МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Северо-Кавказский федеральный университет»

Отчет по лабораторной работе №18

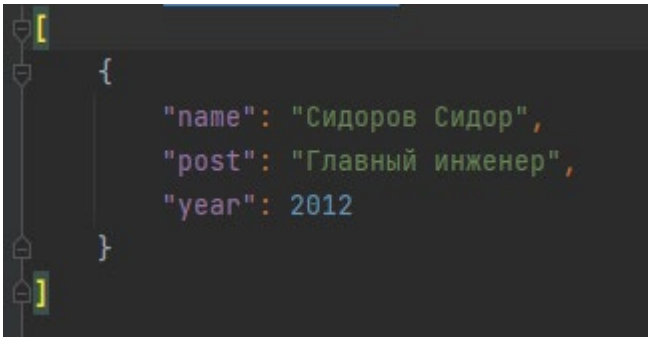
Работа с переменными окружения в Python3
по дисциплине «Основы программной инженерии»

Выполнил студент группы ПИЖ-б-о-20-1
Симоненко А.С. « » 2022г.
Подпись студента_____
Работа защищена « »_____2022г.
Проверил Воронкин Р.А. _____
(подпись)

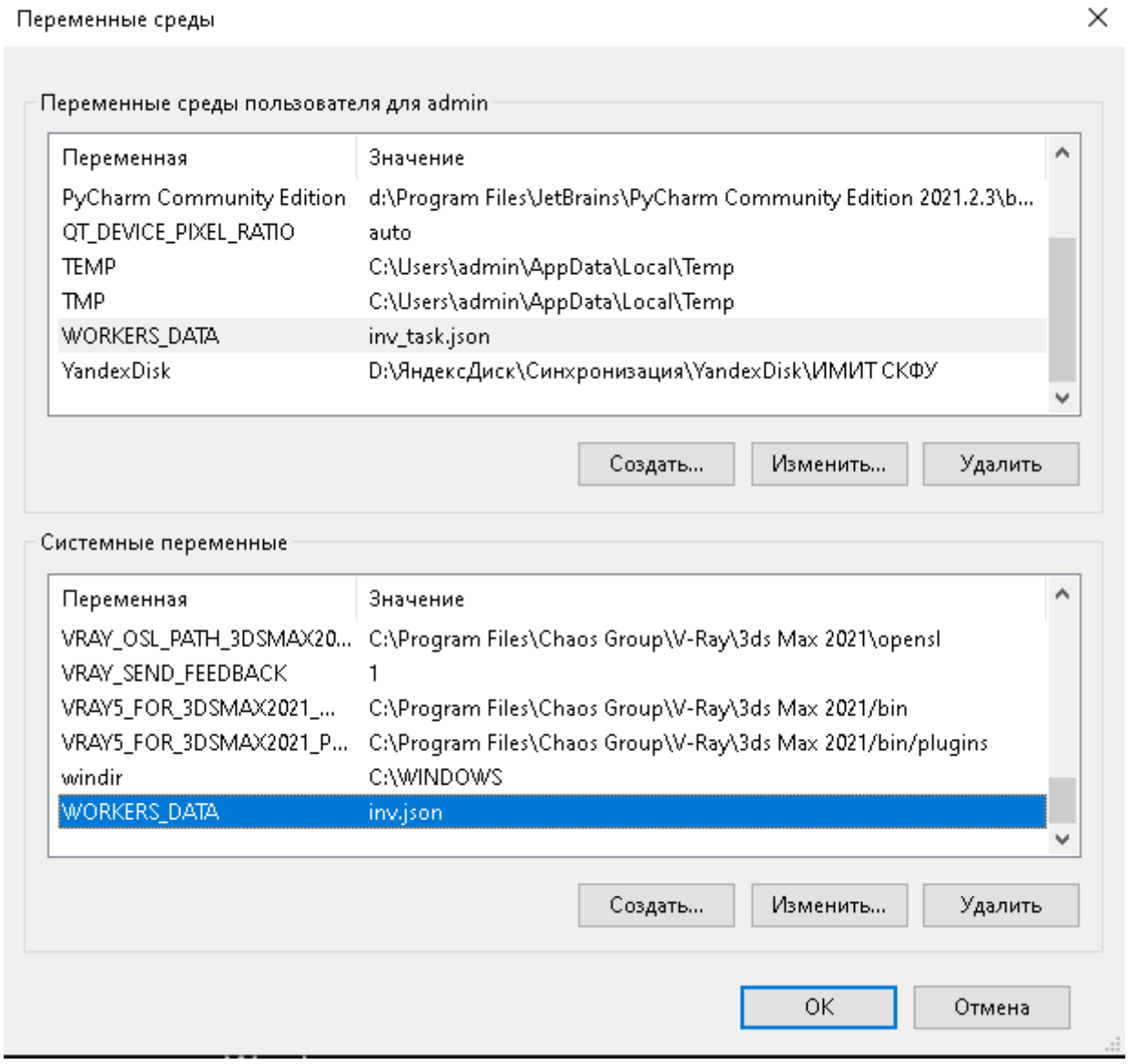
Ставрополь, 2022

Пример 1

```
C:\Users\admin\Desktop\git\LAB_18\examples>python example_1.py add --name="Сидоров
 Сидор" --post="Главный инженер" --year=2012
```



```
[
    {
        "name": "Сидоров Сидор",
        "post": "Главный инженер",
        "year": 2012
    }
]
```



