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 №2.6
дисциплины «Основы программной инженерии»

Выполнил:
Баратов Семен Григорьевич
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

_____
(подпись)

Преподаватель:
Воронкин Р.А., канд. тех. наук, доцент,
доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: Работа со словарями в языке Python.

Цель: приобретение навыков по работе со словарями при написании программ с помощью языка программирования Python версии 3.x.

Результаты выполнения

1. Создали репозиторий с лицензией MIT, добавили в .gitignore необходимые правила для работы с IDE PyCharm, клонировали репозиторий, организовали репозиторий в соответствии с моделью git-flow.

```
Last login: Tue Oct 24 20:29:02 on ttys000
itssyoma@MacBook-Air-Sema Основы программной инженерии % git clone https://github.com/itssyoma/megarepo_21.git
Cloning into 'megarepo_21'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
itssyoma@MacBook-Air-Sema Основы программной инженерии % cd megarepo_21
itssyoma@MacBook-Air-Sema megarepo_21 % git checkout -b develop
Switched to a new branch 'develop'
itssyoma@MacBook-Air-Sema megarepo_21 % git branch release
itssyoma@MacBook-Air-Sema megarepo_21 % git branch develop
fatal: a branch named 'develop' already exists
itssyoma@MacBook-Air-Sema megarepo_21 % git branch hotfix
itssyoma@MacBook-Air-Sema megarepo_21 % git branch feature
itssyoma@MacBook-Air-Sema megarepo_21 %
```

Рисунок 1 – Работа с репозиторием в командной строке.

2. Проработали пример №1.

example.py > ...

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

import sys
from datetime import date


if __name__ == '__main__':
    # Список работников.
    workers = []

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о работнике.
            name = input("Фамилия и инициалы? ")
            post = input("Должность? ")
            year = int(input("Год поступления? "))

            # Создать словарь.
            worker = {
                'name': name,
                'post': post,
                'year': year,
            }

            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            # Заголовок таблицы.
            line = '+-{}-+-{}-+-{}-+-{}-+'.format(
                '-' * 4,
                '-' * 30,
                '-' * 20,
                '-' * 8
            )
            print(line)
            print(
                '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                    "No",
                    "Ф.И.О.",
                    "Должность",
```

Рисунок 2 – Пример №1. Код программы (часть 1).

```
            "Год"
        )
    )
    print(line)

    # Вывести данные о всех сотрудниках.
    for idx, worker in enumerate(workers, 1):
        print(
            '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                idx,
                worker.get('name', ''),
                worker.get('post', ''),
                worker.get('year', 0)
            )
        )

    print(line)

elif command.startswith('select '):
    # Получить текущую дату.
    today = date.today()

    # Разбить команду на части для выделения номера года.
    parts = command.split(' ', maxsplit=1)
    # Получить требуемый стаж.
    period = int(parts[1])

    # Инициализировать счетчик.
    count = 0
    # Проверить сведения работников из списка.
    for worker in workers:
        if today.year - worker.get('year', today.year) >= period:
            count += 1
            print(
                '{:>4}: {}'.format(count, worker.get('name', ''))
            )

    # Если счетчик равен 0, то работники не найдены.
    if count == 0:
        print("Работники с заданным стажем не найдены.")

elif command == 'help':
    # Вывести справку о работе с программой.
    print("Список команд:\n")
    print("add - добавить работника;")
    print("list - вывести список работников;")
    print("select <стаж> - запросить работников со стажем;")
    print("help - отобразить справку;")
    print("exit - завершить работу с программой.")

else:
    print(f"Неизвестная команда {command}", file=sys.stderr)
```

Рисунок 3 – Пример №1. Код программы (часть 2).

