', 'arithmetic.py', 'numbers.py', and 'ind'. The active tab is 'user.py'. The editor displays the content of a '.gitignore' file. The file is numbered from 1 to 22. The content is as follows:

```
1  .idea/
2  # Byte-compiled / optimized / DLL files
3  __pycache__/
4  *.py[cod]
5  *$py.class
6
7  # C extensions
8  *.so
9
10 # Distribution / packaging
11 .Python
12 build/
13 develop-eggs/
14 dist/
15 downloads/
16 eggs/
17 .eggs/
18 lib/
19 lib64/
20 parts/
21 sdist/
22 var/
```

Рисунок 3 – Файл .gitignore

4. Проработка примера №1.



```
Project ▾ lab27 ~\Documents\GitHub\lab27
  ▾ examples
    ex1.py
  workers.log
1 INFO:root:Отображен список сотрудников.
2 INFO:root:Добавлен сотрудник: Матвеев А.И., Главный, поступивший в 2024 году.
3 INFO:root:Отображен список сотрудников.

Run ex1
...

/usr/bin/python3 /Users/aleksandrmatveev/Documents/GitHub/lab27/examples/ex1.py
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
load <имя_файла> - загрузить данные из файла;
save <имя_файла> - сохранить данные в файл;
help - отобразить справку;
exit - завершить работу с программой.
>>> list
+-----+-----+-----+-----+
| No |      Ф.И.О.      | Должность | Год |
+-----+-----+-----+-----+
>>> add
Фамилия и инициалы? Матвеев А.И.
Должность? Главный
Год поступления? 2024
>>> list
+-----+-----+-----+-----+
| No |      Ф.И.О.      | Должность | Год |
+-----+-----+-----+-----+
| 1 | Матвеев А.И.    | Главный   | 2024 |
+-----+-----+-----+-----+
>>> exit

Process finished with exit code 0
```

Рисунок 4 – Демонстрация работы примера №1

5. Выполнение задания №1.

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

if __name__ == '__main__':
    a, b = input('Первое значение: '), input('Второе значение: ')
    try:
        a, b = int(a), int(b)
        print(f'Результат: {a + b}')
    except ValueError as e:
        print(f'Результат: {a + b}')
```

Рисунок 5 – Код программы

```
/usr/bin/python3 /Users/aleksandrmatveev/Documents/GitHub/laba27/individual/ex1.py
Первое значение: 1
Второе значение: 2
Результат: 3
```

Рисунок 6 – Демонстрация работы задания №1

6. Выполнение задания №2.

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

from random import randint

if __name__ == "__main__":
    rows, cols = input("Введите количество строк: "), input(
        "Введите количество столбцов: "
    )
    try:
        rows = int(rows)
        cols = int(cols)
    except ValueError:
        print("В одном из введенных значений не число")
        exit(1)

    matrix = [[randint(-1000, b: 1000) for _ in range(cols)] for _ in range(rows)]
    for i in range(rows):
        for j in range(cols):
            print(matrix[i][j], end=" ")
        print("\n")
```

Рисунок 7 – Код программы

```

Введите количество строк: 5
Введите количество столбцов: 10
280 370 535 446 -925 -580 -624 -171 749 989

988 824 -845 -949 -150 721 972 418 -821 -782

-446 257 -126 438 -849 992 -136 647 588 -520

84 -108 719 999 388 -504 -373 -215 -674 -772

205 -671 677 686 -845 -328 542 -696 -929 78

```

Рисунок 8 – Демонстрация работы программы

7. Выполнение индивидуального задания №1.

```

/Users/aleksandrmatveev/Documents/GitHub/laba27/venv/bin/python /Users/aleksandrmatveev/Documents/GitHub/laba27/individual/individual1.py
Укажите файл (опционально): data.json
Введите команду: help
add - добавление нового работника
select - данные о работнике по его номеру телефона
exit - завершение программ
list - список работников
help - вывод справки

Введите команду: add
Введите фамилию: Матвеев
Введите имя: Александр
Введите номер телефона: 89614776749
Введите дату рождения в формате (ДД:ММ:ГГГГ): 28:05:2004
Введите команду: show
Неизвестная команда show
Введите команду: list
+-----+-----+-----+-----+-----+
| № |          Фамилия          |      Имя      | Номер телефона | Дата рождения |
+-----+-----+-----+-----+-----+
|  1 | Матвеев                   | Александр     | 89614776749    | 28:05:2004    |
+-----+-----+-----+-----+-----+