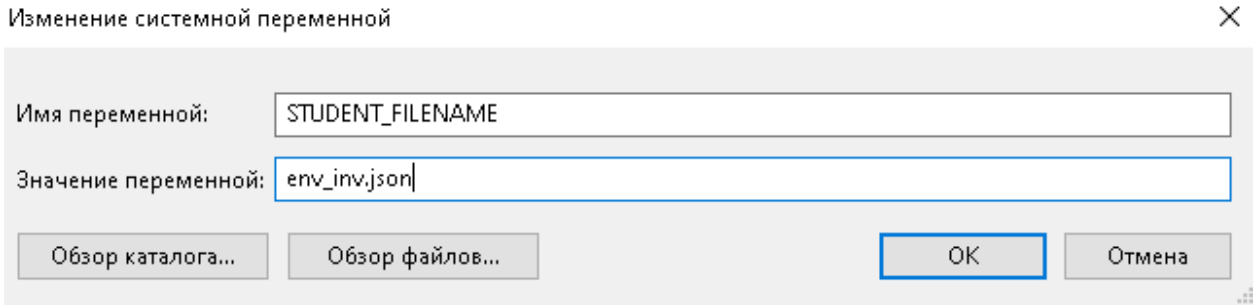
```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import argparse
import json
import os
import sys
from datetime import date


def add_worker(staff, name, post, year):
    """
    Добавить данные о работнике.
    """
    staff.append(
        {
            "name": name,
            "post": post,
            "year": year
        }
    )

    return staff


def display_workers(staff):
    """
    Отобразить список работников.
    """
    # Проверить, что список работников не пуст.
    if staff:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )

        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)
        # Вывести данные о всех сотрудниках.
        for idx, worker in enumerate(staff, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    worker.get('name', ''),
                    worker.get('post', ''),
                    worker.get('year', 0)
                )
            )
        print(line)

    else:
        print("Список работников пуст.")
```

```python
def select_workers(staff, period):
    """
    Выбрать работников с заданным стажем.
    """
    # Получить текущую дату.
    today = date.today()

    # Сформировать список работников.
    result = []

    for employee in staff:
        if today.year - employee.get('year', today.year) >= period:
            result.append(employee)

    # Возвратить список выбранных работников.
    return result


def save_workers(file_name, staff):
    """
    Сохранить всех работников в файл JSON.
    """
    # Открыть файл с заданным именем для записи.
    with open(file_name, "w", encoding="utf-8") as fout:
        # Выполнить сериализацию данных в формат JSON.
        # Для поддержки кирилицы установим ensure_ascii=False
        json.dump(staff, fout, ensure_ascii=False, indent=4)


def load_workers(file_name):
    """
    Загрузить всех работников из файла JSON.
    """
    # Открыть файл с заданным именем для чтения.
    with open(file_name, "r", encoding="utf-8") as fin:
        return json.load(fin)


def main(command_line=None):
    # Создать родительский парсер для определения имени файла.
    file_parser = argparse.ArgumentParser(add_help=False)
    file_parser.add_argument(
        "-d",
        "--data",
        action="store",
        required=False,
        help="The data file name"
    )

    # Создать основной парсер командной строки.
    parser = argparse.ArgumentParser("workers")
    parser.add_argument(
        "--version",
        action="version",
        version="%(prog)s 0.1.0"
    )

    subparsers = parser.add_subparsers(dest="command")

    # Создать субпарсер для добавления работника.
    add = subparsers.add_parser(
        "add",
        parents=[file_parser],
```

```
        help="Add a new worker"
    )

    add.add_argument(
        "-n",
        "--name",
        action="store",
        required=True,
        help="The worker's name"
    )

    add.add_argument(
        "-p",
        "--post",
        action="store",
        help="The worker's post"
    )

    add.add_argument(
        "-y",
        "--year",
