

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:  
Середа Кирилл Витальевич  
2 курс, группа ИВТ-б-о-22-1,  
09.03.01 «Информатика и  
вычислительная техника», очная  
форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Роман Александрович

---

(подпись)

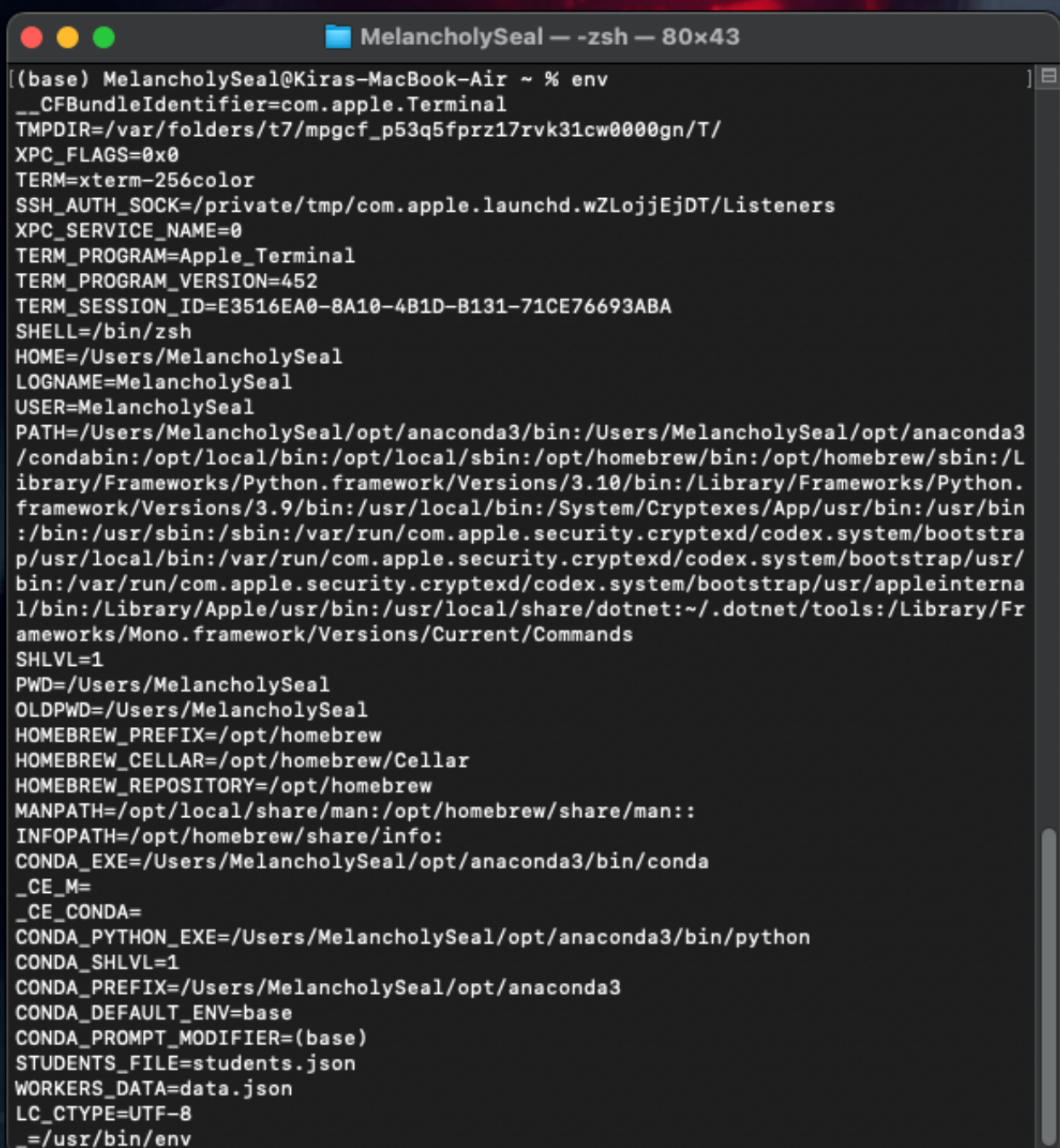
Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2024 г.