```
ПРОБЛЕМЫ   ВЫХОДНЫЕ ДАННЫЕ   КОНСОЛЬ ОТЛАДКИ   ТЕРМИНАЛ   ПОРТЫ

>>> add
Фамилия и инициалы? Баратов С.Г.
Должность? Дизайнер
Год поступления? 2019
>>> list
+-----+----------------------------------+----------------------+----------+
|  No |              Ф.И.О.              |      Должность       |   Год    |
+-----+----------------------------------+----------------------+----------+
|   1 | Баратов С.Г.                     | Дизайнер             |     2019 |
+-----+----------------------------------+----------------------+----------+
>>> select 1
   1: Баратов С.Г.
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
exit - завершить работу с программой.
>>> hi
Неизвестная команда hi
>>>
```

Рисунок 4 – Пример №1. Результат выполнения программы (часть 2).

3. Выполнили задание №1. Создали словарь, связав его с переменной school, и наполнили данными, которые отражают количество учащихся в разных классах (1а, 1б, 2б, 6а, 7в и т. п.). Внесли изменения в словарь согласно следующему: а) в одном из классов изменилось количество

учащихся, б) в школе появился новый класс, с) в школе был расформирован (удален) другой класс. Вычислили общее количество учащихся в школе.

Рисунок 5 – Задание №1. Код и результат выполнения программы.

4. Выполнили задание №2. Создали словарь, где ключами являются числа, а значениями – строки. Применили к нему метод items(), а с помощью полученного объекта dict_items создали новый словарь, "обратный" исходному, т. е. ключами являются строки, а значениями – числа.

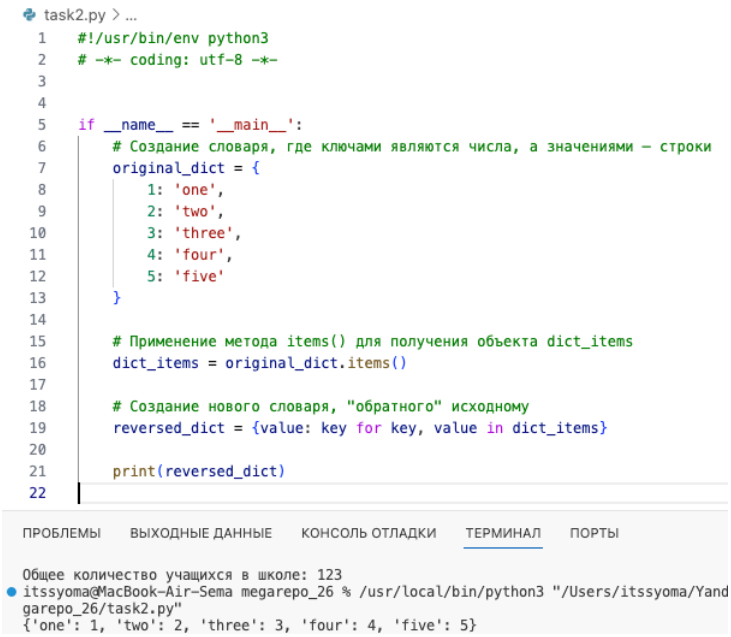

Рисунок 6 – Задание №2. Код и результат выполнения программы.