        action="store",
        type=int,
        required=True,
        help="The year of hiring"
    )

    # Создать субпарсер для отображения всех работников.
    _ = subparsers.add_parser(
        "display",
        parents=[file_parser],
        help="Display all workers"
    )

    # Создать субпарсер для выбора работников.
    select = subparsers.add_parser(
        "select",
        parents=[file_parser],
        help="Select the workers"
    )

    select.add_argument(
        "-P",
        "--period",
        action="store",
        type=int,
        required=True,
        help="The required period"
    )

    # 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("WORKERS_DATA")

    if not data_file:
        print("The data file name is absent", file=sys.stderr)
        sys.exit(1)

    # Загрузить всех работников из файла, если файл существует.
    is_dirty = False
```

```
    if os.path.exists(data_file):
        workers = load_workers(data_file)
    else:
        workers = []

    # Добавить работника.
    if args.command == "add":
        workers = add_worker(
            workers,
            args.name,
            args.post,
            args.year
        )

        is_dirty = True

    # Отобразить всех работников.
    elif args.command == "display":
        display_workers(workers)

    # Выбрать требуемых рааботников.
    elif args.command == "select":
        selected = select_workers(workers, args.period)
        display_workers(selected)

    # Сохранить данные в файл, если список работников был изменен.
    if is_dirty:
        save_workers(data_file, workers)

if __name__ == "__main__":
    main()
```

Индивидуальные задания 1




```
C:\Users\admin\Desktop\git\LAB_18\Individual\lab-18\individuals\invidual_1>python inv_main.py add -n
nmae -g 1 -y 2022
[{'name': 'nmae', 'group': 1, 'z': 2022}]

C:\Users\admin\Desktop\git\LAB_18\Individual\lab-18\individuals\invidual_1>python inv_main.py list
+------+--------------------------------+----------------------+-----------------+
|  №   |            Ф.И.О.              |     Номер группы     |  Успеваемость   |
+------+--------------------------------+----------------------+-----------------+
|    1 | nmae                           | 0                    |            2022 |
+------+--------------------------------+----------------------+-----------------+
```

Индивидуальные задания 2

```
C:\Users\admin\Desktop\git\LAB_18\Individual\lab-18\individuals\invidual_2>pyth
on inv_main.py add -n env_name -g 2 -y 2023
[{'name': 'env_name', 'group': 2, 'z': 2023}]
```

```
C:\Users\admin\Desktop\git\LAB_18\Individual\lab-18\individuals\invidual_2>python inv_main.py list
+------+--------------------------------+---------------------+----------------+
|  №   |             Ф.И.О.             |     Номер группы    |  Успеваемость  |
+------+--------------------------------+---------------------+----------------+
|    1 | env_name                       | 0                   |           2023 |
+------+--------------------------------+---------------------+----------------+
```