

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ-
СКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования**

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №2.18

Тема: «Работа с
переменными окружения в Python3»

Выполнил студент группы

ИВТ-б-о-21-1

Назаров Н.Ю « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель

Воронкин Р.А.



(подпись)

Ставрополь 2022
Выполнение работы.

Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * **Repository name ***

 NikitaNazarov179 ▾ / 2.18 

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

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:
Skip this step if you're importing an existing repository.


☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)


Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▾

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

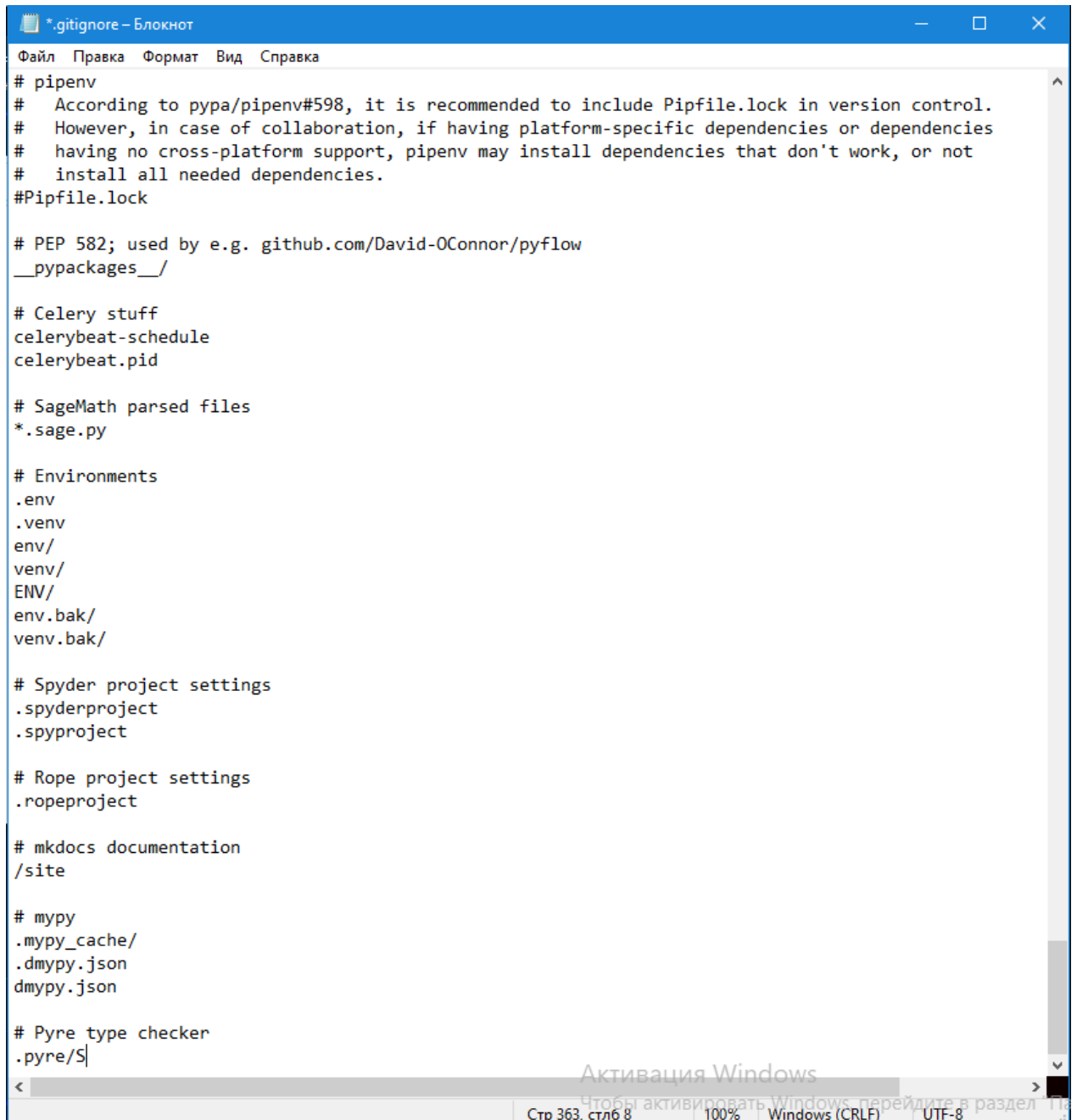
License: MIT License ▾

This will set  **main** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

[Create repository](#)