5. Выполнили индивидуальное задание (вариант 1). Использовать словарь, содержащий следующие ключи: фамилия и инициалы; номер группы; успеваемость (список из пяти элементов). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по возрастанию номера группы; вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4.0; если таких студентов нет, вывести соответствующее сообщение.

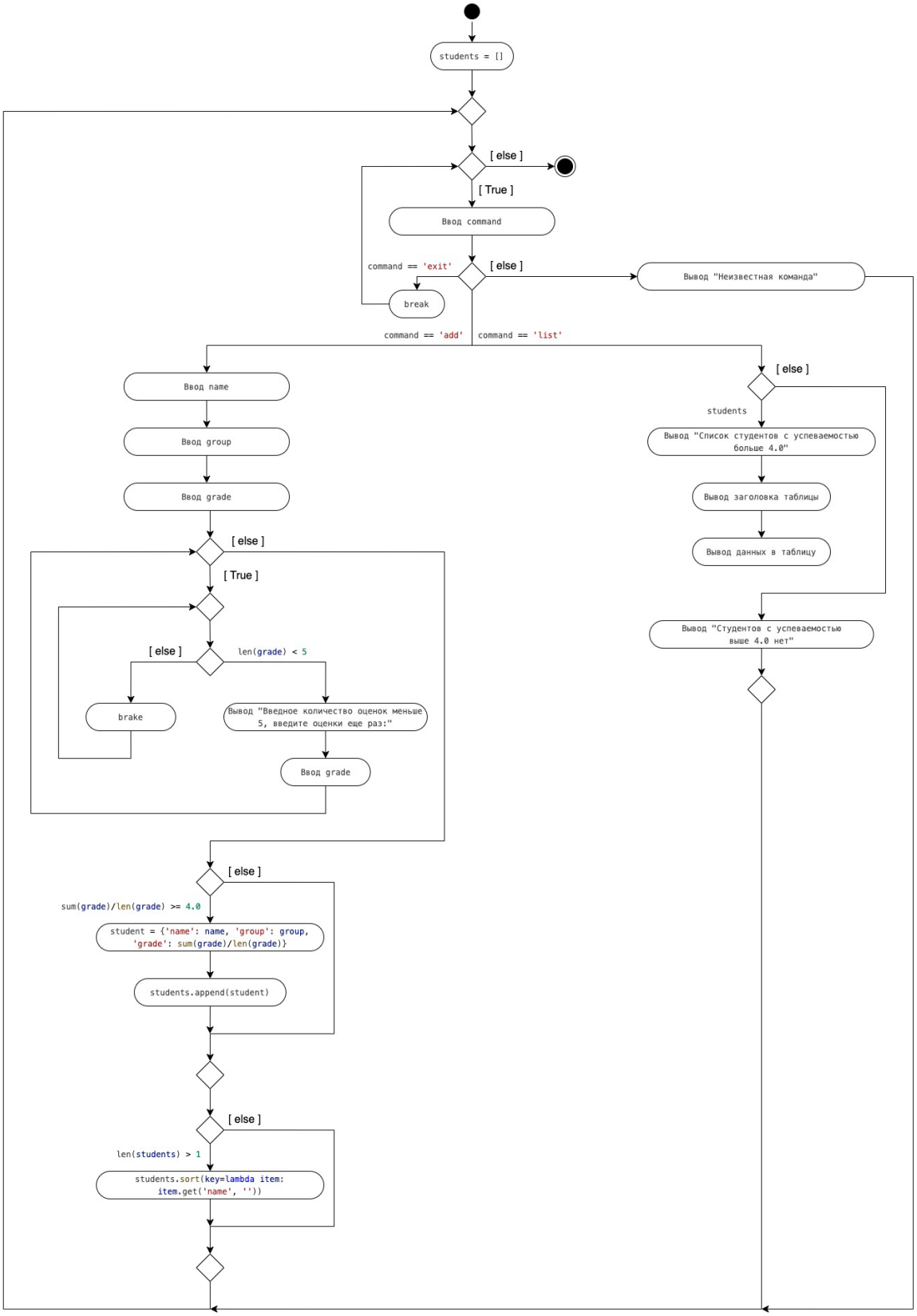

Рисунок 7 – Индивидуальное задание. UML-диаграмма деятельности.

individual.py > ...

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

import sys


if __name__ == '__main__':
    # Список студентов
    students = []

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о студенте.
            name = input("Фамилия и инициалы? ")
            group = input("Номер группы? ")
            grade = list(map(int, input("Успеваемость студента? (Пять оценок через пробел) ").split()))
            while True:
                if len(grade) < 5:
                    print("Введное количество оценок меньше 5, введите оценки еще раз: ", file=sys.stderr)
                    grade = list(map(int, input("Успеваемость студента? (Пять оценок через пробел) ").split()))
                else:
                    break

            # Создать словарь.
            if sum(grade)/len(grade) >= 4.0:
                student = {
                    'name': name,
                    'group': group,
                    'grade': sum(grade)/len(grade),
                }
                # Добавить словарь в список.
                students.append(student)
```