Тема: Разработка приложений с интерфейсом командной строки (CLI) в Python3

Цель: приобретение построения приложений с интерфейсом командной строки с помощью языка программирования Python версии 3.x.

Ход выполнения заданий

A screenshot of a macOS terminal window titled "MelancholySeal — -zsh — 80x43". The terminal displays the output of the 'env' command, listing various environment variables. The variables include system paths like PATH, user-specific paths like HOME and PWD, and configuration variables like CONDA\_PREFIX and CONDA\_DEFAULT\_ENV. The terminal text is as follows:

```
[(base) MelancholySeal@Kiras-MacBook-Air ~ % env
__CFBundleIdentifier=com.apple.Terminal
TMPDIR=/var/folders/t7/mpgcf_p53q5fprz17rvk31cw0000gn/T/
XPC_FLAGS=0x0
TERM=xterm-256color
SSH_AUTH_SOCK=/private/tmp/com.apple.launchd.wZLojjEjDT/Listeners
XPC_SERVICE_NAME=0
TERM_PROGRAM=Apple_Terminal
TERM_PROGRAM_VERSION=452
TERM_SESSION_ID=E3516EA0-8A10-4B1D-B131-71CE76693ABA
SHELL=/bin/zsh
HOME=/Users/MelancholySeal
LOGNAME=MelancholySeal
USER=MelancholySeal
PATH=/Users/MelancholySeal/opt/anaconda3/bin:/Users/MelancholySeal/opt/anaconda3/condabin:/opt/local/bin:/opt/local/sbin:/opt/homebrew/bin:/opt/homebrew/sbin:/Library/Frameworks/Python.framework/Versions/3.10/bin:/Library/Frameworks/Python.framework/Versions/3.9/bin:/usr/local/bin:/System/Cryptexes/App/usr/bin:/usr/bin:/bin:/usr/sbin:/sbin:/var/run/com.apple.security.cryptexd/codex.system/bootstrap/usr/local/bin:/var/run/com.apple.security.cryptexd/codex.system/bootstrap/usr/bin:/var/run/com.apple.security.cryptexd/codex.system/bootstrap/usr/appleinternal/bin:/Library/Apple/usr/bin:/usr/local/share/dotnet:~/dotnet/tools:/Library/Frameworks/Mono.framework/Versions/Current/Commands
SHLVL=1
PWD=/Users/MelancholySeal
OLDPWD=/Users/MelancholySeal
HOMEBREW_PREFIX=/opt/homebrew
HOMEBREW_CELLAR=/opt/homebrew/Cellar
HOMEBREW_REPOSITORY=/opt/homebrew
MANPATH=/opt/local/share/man:/opt/homebrew/share/man::
INFOPATH=/opt/homebrew/share/info:
CONDA_EXE=/Users/MelancholySeal/opt/anaconda3/bin/conda
_CE_M=
_CE_CONDA=
CONDA_PYTHON_EXE=/Users/MelancholySeal/opt/anaconda3/bin/python
CONDA_SHLVL=1
CONDA_PREFIX=/Users/MelancholySeal/opt/anaconda3
CONDA_DEFAULT_ENV=base
CONDA_PROMPT_MODIFIER=(base)
STUDENTS_FILE=students.json
WORKERS_DATA=data.json
LC_CTYPE=UTF-8
_=/usr/bin/env
```

Рисунок 1 – Переменные окружения

- 1) Проработал пример лабораторной работы

```
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python exml.py --help
usage: workers [-h] [--version] {add,display,select} ...

positional arguments:
  {add,display,select}
    add                Add a new worker
    display            Display all workers
    select             Select the workers

optional arguments:
  -h, --help            show this help message and exit
  --version             show program's version number and exit
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python exml.py add -d='data.json' -n='Анатолий' -p='Кассир' -y='2019'
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python exml.py display -d='data.json'
```

