МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 2.18

«Работа с переменными окружения в Python3»

по дисциплине «Основы программной инженерии»

Выполнил студент группы
ПИЖ-б-о-21-1
Зиберов Александр
« » мая 2023 г.
Подпись студента_____
Работа защищена
« »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь, 2023

Цель работы:

Приобретение построения приложений с интерфейсом командной строки с помощью языка программирования Python версии 3.x.

Выполнение работы:

Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT, рисунок 1.

Ссылка: https://github.com/afk552/lab2.18

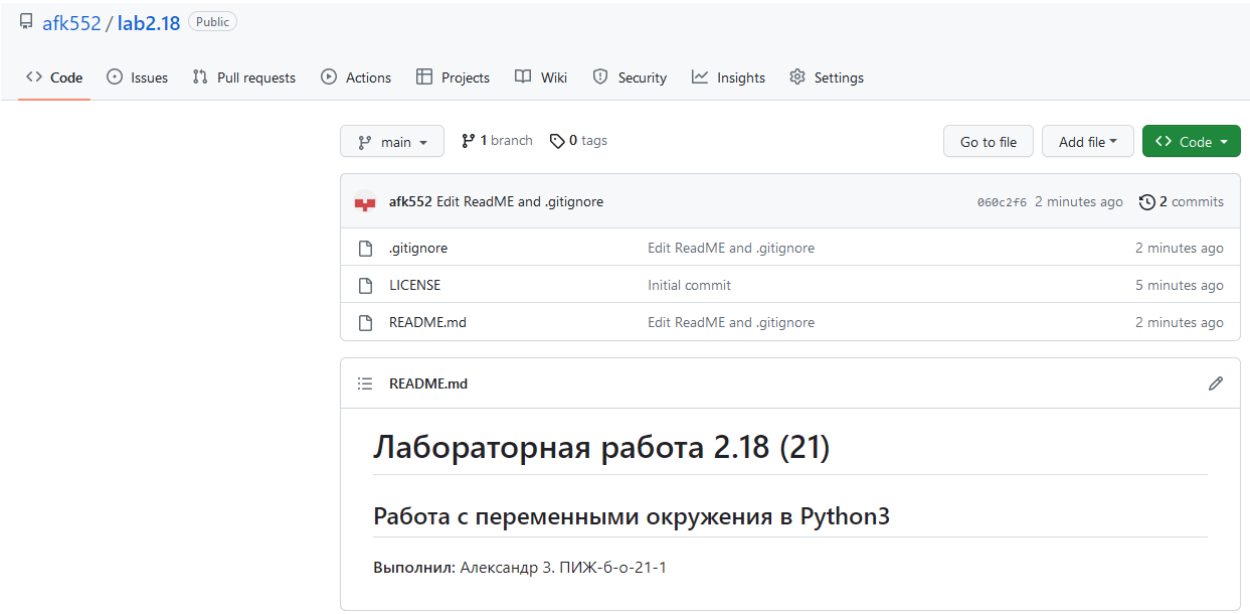

Рисунок 1 – Удаленный репозиторий на GitHub

Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm, рисунок 2.

