

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 №4
дисциплины «Анализ данных»

Выполнил:
Говоров Егор Юрьевич
2 курс, группа ИВТ-б-о-22-1,
15.04.2024 «Информатика
и вычислительная
техника», направленность
(профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

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

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: Работа с переменными окружения в Python3

Цель работы: приобретение навыков по работе с переменными окружения с помощью языка программирования Python версии 3.x.

Ход работы

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

2. Дополнил файл .gitignore необходимыми правилами.

3. Организовал созданный репозиторий в соответствии с необходимыми требованиями.

4. Проработал примеры лабораторной работы. Создал для них отдельные модули языка Python. Привел в отчете скриншоты результата выполнения программ примеров при различных исходных данных, вводимых с клавиатуры.

```
D:\Users\Admin\Рабочий стол\Data_Analize\Data_Analize_Govorov-4\examples>python example.py add -n "Москва-Питер" -p "student" -y "2004"

D:\Users\Admin\Рабочий стол\Data_Analize\Data_Analize_Govorov-4\examples>python example.py display
+-----+-----+-----+-----+
| № | Ф.И.О. | Должность | Год |
+-----+-----+-----+-----+
| 1 | Egorchik | manager | 2004 |
| 2 | Москва-Питер | student | 2004 |
+-----+-----+-----+-----+

D:\Users\Admin\Рабочий стол\Data_Analize\Data_Analize_Govorov-4\examples>
```

Рисунок 1. Результат работы программы из примера 1

5. Выполнил индивидуальные задания, согласно варианту 5. Привёл в отчете скриншоты работы программ.

Задание. Для своего варианта лабораторной работы 2.17 добавьте возможность получения имени файла данных, используя соответствующую переменную окружения.

```
D:\Users\Admin\Рабочий стол\Data_Analize\Data_Analize_Govorov-4\individ>python ind1.py add -d "Москва-Питер" -n "135" -t "грузовой"
```

```
D:\Users\Admin\Рабочий стол\Data_Analize\Data_Analize_Govorov-4\individ>python ind1.py display
```

Название пункта назначения	Номер рейса	Тип самолета
Москва-Ростов	113	Грузовой
Москва-Питер	135	грузовой

```
D:\Users\Admin\Рабочий стол\Data_Analize\Data_Analize_Govorov-4\individ>
```

Рисунок 2. Результат работы программы из индивидуального задания 1

```
args = parser.parse_args()
data_file = args.filename
if not data_file:
    data_file = os.environ.get("FLIGHTS_DATA")
if not data_file:
    print("Отсутствует имя файла данных", file=sys.stderr)
    sys.exit(1)

is_dirty = False
if os.path.exists(data_file):
    flights = load_flights(data_file)
else:
    flights = []

if args.command == "add":
    flights = add_flight(
        flights,
        args.destination,
        args.number,
        args.type
    )
    is_dirty = True

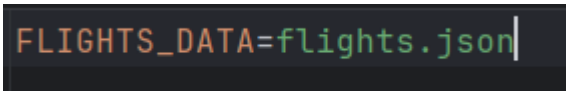
elif args.command == "display":
    display_flights(flights)

elif args.command == "search":
    selected_flights = search_flights_by_destination(flights, args.d
    display_flights(selected_flights)

if is_dirty:
    save_flights(data_file, flights)
```

Рисунок 3. Изменения в коде

Задание. Самостоятельно изучите работу с пакетом python-dotenv . Модифицируйте программу задания 1 таким образом, чтобы значения необходимых переменных окружения считывались из файла .env .



```
FLIGHTS_DATA=flights.json
```

Рисунок 4. Содержимое .env

Контрольные вопросы

1. Назначение переменных окружения заключается в хранении конфигурационной информации, которая может использоваться программами во время их выполнения. Они обеспечивают способ передачи информации между процессами в операционной системе.

2. В переменных окружения может храниться различная информация, такая как пути к исполняемым файлам, настройки языковых сред и окружения, настройки сети, параметры безопасности и т. д.

3. Для доступа к переменным окружения в ОС Windows можно воспользоваться командой set, например:

```
set VARIABLE_NAME=value
```

4. Переменная PATH содержит список директорий, в которых операционная система будет искать исполняемые файлы без указания полного пути. Переменная PATHEXT содержит список расширений файлов, которые будут рассматриваться как исполняемые файлы.

5. Для создания или изменения переменной окружения в Windows можно воспользоваться панелью управления или командной строкой. Например, с помощью команды set:

```
set VARIABLE_NAME=value
```

6. В ОС Linux переменные окружения представляют собой именованные значения, которые могут быть использованы программами и процессами при их выполнении.

7. Переменные окружения предоставляют информацию о конфигурации окружения, доступную для всех процессов, запущенных в рамках этой среды. Переменные оболочки, с другой стороны, являются переменными, доступными только в рамках текущей оболочки.

8. Для вывода значения переменной окружения в Linux можно воспользоваться командой echo:

```
echo $VARIABLE_NAME
```

9. Некоторые известные переменные окружения в Linux:

PATH: список директорий для поиска исполняемых файлов. **HOME:** домашняя директория текущего пользователя.

LANG, LC_ALL: настройки локали и языка.

LD_LIBRARY_PATH: список директорий для поиска динамических библиотек.

10. Некоторые известные переменные оболочки в Linux:

SHELL: путь к исполняемому файлу оболочки.

PS1: строка приглашения командной строки.

PWD: текущая рабочая директория.

11. Для установки переменных оболочки в Linux можно использовать файлы настройки оболочки, такие как ~/.bashrc или ~/.bash_profile.

12. Для установки переменных окружения в Linux также можно использовать файлы настройки оболочки или переменные окружения в системных файловых системах.

13. Делая переменные окружения Linux постоянными, мы гарантируем их доступность для всех процессов, запущенных в рамках этого окружения, включая новые сеансы оболочки и процессы, запущенные из них.

14. Переменная окружения PYTHONHOME используется для указания директории, в которой установлен интерпретатор Python.

15. Переменная окружения PYTHONPATH используется для указания дополнительных директорий, в которых интерпретатор Python будет искать модули.

16. Другие переменные окружения, используемые для управления работой интерпретатора Python, включают PYTHONSTARTUP, PYTHONUSERBASE, PYTHONIOENCODING, PYTHONBREAKPOINT и т. д.

17. Для чтения переменных окружения в программах на языке программирования Python можно использовать модуль os:

```
import os
```

```
variable_value = os.environ.get("VARIABLE_NAME")
```

18. Для проверки, установлено или нет значение переменной окружения, можно воспользоваться методом `get` объекта `os.environ`. Если значение переменной не установлено, метод вернет `None`.

19. Для присвоения значения переменной окружения в программах на языке программирования Python можно использовать методы модуля `os.environ`:

```
import os  
os.environ["VARIABLE_NAME"] = "value"
```

Вывод: в ходе выполнения работы были приобретены навыки с по работе переменными окружения с помощью языка программирования Python версии 3.x.