

РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций
«Работа с исключениями в языке Python»

Отчет по лабораторной работе № 4.4
по дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ИВТ-б-о-21-1

Шайдеров Дмитрий Викторович.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2024

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

Порядок выполнения работы:

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

Owner * dshayderov / Repository name * OOP_lw_4.4
✓ OOP_lw_4.4 is available.

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

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](#).

Рисунок 1 - Создание репозитория

2. Выполните клонирование созданного репозитория.

```
C:\Users\Asus\Desktop\Учеба\5 семестр\ООП>git clone https://github.com/dshayderov/OOP_lw_4.4.git
Cloning into 'OOP_lw_4.4'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (11/11), 4.90 KiB | 4.90 MiB/s, done.
Resolving deltas: 100% (2/2), done.
```

Рисунок 2 - Клонирование репозитория

3. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```
C:\Users\Asus\Desktop\Учеба\5 семестр\ООП\ООП_lw_4.4>git checkout -b develop
Switched to a new branch 'develop'

C:\Users\Asus\Desktop\Учеба\5 семестр\ООП\ООП_lw_4.4>
```

Рисунок 3 - Ветвление по модели git-flow

4. Проработать примеры лабораторной работы.

```
C:\Users\Asus\Desktop\Учеба\5 семестр\ООП\ООР_lw_4.4\Project>python primer_1.py
>>> add
Фамилия и инициалы? Петров И.И.
Должность? Директор
Год поступления? 1997
>>> add
Фамилия и инициалы? Иванова А.О.
Должность? Бухгалтер
Год поступления? 2012
>>> list
+-----+-----+-----+-----+
| № | Ф.И.О. | Должность | Год |
+-----+-----+-----+-----+
| 1 | Иванова А.О. | Бухгалтер | 2012 |
| 2 | Петров И.И. | Директор | 1997 |
+-----+-----+-----+-----+
>>> save staf.txt
```

Рисунок 4 - Результат выполнения примера 1

5. Решите следующую задачу: напишите программу, которая запрашивает ввод двух значений. Если хотя бы одно из них не является числом, то должна выполняться конкатенация, т. е. соединение, строк. В остальных случаях введенные числа суммируются.

```
zadanie_1 x
"C:\Users\Asus\Desktop\Учеба\5 семестр\
Первое значение: 5
Второе значение: d
Результат: 5d
Process finished with exit code 0
```

Рисунок 5 - Результат выполнения задания 1

6. Решите следующую задачу: напишите программу, которая будет генерировать матрицу из случайных целых чисел. Пользователь может указать число строк и столбцов, а также диапазон целых чисел. Произведите обработку ошибок ввода пользователя.

```
zadanie_2 x
"C:\Users\Asus\Desktop\Учеба\5 семестр
Количество строк матрицы: 3
Количество столбцов матрицы: 3
Начало диапазона чисел: 4
Конец диапазона чисел: 7
[6, 5, 5]
[7, 7, 5]
[7, 6, 4]
```

Рисунок 6 - Результат выполнения задания 2

7. Выполнить индивидуальные задания.

Задание 1.

Выполнить индивидуальное задание 1 лабораторной работы 2.19, добавив возможность работы с исключениями и логгирование.

Использовать словарь, содержащий следующие ключи: название пункта назначения рейса; номер рейса; тип самолета. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть размещены в алфавитном порядке по названиям пунктов назначения; вывод на экран пунктов назначения и номеров рейсов, обслуживаемых самолетом, тип которого введен с клавиатуры; если таких рейсов нет, выдать на дисплей соответствующее сообщение.

```
C:\Users\Asus\Desktop\Учеба\5 семестр\00П\00P_lw_4.4\Project>python ind_1.py
>>> add
Пункт назначения: Санкт-Петербург
Номер рейса: 312
Тип самолёта: Грузовой
>>> add
Пункт назначения: Москва
Номер рейса: 456
Тип самолёта: Пассажирский
>>> list
+-----+-----+-----+-----+
| № | Пункт назначения | Номер рейса | Тип самолета |
+-----+-----+-----+-----+
| 1 | Москва | 456 | Пассажирский |
| 2 | Санкт-Петербург | 312 | Грузовой |
+-----+-----+-----+-----+
>>> select Грузовой
1: Санкт-Петербург - №312
```

Рисунок 7 - Результат выполнения индивидуального задания 1

Задание 2.

Изучить возможности модуля logging. Добавить для предыдущего задания вывод в файлы лога даты и времени выполнения пользовательской команды с точностью до миллисекунды.

```
2024-02-02 15:16:01.112 Добавлен самолёт: Пассажирский, № рейса 123,
отправляющийся в Москва.
2024-02-02 15:16:03.676 Отображен список самолётов.
```

Рисунок 9 - Результат выполнения индивидуального задания 2

Контрольные вопросы:

1. Какие существуют виды ошибок в языке программирования Python?

Синтаксические ошибки, возникающие, если программа написана с нарушением требований Python к синтаксису, и исключения, если в процессе выполнения возникает ошибка.

2. Как осуществляется обработка исключений в языке программирования Python?

Блок кода, в котором возможно появление исключительной ситуации необходимо поместить во внутрь синтаксической конструкции `try... except`. Если в блоке `try` возникнет ошибка, программа выполнит блок `except`.

3. Для чего нужны блоки `finally` и `else` при обработке исключений?

Не зависимо от того, возникнет или нет во время выполнения кода в блоке `try` исключение, код в блоке `finally` все равно будет выполнен. Если необходимо выполнить какой-то программный код, в случае если в процессе выполнения блока `try` не возникло исключений, то можно использовать оператор `else`.

4. Как осуществляется генерация исключений в языке Python?

Для принудительной генерации исключения используется инструкция `raise`.

5. Как создаются классы пользовательских исключений в языке Python?

Для реализации собственного типа исключения необходимо создать класс, являющийся наследником от одного из классов исключений.

6. Каково назначение модуля `logging`?

Для вывода специальных сообщений, не влияющих на функционирование программы, в Python применяется библиотека логов. Чтобы воспользоваться ею, необходимо выполнить импорт в верхней части файла. С помощью `logging` на Python можно записывать в лог и исключения.

7. Какие уровни логгирования поддерживаются модулем logging? Приведите примеры, в которых могут быть использованы сообщения с этим уровнем логгирования.

- **Debug:** самый низкий уровень логгирования, предназначенный для отладочных сообщений, для вывода диагностической информации о приложении.

- **Info:** этот уровень предназначен для вывода данных о фрагментах кода, работающих так, как ожидается.

- **Warning:** этот уровень логгирования предусматривает вывод предупреждений, он применяется для записи сведений о событиях, на которые программист обычно обращает внимание. Такие события вполне могут привести к проблемам при работе приложения. Если явно не задать уровень логгирования — по умолчанию используется именно warning.

- **Error:** этот уровень логгирования предусматривает вывод сведений об ошибках — о том, что часть приложения работает не так как ожидается, о том, что программа не смогла правильно выполниться.

- **Critical:** этот уровень используется для вывода сведений об очень серьёзных ошибках, наличие которых угрожает нормальному функционированию всего приложения. Если не исправить такую ошибку — это может привести к тому, что приложение прекратит работу.

Вывод: были приобретены навыки по работе с исключениями при написании программ с помощью языка программирования Python версии 3.x.