Рисунок 1 – создал репозиторий



```
*.gitignore – Блокнот
Файл  Правка  Формат  Вид  Справка

# pipenv
#   According to pypa/pipenv#598, it is recommended to include Pipfile.lock in version control.
#   However, in case of collaboration, if having platform-specific dependencies or dependencies
#   having no cross-platform support, pipenv may install dependencies that don't work, or not
#   install all needed dependencies.
#Pipfile.lock

# PEP 582; used by e.g. github.com/David-OConnor/pyflow
__pypackages__

# Celery stuff
celerybeat-schedule
celerybeat.pid

# SageMath parsed files
*.sage.py

# Environments
.env
.venv
env/
venv/
ENV/
env.bak/
venv.bak/

# Spyder project settings
.spyderproject
.spyproject

# Rope project settings
.ropeproject

# mkdocs documentation
/site

# mypy
.mypy_cache/
.dmypy.json
dmypy.json

# Pyre type checker
.pyre/S
```

Рисунок 2 – редактирование gitignore

```
H:\cross\git\nazarov\2.18>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [H:/cross/git/nazarov/2.18/.git/hooks]

H:\cross\git\nazarov\2.18>
```

Рисунок 3 – организация репозитория в соответствии с gitflow

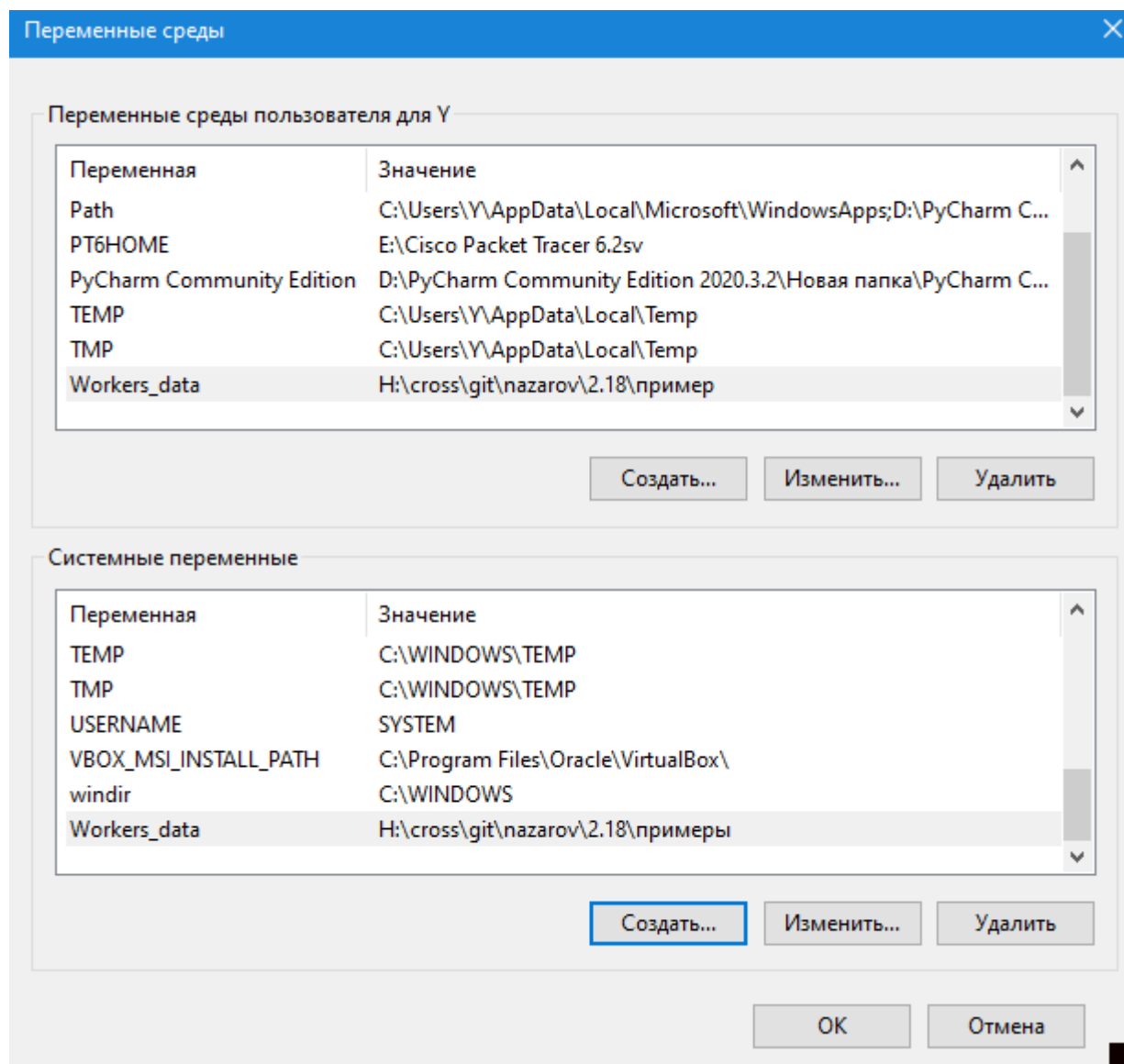


Рисунок 4 – созданная переменная среды

```
c:\Users\Admin\Desktop\git\Python13-2.18\Primers>python primer.py display
+-----+-----+-----+-----+
| No | Ф.И.О. | Должность | Год |
+-----+-----+-----+-----+
| 1 | Баранов ВИ | Сварщик | 2012 |
| 2 | Краманюк ДИ | Старший Сварщик | 2014 |
| 3 | Сидоров Сидор | Главный инженер | 2012 |
+-----+-----+-----+-----+

c:\Users\Admin\Desktop\git\Python13-2.18\Primers>python primer.py add --name="Меладзе Бармут" --post="Главный слесарь" --year=2011

c:\Users\Admin\Desktop\git\Python13-2.18\Primers>python primer.py display
+-----+-----+-----+-----+
| No | Ф.И.О. | Должность | Год |
+-----+-----+-----+-----+
| 1 | Баранов ВИ | Сварщик | 2012 |
| 2 | Краманюк ДИ | Старший Сварщик | 2014 |
| 3 | Сидоров Сидор | Главный инженер | 2012 |
| 4 | Меладзе Бармут | Главный слесарь | 2011 |
+-----+-----+-----+-----+

c:\Users\Admin\Desktop\git\Python13-2.18\Primers>
```