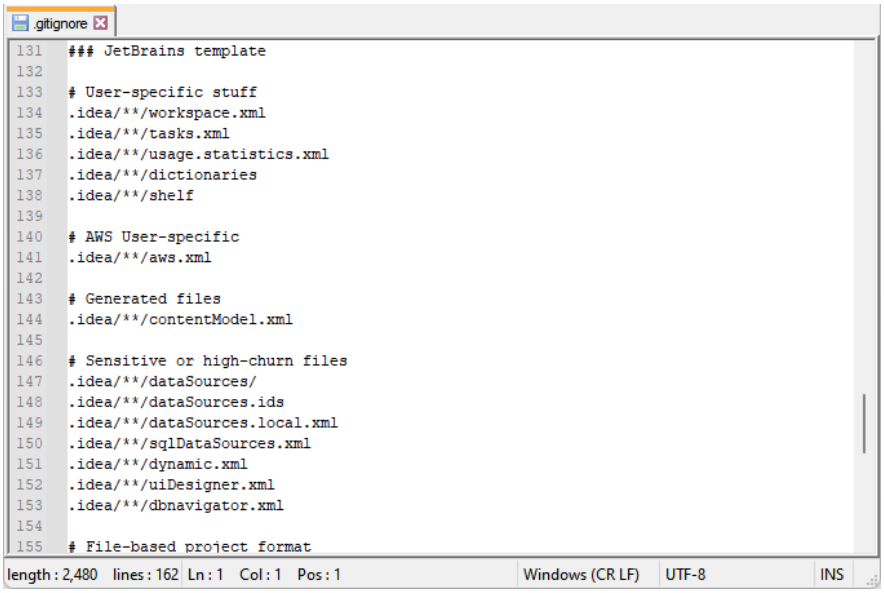

Рисунок 2 – Окно блокнота

Организуйте свой репозиторий в соответствие с моделью ветвления git-flow, рисунок 3.

```
C:\git\lab2.18>git checkout -b develop
Switched to a new branch 'develop'

C:\git\lab2.18>git branch
* develop
  main

C:\git\lab2.18>
```

Рисунок 3 – Окно командной строки

Создайте проект PyCharm в папке репозитория, рисунок 4.

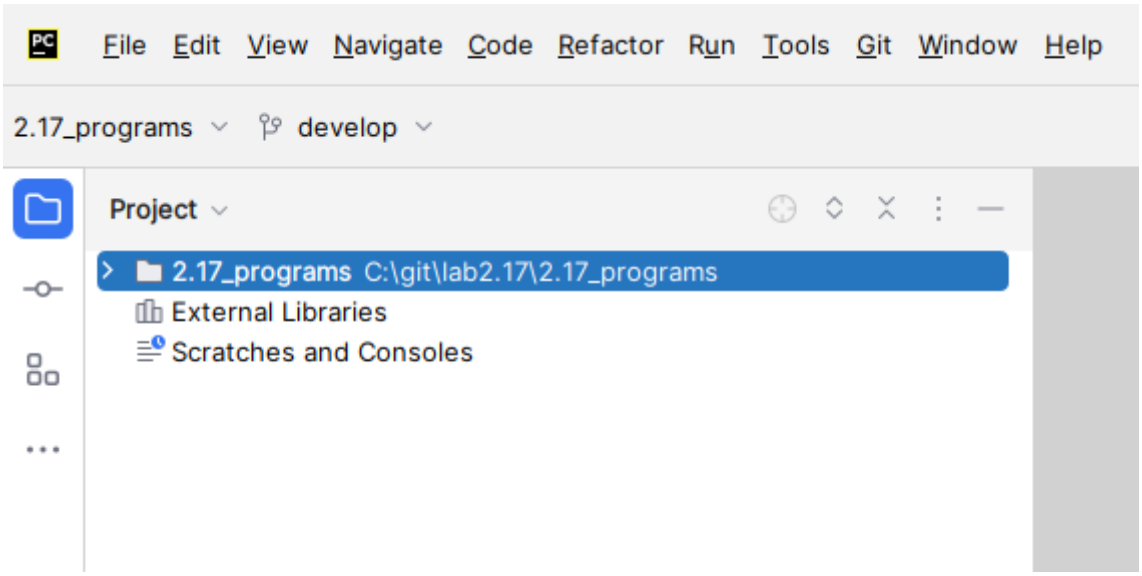

Рисунок 4 – Окно проекта в PyCharm

Примеры

ex1.py

```
1  # Импортируем модуль os
2  import os
3
4  # Создаём цикл, чтобы вывести все переменные среды
5  print("The keys and values of all environment variables:")
6  for key in os.environ:
7      print(key, '=>', os.environ[key])
8
9  # Выводим значение одной переменной
10 print("The value of HOME is: ", os.environ['WINDIR'])
11
```