Рисунок 8 – Индивидуальное задание. Код программы (часть 1).

individual.py > ...

```python

            # Отсортировать список в случае необходимости.
            if len(students) > 1:
                students.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            if students:
                # Заголовок таблицы.
                line = '+-{}-+-{}-+-{}-+'.format(
                    '-' * 4,
                    '-' * 30,
                    '-' * 20
                )
                print(line)
                print(
                    '| {:^4} | {:^30} | {:^20} |'.format(
                        "No",
                        "Ф.И.О.",
                        "Группа"
                    )
                )
                print(line)

                # Вывести данные о всех сотрудниках.
                for idx, student in enumerate(students, 1):
                    print(
                        '| {:>4} | {:<30} | {:<20} |'.format(
                            idx,
                            student.get('name', ''),
                            student.get('group', '')
                        )
                    )

                print(line)

            else:
                print("Студентов с успеваемостью выше 4.0 нет")

        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)
```

Рисунок 9 – Индивидуальное задание. Код программы (часть 2).

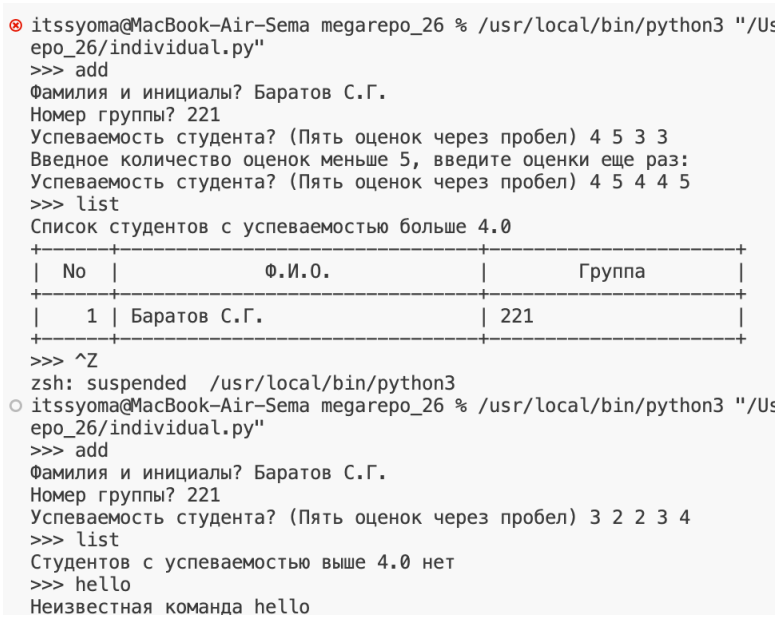

Рисунок 10 – Индивидуальное задание.

Результат выполнения программы.

Ответы на контрольные вопросы

1. Что такое словари в языке Python?

Словари в языке Python – это структура данных, которая хранит коллекцию элементов в формате ключ-значение.

2. Может ли функция len() быть использована при работе со словарями?

Да, функция len() может быть использована для определения количества элементов в словаре.

3. Какие методы обхода словарей Вам известны?

Методы обхода словарей включают использование циклов (например, for) для перебора ключей или значений, использование методов items(), keys() и values().

4. Какими способами можно получить значения из словаря по ключу?

Значения из словаря по ключу можно получить с помощью оператора [] или метода get().

5. Какими способами можно установить значение в словаре по ключу?

Значение в словаре по ключу можно установить с помощью оператора [] или метода update().

6. Что такое словарь включений?

Словарь включений – это способ создания словаря с помощью компактного синтаксиса, используя циклы и условные выражения.

7. Самостоятельно изучите возможности функции zip() приведите примеры ее использования.

Функция zip() используется для объединения элементов нескольких последовательностей в одну последовательность кортежей.

```
list1 = [1, 2, 3]
list2 = ['a', 'b', 'c']
zipped = zip(list1, list2)
print(list(zipped))  # [(1, 'a'), (2, 'b'), (3, 'c')]
```

8. Самостоятельно изучите возможности модуля datetime. Каким функционалом по работе с датой и временем обладает этот модуль?

Модуль datetime предоставляет функционал для работы с датой и временем, включая создание объектов даты, времени, интервала времени, форматирование и парсинг даты и времени, арифметические операции с датами и временем и многое другое.