№	Ф.И.О.	Должность	Год
1	Анатолий	Кассир	2019

```
(DA_4) MelancholySeal@Kiras-MacBook-Air prog %
```

Рисунок 2 – Использование примера

## 2) Сделал индивидуальное задание 1

### Задание 1

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

```
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python ind1.py add -f people.json -n "Иван Иванов" -b 1990-01-01 -p 1234567890
Файл people.json не существует.
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python ind1.py find -f people.json -p 1234567890
Ф.И.О.: Иван Иванов
Дата рождения: 1990-01-01
Телефон: 1234567890
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python ind1.py display -f people.json
```

Ф.И.О.	Дата рождения	Телефон
Иван Иванов	1990-01-01	1234567890

```
(DA_4) MelancholySeal@Kiras-MacBook-Air prog %
```

Рисунок 3 – Вывод индивидуального задания 1

## 3) Сделал индивидуальное задание 2

### Задание 2

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

```

CONDA_PROMPT_MODIFIER=(DA_4)
CONDA_PREFIX_1=/Users/MelancholySeal/opt/anaconda3
PEOPLE_FILE=people.json
_=/usr/bin/env
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python ind2.py add -n "Иван Иванов" -b 1990-01-01 -p 1234567890
Файл people.json не существует.
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python ind2.py display
+-----+-----+-----+
|          Ф.И.О.          |   Дата рождения   |   Телефон   |
+-----+-----+-----+
| Иван Иванов              | 1990-01-01        | 1234567890  |
+-----+-----+-----+
(DA_4) MelancholySeal@Kiras-MacBook-Air prog % python ind2.py find -p 1234567890
Ф.И.О.: Иван Иванов
Дата рождения: 1990-01-01
Телефон: 1234567890
(DA_4) MelancholySeal@Kiras-MacBook-Air prog %

```

Рисунок 4 – Вывод индивидуального задания 2

### Ответы на вопросы

1. **Каково назначение переменных окружения?** Переменные окружения используются для передачи информации процессам, которые запущены в оболочке. Они позволяют конфигурировать поведение программ, предоставляя параметры, такие как пути к исполняемым файлам, настройки локали и системные параметры.

2. **Какая информация может храниться в переменных окружения?** Переменные среды хранят информацию о среде операционной системы. Эта информация может включать такие сведения, как путь к операционной системе, количество процессоров, используемых операционной системой, расположение временных папок, а также специфичные настройки для различных приложений и процессов.

3. **Как получить доступ к переменным окружения в ОС Windows?** Нужно открыть окно "Свойства системы" и нажать на кнопку "Переменные среды". Также можно получить доступ через командную строку командой `echo %VARIABLE_NAME%` или через PowerShell командой `$env:VARIABLE_NAME`.

4. **Каково назначение переменных PATH и PATHEXT?**

- **PATH** позволяет запускать исполняемые файлы и скрипты, расположенные в определенных каталогах, без указания их точного местоположения.

- **PATHEXT** дает возможность не указывать расширение файла, если оно прописано в ее значениях, тем самым упрощая выполнение команд.

5. **Как создать или изменить переменную окружения в Windows?** В окне "Переменные среды" нужно нажать на кнопку "Создать" или "Изменить", затем ввести имя переменной и значение (путь или данные).

6. **Что представляют собой переменные окружения в ОС Linux?** Переменные окружения в Linux представляют собой набор именованных значений, используемых операционной системой и приложениями для настройки среды выполнения и передачи конфигурационных параметров.

7. **В чем отличие переменных окружения от переменных оболочки?**

- **Переменные окружения** доступны в масштабах всей системы и наследуются всеми дочерними процессами и оболочками.