Рисунок 5 – Код примера 1

```
The value of HOME is:  C:\WINDOWS

Process finished with exit code 0
```

Рисунок 6 – Выполнение примера 1

ex2.py

```python
# Импортируем модуль os
import ...

while True:
    # Принимаем имя переменной среды
    key_value = input("Enter the key of the environment variable:")
    # Проверяем, инициализирована ли переменная
    try:
        if os.environ[key_value]:
            print(
                "The value of",
                key_value,
                " is ",
                os.environ[key_value]
            )
    # Если переменной не присвоено значение, то ошибка
    except KeyError:
        print(key_value, 'environment variable is not set.')
        # Завершаем процесс выполнения скрипта
        sys.exit(1)

```

Рисунок 7 – Код примера 2

```
C:\git\lab2.18\venv\Scripts\python.exe C:\git\lab2.18\examples\ex2.py
Enter the key of the environment variable:windir
The value of windir  is  C:\WINDOWS
Enter the key of the environment variable:1
1 environment variable is not set.

Process finished with exit code 1
```

Рисунок 8 – Выполнение примера 2

```python
# Импортируем модуль os
import os

# Проверяем значение переменной среды
if os.environ.get('DEBUG') == 'True':
    print('Debug mode is on')
else:
    print('Debug mode is off')

```

Рисунок 9 – Код примера 3

```
C:\git\lab2.18\venv\Scripts\python.exe C:\git\lab2.18\examples\ex3.py
Debug mode is off

Process finished with exit code 0
```

Рисунок 10 – Выполнение примера 3

Индивидуальное задание.

Задание 1. Для своего варианта лабораторной работы 2.17 добавьте возможность получения имени файла данных, используя соответствующую переменную окружения.

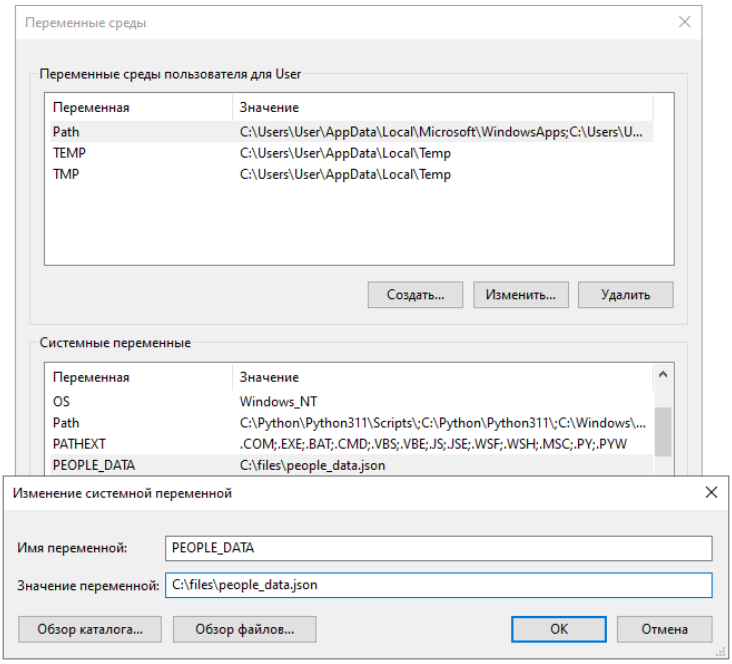

Рисунок 11 – Установка переменной окружения

```
    # Выполнить разбор аргументов командной строки
    args = parser.parse_args(command_line)

    # Получить имя файла
    data_file = args.data
    if not data_file:
        data_file = os.environ.get("PEOPLE_DATA")
        print (data_file)
    if not data_file:
        print("Нет файла с данными!", file=sys.stderr)
        sys.exit(1)
```

