МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 4.4 «Работа с исключениями в языке Python»

по дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ИВТ-б	5-o-20-	1
Дыбов Д.В. « »20 г.		
Подпись студента		
Работа защищена « »	20	_г.
Проверил Воронкин Р.А	_	
(поличсь)		

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

Ход Работы

- 1. Создал новый репозиторий для лабораторной работы №4.4;
- 2. Клонировал созданный репозиторий на компьютер;
- 3. Создал новый РуСharm проект в папке репозитория;
- 4. Проработал пример:

```
>>> add
Фамилия и инициалы? Иванов И.И.
Должность? Доцент
Год поступления? 2000
>>> list
+----+
| № | Ф.И.О. | Должность | Год |
+----+
| 1 | Иванов И.И. | Доцент | 2000 |
+----+
>>> save workers.json
```

Рисунок 1 – Проверка примера на работоспособность

5. Проверил .log файл:

```
INFO:root:Добавлен сотрудник: Иванов И.И., Доцент, поступивший в 2000 году. INFO:root:Отображен список сотрудников. INFO:root:Сохранены данные в файл workers.json. INFO:root:Загружены данные из файла workers.json.
```

Рисунок 2 – Записи в workers.json

- 6. Выполнил первое индивидуальное задание;
- 7. Проверил задание на работоспособность:

```
PS C:\Program Files\Git\laba4.4> python Individual1.py add airport.json -p="Токио" -n=12 -m=39
PS C:\Program Files\Git\laba4.4> python Individual1.py add airport.json -p="Москва" -n=34 -m=45
PS C:\Program Files\Git\laba4.4> python Individual1.py display airport.json
+----+
| № | Пункт назначения | Номер рейса | Тип самолёта |
+-----+
| 1 | Токио | 12 | 39 |
| 2 | Москва | 34 | 45 |
+-----+
```

Рисунок 3 – Выполнение первого индивидуального задания

8. Проверил airport.json файл:

```
WARNING:root:Файл не найден, создается новый
INFO:root:Добавлен рейс
INFO:root:Данные сохранены в файл: airport.json
INFO:root:Файл найден
INFO:root:Добавлен рейс
INFO:root:Данные сохранены в файл: airport.json
INFO:root:Файл найден
INFO:root:Отображён список рейсов
INFO:root:Файл найден
INFO:root:Добавлен рейс
INFO:root:Данные сохранены в файл: airport.json
INFO:root:Файл найден
INFO:root:Добавлен рейс
INFO:root:Данные сохранены в файл: airport.json
INFO:root:Файл найден
INFO:root:Отображён список рейсов
```

Рисунок 4 – Записи в airport.json

- 9. Выполнил второе индивидуальное задание;
- 10. Проверил задание на работоспособность:

Рисунок 5 – Выполнение второе индивидуального задания

11. Проверил airport.json файл:

```
2022-05-22 22:52:18,722 - WARNING - main: 170 - Файл не найден, создается новый 2022-05-22 22:52:18,722 - INFO - main: 175 - Добавлен рейс 2022-05-22 22:52:18,722 - INFO - save_airplane: 90 - Данные сохранены в файл: airport2.json 2022-05-22 22:52:58,635 - INFO - main: 167 - Файл найден 2022-05-22 22:52:58,635 - INFO - main: 175 - Добавлен рейс 2022-05-22 22:52:58,636 - INFO - save_airplane: 90 - Данные сохранены в файл: airport2.json 2022-05-22 22:53:17,567 - INFO - main: 167 - Файл найден 2022-05-22 22:53:17,569 - INFO - main: 178 - Отображён список рейсов
```

Рисунок 6 – Записи в airport.json

- 12. Выполнил первую задачу;
- 13. Проверил задачу на работоспособность:

```
Введите первое значение: 3
Введите второе значение: 5
Результат: 8
Process finished with exit code 0
```

Рисунок 7 – Результат выполнения первой задачи

- 14. Выполнил первую задачу;
- 15. Проверил задачу на работоспособность:

```
Введите количество столбцов: 4
Введите минимальную границу диапазона: 3
Введите максимальную границу диапазона: 9
[[3, 5, 6, 5], [5, 7, 4, 7], [3, 4, 7, 6], [5, 5, 7, 3], [7, 3, 3, 3]]
Process finished with exit code 0
```