- **Переменные оболочки** локальны для текущей сессии оболочки и не наследуются дочерними процессами, если не экспортированы командой **export**.

8. **Как вывести значение переменной окружения в Linux?** Наиболее часто используемая команда для вывода переменных окружения – **printenv** или **echo**:

```
printenv VARIABLE_NAME
```

```
echo $VARIABLE_NAME
```

9. **Какие переменные окружения Linux Вам известны?**

- **USER:** текущий пользователь.
- **PWD:** текущая директория.
- **HOME:** домашняя директория текущего пользователя.
- **SHELL:** путь к оболочке текущего пользователя.
- **EDITOR:** заданный по умолчанию редактор.
- **LOGNAME:** имя пользователя, используемое для входа в систему.

- **PATH:** пути к каталогам, в которых будет производиться поиск команд.

- **LANG:** текущие настройки языка и кодировки.
- **TERM:** тип текущего эмулятора терминала.
- **MAIL:** место хранения почты текущего пользователя.
- **LS\_COLORS:** задает цвета для команды **ls**.

10. **Какие переменные оболочки Linux Вам известны?**

- **BASHOPTS:** список задействованных параметров оболочки.
- **BASH\_VERSION:** версия запущенной оболочки **bash**.
- **COLUMNS:** количество столбцов для отображения выходных данных.

- **HISTFILESIZE:** максимальное количество строк для файла истории команд.

- **HISTSIZE:** количество строк из файла истории команд, хранимых в памяти.

- **HOSTNAME:** имя текущего хоста.
- **IFS:** внутренний разделитель поля в командной строке.
- **PS1:** внешний вид строки приглашения ввода команд.
- **PS2:** вторичная строка приглашения.
- **UID:** идентификатор текущего пользователя.

11. **Как установить переменные оболочки в Linux?**

**NEW\_VAR='значение'**

12. **как установить переменные окружения в Linux?** Команда **export** используется для задания переменных окружения:

**export VARIABLE\_NAME='значение'**

13. **Для чего необходимо делать переменные окружения Linux постоянными?** Чтобы переменные сохранялись после закрытия сеанса оболочки и были доступны при каждом запуске системы или новой оболочки. Это позволяет использовать их в постоянной конфигурации среды.

14. Для чего используется переменная окружения **PYTHONHOME**? Переменная среды **PYTHONHOME** изменяет расположение стандартных библиотек Python, указывая альтернативный путь к каталогу, где находятся компоненты Python.

15. Для чего используется переменная окружения **PYTHONPATH**? Переменная среды **PYTHONPATH** изменяет путь поиска по умолчанию для файлов модулей, добавляя дополнительные директории к стандартным путям поиска.

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

- **PYTHONSTARTUP**
- **PYTHONOPTIMIZE**
- **PYTHONBREAKPOINT**
- **PYTHONDEBUG**
- **PYTHONINSPECT**
- **PYTHONUNBUFFERED**
- **PYTHONVERBOSE**
- **PYTHONCASEOK**
- **PYTHONDONTWRITEBYTECODE**
- **PYTHONPYCACHEPREFIX**
- **PYTHONHASHSEED**
- **PYTHONIOENCODING**
- **PYTHONNOUSERSITE**
- **PYTHONUSERBASE**
- **PYTHONWARNINGS**
- **PYTHONFAULTHANDLER**

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

```
import os  
  
value = os.environ.get('MY_ENV_VARIABLE')
```

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

```
import os

if 'MY_ENV_VARIABLE' in os.environ:
    print("Переменная окружения установлена.")
else:
    print("Переменная окружения не установлена.")
```

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

```
import os

os.environ['MY_ENV_VARIABLE'] = 'value'
```

**Вывод**

В ходе выполнения работы приобретены навыки работы с переменными окружения, использования их в программах на языке программирования Python версии 3.x, а также навыки конфигурирования окружения для различных систем и приложений