Рисунок 12 – Код программы индивидуального задания 1

```
(venv) C:\git\lab2.18>py ind1_var.py display
C:\temp\people_data.json
+------+---------------------------------+----------------+----------------------+
| №п/п |          Фамилия Имя            | Номер телефона |    Дата рождения     |
+------+---------------------------------+----------------+----------------------+
|    1 | Test Testov                     | +7947327472    |          02.02.2002 |
+------+---------------------------------+----------------+----------------------+

(venv) C:\git\lab2.18>
```

Рисунок 13 – Результат выполнения индивидуального задания 1

Задание 2. Самостоятельно изучите работу с пакетом python-dotenv. Модифицируйте программу задания 1 таким образом, чтобы значения необходимых переменных окружения считывались из файла .env.

```
    # Получение значения переменной окружения через python-dotenv
    dotenv_path = os.path.join(os.path.dirname(__file__), '.env')
    if os.path.exists(dotenv_path):
        load_dotenv(dotenv_path)
```

Рисунок 14 – Код программы индивидуального задания 2

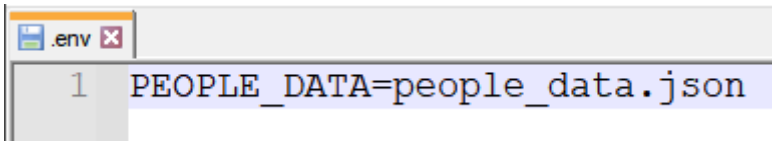

Рисунок 15 – Содержимое файла .env

```
(venv) C:\git\lab2.18\individual>py ind2_dotenv.py display
C:\temp\people_data.json
+------+---------------------------------+----------------+---------------------+
| №п/п |          Фамилия Имя            | Номер телефона |    Дата рождения    |
+------+---------------------------------+----------------+---------------------+
|    1 | Test Testov                     | +7947327472    |          02.02.2002 |
+------+---------------------------------+----------------+---------------------+

(venv) C:\git\lab2.18\individual>
```

Рисунок 16 – Результат выполнения программы индивидуального задания 2

Вывод: В результате выполнения работы была изучена работа с переменными окружения и модулем python-dotenv.

Контрольные вопросы:

1. Каково назначение переменных окружения?

Переменная среды (переменная окружения) – это короткая ссылка на какой-либо объект в системе. С помощью таких сокращений, например, можно создавать универсальные пути для приложений, которые будут работать на любых ПК, независимо от имен пользователей и других параметров.

Переменные окружения Windows дают программам информацию об операционной системе. Ниже в таблице приведен стандартный список переменных окружения, которые создает Windows. Различные программы также создают свои переменные окружения, которые используют в служебных целях для хранения какой-либо информации.

2. Какая информация может храниться в переменных окружения?

Имя ОС; пути, где осуществляется поиск исполняемых файлов; место расположения каталога профиля текущего пользователя и т.д.

3. Как получить доступ к переменным окружения в ОС Windows?

(sysdm.cpl) -> Дополнительно -> Переменные среды

set в командной строке.

4. Каково назначение переменных PATH и PATHEXT?

«PATH» позволяет запускать исполняемые файлы и скрипты, «лежащие» в определенных каталогах, без указания их точного местоположения. Например, если ввести в «Командную строку» explorer.exe, система осуществит поиск по папкам, указанным в значении переменной, найдет и запустит соответствующую программу.

«PATHEXT» Возвращает список расширений файлов, которые рассматриваются операционной системой как исполняемые.

Принцип работы: система перебирает расширения по очереди, пока не будет найден соответствующий объект, причем делает это в директориях, указанных в PATH.

5. Как создать или изменить переменную окружения в Windows?