Рисунок 5 – результат выполнения 1 примера

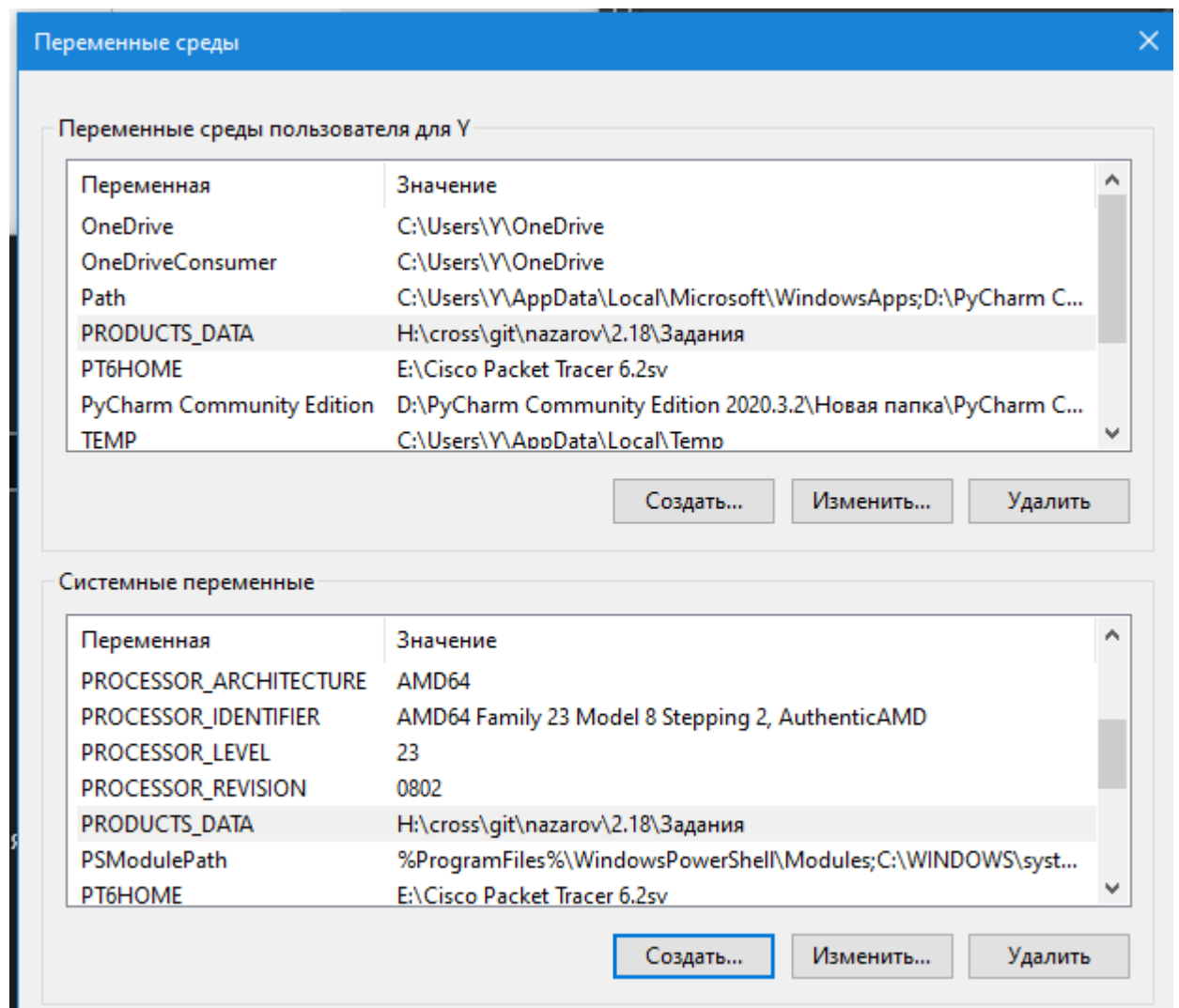


Рисунок 6 – создал переменную среды для выполнения индивидуального задания

```
# Получить имя файла
data_file = args.data
if not data_file:
    data_file = os.environ.get("PRODUCTS_DATA")
if not data_file:
    print("The data file name is absent", file=sys.stderr)
    sys.exit(1)
```

Рисунок 7 – дополнил код из работы 2.17 для выполнения 1 индивидуального задания

```

# Получить имя файла
data_file = args.data
dotenv_path = os.path.join(os.path.dirname(__file__), ".env")
if os.path.exists(dotenv_path):
    load_dotenv(dotenv_path)
if not data_file:
    data_file = os.getenv("PRODUCTS_DATA")
if not data_file:
    print("The data file name is absent", file=sys.stderr)
    sys.exit(1)

```

Рисунок 8 – дополнил код из индивидуального задания 1

Вывод: в результате выполнения лабораторной работы были получены теоретические сведения и практические навыки для работы с переменными окружениями с помощью языка программирования Python версии 3.x.

Ответы на контрольные вопросы:

1. Каково назначение переменных окружения?

Переменные окружения используются для передачи информации процессам, которые запущены в оболочке.

2. Какая информация может храниться в переменных окружения?

Переменные среды хранят информацию о среде операционной системы. Эта информация включает такие сведения, как путь к операционной системе, количество процессоров, используемых операционной системой, и расположение временных папок.

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

Нужно открыть окно свойства системы и нажать на кнопку “Переменные среды”.

4. Каково назначение переменных PATH и PATHNEXT?

PATH позволяет запускать исполняемые файлы и скрипты, «лежащие» в определенных каталогах, без указания их точного местоположения.

PATHNEXT дает возможность не указывать даже расширение файла, если оно прописано в ее значениях.

5. Как создать или изменить переменную окружения в Windows?

В окне “Переменные среды” нужно нажать на кнопку “Создать”, затем ввести имя переменной и путь.

6. Что представляют собой переменные окружения в ОС Linux?

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