```

Рисунок 9 – Демонстрация работы программы

```

1  INFO:__main__:Выведена справочная информация.
2  INFO:__main__:Добавление сотрудника Матвеев Александр
3  ERROR:__main__:Введена неверная команда: show
4  INFO:__main__:Выведена информация о всех сотрудниках.
5  INFO:__main__:Прекращена работа программы.

```

Рисунок 10 – Файл с логами работы программы

8. Выполнение индивидуального задания №2.

```
Укажите файл (опционально): data.json
Введите команду: list
+-----+-----+-----+-----+-----+
| № |          Фамилия          |      Имя      | Номер телефона | Дата рождения |
+-----+-----+-----+-----+-----+
|  1 | Матвеев                  | Александр     | 89614776749    | 28:05:2004    |
+-----+-----+-----+-----+-----+
Введите команду: add
Введите фамилию: Скуфьин
Введите имя: Алексей
Введите номер телефона: 32949824397
Введите дату рождения в формате (ДД:ММ:ГГГГ): 29:04:2000
Введите команду: list
+-----+-----+-----+-----+-----+
| № |          Фамилия          |      Имя      | Номер телефона | Дата рождения |
+-----+-----+-----+-----+-----+
|  1 | Матвеев                  | Александр     | 89614776749    | 28:05:2004    |
|  2 | Скуфьин                  | Алексей       | 32949824397    | 29:04:2000    |
+-----+-----+-----+-----+-----+
Введите команду: exit
```

Рисунок 11 – Демонстрация работы программы

```
INFO:__main__:2024-04-23 15:59:01.043182 Запуск программы.
INFO:__main__:2024-04-23 15:59:05.375265 Выведена информация о всех сотрудниках.
INFO:__main__:2024-04-23 15:59:36.608033 Добавление сотрудника Скуфьин Алексей
INFO:__main__:2024-04-23 15:59:38.820351 Выведена информация о всех сотрудниках.
INFO:__main__:2024-04-23 15:59:43.956858 Прекращена работа программы.
```

Рисунок 12 – Файл с логами работы программы

Контрольные вопросы:

1. В Python существуют следующие виды ошибок:

Синтаксические ошибки (`SyntaxError`): возникают, когда интерпретатор обнаруживает синтаксическую ошибку в коде.

Исключения: ошибочные ситуации во время выполнения программы, которые заставляют программу завершить выполнение.

2. Обработка исключений в Python осуществляется с помощью блоков `try` и `except`.

3. Блок `finally` используется для кода, который должен быть выполнен в любом случае, независимо от того, было ли исключение или нет. Блок `else` выполняется, если в блоке `try` не было исключений:

4. Генерация исключений в Python осуществляется с помощью оператора `raise`.

5. Создание классов пользовательских исключений осуществляется путем наследования от класса `Exception` или его подклассов.

6. Модуль `logging` предназначен для логгирования сообщений. Он предоставляет возможность записывать логи с различным уровнем важности в разные выходы (файл, консоль и т.д.) и конфигурировать формат логов.

7. Модуль `logging` поддерживает следующие уровни логгирования:

DEBUG: Низкий уровень важности, используется для диагностики проблем во время разработки.

INFO: Информационные сообщения о нормальном функционировании программы.

WARNING: Предупреждение о возможных проблемах, которые не мешают работе программы.

ERROR: Ошибка, из-за которой выполнение определенной функции программы не может быть продолжено.

CRITICAL: Очень серьезная ошибка, указывающая на возможный крах программы.