Нажмите Win+R и введите sysdm.cpl, чтобы быстро открыть свойства системы. Перейдите во вкладку "Дополнительно" и снизу нажмите на "Переменные среды". Переменные среды свойства системы вход. Вы увидите системные и пользовательские переменные среды. Вы можете добавить, удалить или изменить значение для переменных. Системные и пользовательские переменные среды.

6. Что представляют собой переменные окружения в ОС LInux?

Переменные окружения в Linux представляют собой набор именованных значений, используемых другими приложениями.

Переменные окружения применяются для настройки поведения приложений и работы самой системы. Например, переменная окружения может хранить информацию о путях к исполняемым файлам, заданном по умолчанию текстовом редакторе, браузере, языковых параметрах (локали) системы или настройках раскладки клавиатуры.

7. В чем отличие переменных окружения от переменных оболочки?

Переменные окружения (или «переменные среды») — это переменные, доступные в масштабах всей системы и наследуемые всеми дочерними процессами и оболочками.

Переменные оболочки — это переменные, которые применяются только к текущему экземпляру оболочки. Каждая оболочка, например, bash или zsh, имеет свой собственный набор внутренних переменных.

Переменные окружения используются для передачи информации в процессы, которые порождаются из оболочки. Переменные оболочки (Shell Variables) - это переменные, которые содержатся исключительно в оболочке, в которой они были установлены или определены.

8. Как вывести значение переменной окружения в Linux?

Наиболее часто используемая команда для вывода переменных окружения — printenv. Если команде в качестве аргумента передать имя переменной (printenv HOME), то будет отображено значение только этой переменной. Если же вызвать printenv без аргументов, то выведется построчный список всех переменных окружения.

9. Какие переменные окружения Linux Вам известны?

Наиболее распространенных переменные окружения:

USER — текущий пользователь.

PWD — текущая директория.

OLDPWD — предыдущая рабочая директория. Используется оболочкой для того, чтобы вернуться в предыдущий каталог при выполнении команды cd - .

HOME — домашняя директория текущего пользователя.

SHELL — путь к оболочке текущего пользователя (например, bash или zsh).

EDITOR — заданный по умолчанию редактор. Этот редактор будет вызываться в ответ на команду edit .

LOGNAME — имя пользователя, используемое для входа в систему.

PATH — пути к каталогам, в которых будет производиться поиск вызываемых команд. При выполнении команды система будет проходить по данным каталогам в указанном порядке и выберет первый из них, в котором будет находиться исполняемый файл искомой команды.

LANG — текущие настройки языка и кодировки.

TERM — тип текущего эмулятора терминала.

MAIL — место хранения почты текущего пользователя.

LS_COLORS — задает цвета, используемые для выделения объектов (например, различные типы файлов в выводе команды ls будут выделены разными цветами).

10. Какие переменные оболочки Linux Вам известны?

Наиболее распространенные переменные оболочки:

BASHOPTS — список задействованных параметров оболочки, разделенных двоеточием.

BASH_VERSION — версия запущенной оболочки bash.

COLUMNS — количество столбцов, которые используются для отображения выходных данных.

DIRSTACK — стек директорий, к которому можно применять команды pushd и popd .

HISTFILESIZE — максимальное количество строк для файла истории команд.

HISTSIZE — количество строк из файла истории команд, которые можно хранить в памяти.

HOSTNAME — имя текущего хоста.

IFS — внутренний разделитель поля в командной строке (по умолчанию используется пробел).

PS1 — определяет внешний вид строки приглашения ввода новых команд.

PS2 — вторичная строка приглашения.

SHELLOPTS — параметры оболочки, которые можно устанавливать с помощью команды set .

UID — идентификатор текущего пользователя.

11. Как установить переменные оболочки в Linux?

В терминале ввести: NEW_VAR='var_name'

Проверить можно командой: printenv NEW_VAR

12. Как установить переменные окружения в Linux?

Команда export используется для задания переменных окружения.