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 текущему экземпляру оболочки. Каждая оболочка, например, `bash` или `zsh`, имеет свой собственный набор внутренних переменных.

8. Как вывести значение переменной окружения в Linux?

Наиболее часто используемая команда для вывода переменных окружения – `printenv`.

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. 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.

10. Какие переменные оболочки Linux Вам известны?

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.

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?

Чтобы создать новую переменную оболочки с именем, нужно ввести имя этой переменной потом знак равенства и указать значение новой переменной

12. Как установить переменные окружения в Linux?

Команда `export` используется для задания переменных окружения.

С помощью данной команды мы экспортируем указанную переменную, в результате чего она будет видна во всех вновь запускаемых дочерних командных оболочках.

13. Для чего необходимо делать переменные окружения Linux постоянными?

Чтобы переменная сохранялась после закрытия сеанса оболочки.

14. Для чего используется переменная окружения PYTHONHOME?

Переменная среды `PYTHONHOME` изменяет расположение стандартных библиотек Python.

15. Для чего используется переменная окружения PYTHONPATH?

Переменная среды `PYTHONPATH` изменяет путь поиска по умолчанию для файлов модуля.

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

`PYTHONSTARTUP` `PYTHONOPTIMIZE` `PYTHONBREAKPOINT`
`PYTHONDEBUG` `PYTHONINSPECT` `PYTHONUNBUFFERED`
`PYTHONVERBOSE` `PYTHONCASEOK` `PYTHON-`
`DONTWRITEBYTECODE`
`PYTHONPYCACHEPREFIX` `PYTHONHASHSEED` `PYTHONIOENCODING`
`PYTHONNOUSERSITE` `PYTHONUSERBASE`
`PYTHONWARNINGS` `PYTHONFAULTHANDLER`

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

Путём использования модуля `os`, при помощи которого программист может получить и изменить значения всех переменных среды.

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

При помощи модуля `os` можно просмотреть все переменные окружения, у которых есть значение.

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

Для присвоения значения любой переменной среды используется функция `setdefault()`.