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

Отчет по лабораторной работе №9

Работа со словарями в языке Python

по дисциплине «Основы программной инженерии»

Выполнил студент группы ПИЖ-б-о-20-1
Симоненко А.С. « » 2021г.
Подпись студента_____
Работа защищена « »_____2021г.
Проверил Воронкин Р.А. _____
(подпись)

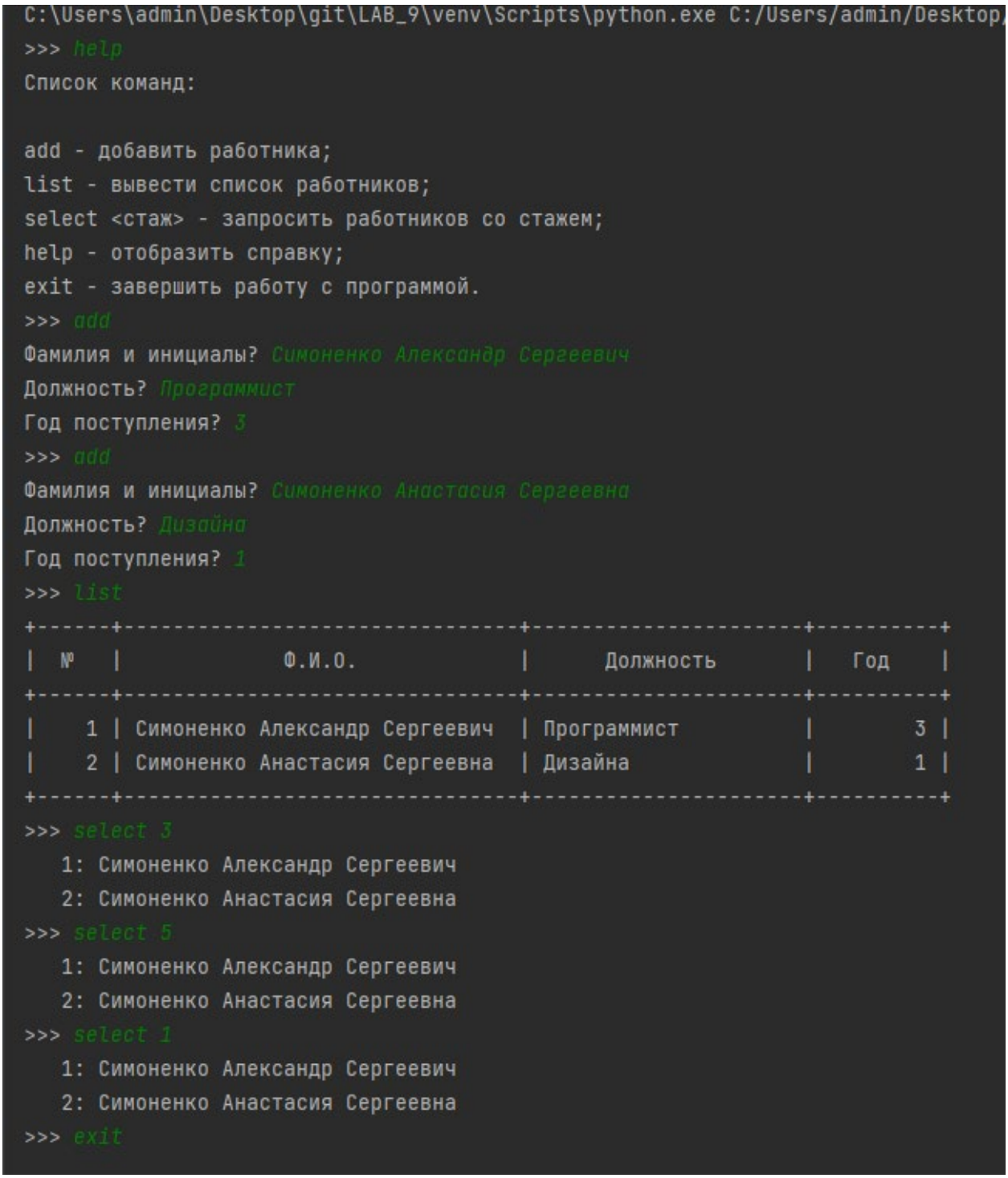
Ставрополь 2021

Пример 1. Использовать словарь, содержащий следующие ключи: фамилия и инициалы работника; название занимаемой должности; год поступления на работу

```
#!/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(
                    "№",
                    "Ф.И.О.",
                    "Должность",
                    "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)
```



Решите задачу: создайте словарь, связав его с переменной school , и наполните данными, которые бы отражали количество учащихся в разных классах (1а, 1б, 2б, 6а, 7в и т. п.). Внесите 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. Вычислите общее количество учащихся в школе.