Для создания переменной окружения экспортируем нашу недавно созданную переменную оболочки: export NEW_VAR

Проверить можно командой: printenv NEW_VAR

13. Для чего необходимо делать переменные окружения Linux постоянными?

Это необходимо, если нужно, чтобы переменная сохранялась после закрытия сеанса оболочки. Для этого, необходимо прописать её в специальном файле. Прописать переменную можно как для текущего пользователя, так и для всех пользователей.

14. Для чего используется переменная окружения PYTHONHOME?

Переменная среды PYTHONHOME изменяет расположение стандартных библиотек Python. По умолчанию библиотеки ищутся в prefix/lib/pythonversion и exec_prefix/lib/pythonversion , где prefix и exec_prefix - это каталоги, зависящие от установки, оба каталога по умолчанию - /usr/local . Когда для PYTHONHOME задан один каталог, его значение заменяет prefix и

exec_prefix . Чтобы указать для них разные значения, установите для PYTHONHOME значение prefix:exec_prefix.

15. Для чего используется переменная окружения PYTHONPATH?

Переменная среды PYTHONPATH изменяет путь поиска по умолчанию для файлов модуля. Формат такой же, как для оболочки PATH : один или несколько путей к каталогам, разделенных os.pathsep (например, двоеточие в Unix или точка с запятой в Windows). Несуществующие каталоги игнорируются. $ unset NEW_VAR Помимо обычных каталогов, отдельные записи PYTHONPATH могут относиться к zip-файлам, содержащим чистые модули Python в исходной или скомпилированной форме. Модули расширения нельзя импортировать из zip-файлов. Путь поиска по умолчанию зависит от установки Python, но обычно начинается с префикса /lib/pythonversion . Он всегда добавляется к PYTHONPATH.

16. Какие еще переменные окружения используются для управления работой интерпретатора Python?

PYTHONSTARTUP, PYTHONBREAKPOINT, PYTHONDEBUG, PYTHONINSPECT, PYTHONBUFFERED, PYTHONVERBOSE, PYTHONCASEOK, PYTHONDONTWRITEBYTECODE, PYTHONPYCACHEPREFIX, PYTHONHASHSEED и другие.

17. Как осуществляется чтение переменных окружения в программах на языке программирования Python?

Для начала потребуется импортировать модуль os, чтобы считывать переменные. Для доступа к переменным среды в Python используется объект os.environ . С его помощью программист может получить и изменить значения всех переменных среды.

Следующий код позволяет прочитать и вывести все переменные окружения, а также определенную переменную. Для вывода имен и значений

всех переменных используется цикл for . Затем выводится значение переменной HOME.

```
# Импортируем модуль os
import os
# Создаём цикл, чтобы вывести все переменные среды
print("The keys and values of all environment variables:")
for key in os.environ:
    print(key, '=>', os.environ[key])
# Выводим значение одной переменной
print("The value of HOME is: ", os.environ['HOME'])
```

После выполнения скрипта мы увидим следующий результат. Сперва был выведен список всех переменных окружения, а затем – значение переменной HOME.

18. Как проверить, установлено или нет значение переменной окружения в программах на языке программирования Python?

Бесконечный цикл while непрерывно принимает от пользователя имена переменных и проверяет их значения до тех пор, пока пользователь не введёт имя переменной, которой не присвоено значение. Если пользователь вводит имя переменной окружения, которой присвоено значение, это значение выводится, если же нет — выводится соответствующее сообщение и процесс останавливается.

19. Как присвоить значение переменной окружения в программах на языке программирования Python?

Создаём Python-файл со следующим кодом. Для проверки переменной DEBUG на истинность здесь используется функция get() . Программа выводит разные сообщения в зависимости от значения переменной.

```
# Импортируем модуль os
import os
```

```
# Проверяем значение переменной среды
if os.environ.get('DEBUG') == 'True':
    print('Debug mode is on')
else:
    print('Debug mode is off')
```