Рисунок 7 – Результат выполнения первой задачи

11. Проверил пример на наличие ошибок:

PythonChecker Makes Your Code Great Again #!/usr/bin/env python3

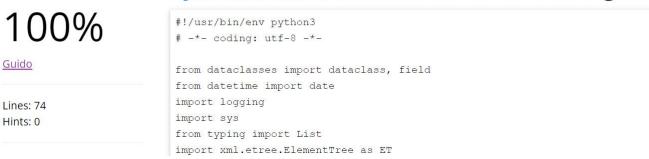


Рисунок 8 – Проверка примера на наличие ошибок

12. Проверил первое индивидуальное задание на наличие ошибок:

PythonChecker Makes Your Code Great Again 100% ##!/usr/bin/env python3 # -*- coding: utf-8 -*-Guido import argparse import json import pathlib Lines: 124 import logging Hints: 0

Рисунок 9 – Проверка первого индивидуального задания на наличие ошибок

13. Проверил второе индивидуальное задание на наличие ошибок:

```
PythonChecker Makes Your Code Great Again
100%
                      ##!/usr/bin/env python3
                      # -*- coding: utf-8 -*-
Guido
                      import argparse
                      import json
                      import pathlib
Lines: 124
                      import logging
Hints: 0
```

Рисунок 10 – Проверка второго индивидуального задания на наличие ошибок

14. Проверил первую задачу на наличие ошибок:

PythonChecker Makes Your Code Great Again

100%

Guido

Lines: 34 Hints: 0 #!/usr/bin/env python3
-*- coding: utf-8 -*
"""

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.

Рисунок 11 – Проверка первой задачи на наличие ошибок

15. Проверил вторую задачу на наличие ошибок:

PythonChecker Makes Your Code Great Again

100%

Guido

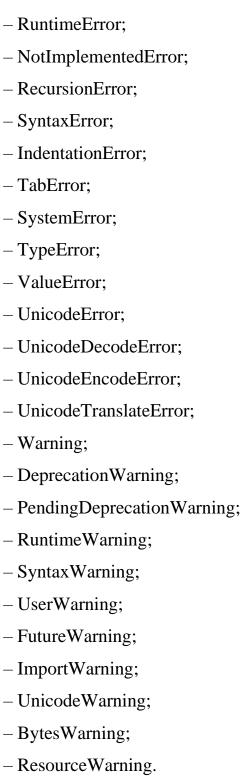
Lines: 34 Hints: 0 #!/usr/bin/env python3
-*- coding: utf-8 -*"""
Решите следующую задачу: напишите программу, которая будет генерировать матрицу
из случайных целых чисел. Пользователь может указать число строк и столбцов,
а также диапазон целых чисел. Произведите обработку ошибок ввода пользователя.
"""

Рисунок 12 – Проверка второй задачи на наличие ошибок

Ответы на контрольные вопросы

- 1. Какие существуют виды ошибок в языке программирования Python?
- SystemExit;
- Keyboard Interrupt;
- GeneratorExit:
- Exception;
- StopIteration;
- StopAsyncIteration;
- ArithmeticError;
- FloatingPointError;
- OverflowError;

ZeroDivisionError; AssertionError; AttributeError; - BufferError: - EOFError; ImportError; ModuleNotFoundError; LookupError; IndexError; - KeyError; - MemoryError; - NameError; UnboundLocalError; - OSError; - Blocking IOError; – ChildProcessError; ConnectionError; BrokenPipeError; ConnectionAbortedError; ConnectionRefusedError; ConnectionResetError; - FileExistsError; – FileNotFoundError; InterruptedError; - IsADirectoryError; NotADirectoryError; PermissionError; - ProcessLookupError; TimeoutError; - ReferenceError;



2. Как осуществляется обработка исключений в языке программирования Python? Обработка исключений нужна для того, чтобы приложение не завершалось аварийно каждый раз, когда возникает исключение. Для этого блок кода, в котором возможно появление исключительной ситуации необходимо поместить во внутрь синтаксической конструкции try... 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 пользовательский 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 журналирования.

DEBUG:root:Debug

message!INFO:root:Info message!

WARNING:root:Warning

message!ERROR:root:Error message!

CRITICAL:root:Critical message!

Вывод: в ходе выполнения лабораторной работы были приобретены простейшие навыки по работе с исключениями в языке программирования Python.