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

if __name__ == '__main__':
    school = {'1a': 18, '1б': 15, '2а': 25, '2б': 31, '3а': 22, '3б': 19, '4а': 24, "4б": 27}
    while True:
        n = input('Введите название операции >>> ')
        if n == 'change':
            school.update({input(f'Название изменяемого класса: '):
                           int(input(f'Количество учеников изменяемого класса: '))})
        elif n == 'new':
            school.update({input(f'Название класса №: '): int(input(f'Количество учеников класса №: '))})
        elif n == 'remove':
            del school[input(f'Название расформировываемого класса: ')]
        elif n == 'print':
            print(school)
        elif n == 'sum':
            print(sum(school.values()))
        elif n == "help":
          print('change - Изменилось количество учеников:')
          print('new - В школе появился новый класс')
          print('remove - В школе был расформирован (удален) класс')
          print('print - Выгрузка данных')
          print('sum - Число учеников')
          print('exit - Выход')
        elif n == 'exit':
            break
```

```
C:\Users\admin\Desktop\git\LAB_9\venv\Scripts\python.exe C:/Users/admin/Desktop/git/
Введите название операции >>> help
change - Изменилось количество учеников:
new - В школе появился новый класс
remove - В школе был расформирован (удален) класс
print - Выгрузка данных
sum - Число учеников
exit - Выход
Введите название операции >>> print
{'1а': 18, '1б': 15, '2а': 25, '2б': 31, '3а': 22, '3б': 19, '4а': 24, '4б': 27}
Введите название операции >>> change
Название изменяемого класса: 1а
Количество учеников изменяемого класса: 23
Введите название операции >>> print
{'1а': 23, '1б': 15, '2а': 25, '2б': 31, '3а': 22, '3б': 19, '4а': 24, '4б': 27}
Введите название операции >>> remove
Название расформировываемого класса: 1а
Введите название операции >>> print
{'1б': 15, '2а': 25, '2б': 31, '3а': 22, '3б': 19, '4а': 24, '4б': 27}
Введите название операции >>> new
Название класса №: 11а
Количество учеников класса №: 31
Введите название операции >>> print
{'1б': 15, '2а': 25, '2б': 31, '3а': 22, '3б': 19, '4а': 24, '4б': 27, '11а': 31}
Введите название операции >>> sum
194
Введите название операции >>> exit

Process finished with exit code 0
```

Решите задачу: создайте словарь, где ключами являются числа, а значениями – строки. Примените к нему метод items(), с с помощью полученного объекта dict_items создайте новый словарь, "обратный" исходному, т. е. ключами являются строки, а значениями – числа.

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

def rev_key(dct):
    dct_new = dict()
    for i, v in dct.items():
        for w in v:
            dct_new[w] = dct_new.get(w, []) + [i]
    return dct_new


dct = {1: 'acc', 2: 'cab', 3: 'ccb'}
print(rev_key(dct))
```

```
C:\Users\admin\Desktop\git\LAB_9\venv\Scripts\python.exe "C
{'a': [1, 2], 'c': [1, 1, 2, 3, 3], 'b': [2, 3]}
```

Индивидуальные задания

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.

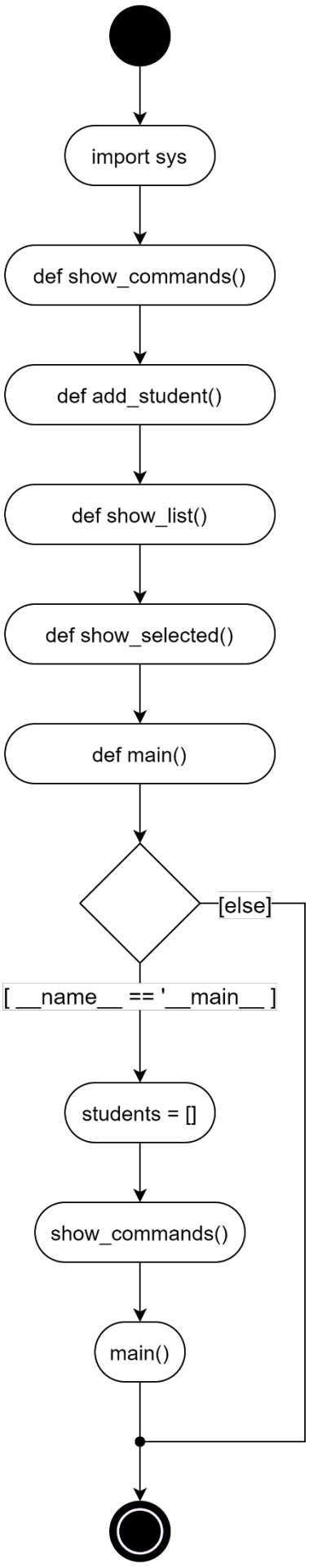
```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

def show_commands():
    print("Список команд:\n")
    print("add - добавить студента;")
    print("list - вывести список студентов;")
    print("select <средний балл> - запросить студентов с баллом выше 4.0;")
    print("exit - завершить работу с программой.")


def add_student():
    # Запросить данные о студенте.
    name = input("Фамилия и инициалы? ")
    group = input("Номер группы? ")
    grade = str(input('Успеваемость: '))
    # Создать словарь.
    student = {
        'name': name,
        'group': group,
        'grade': grade,
    }
    # Добавить словарь в список.
    students.append(student)
    # Отсортировать список в случае необходимости.
    if len(students) > 1:
        students.sort(key=lambda item: item.get('group')[::-1])
```

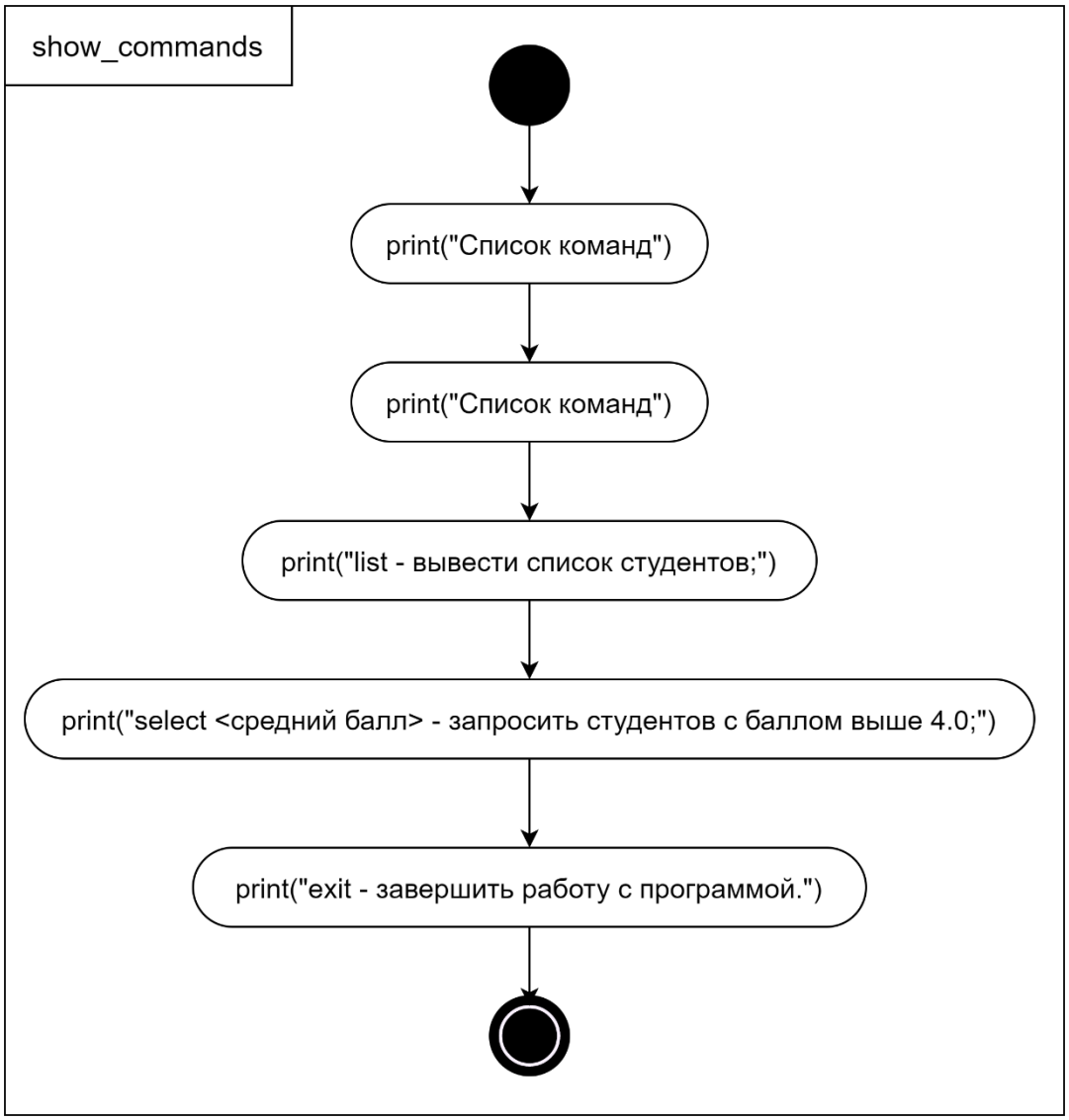
```python
def show_list():
    # Заголовок таблицы.
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
        '-' * 4,
        '-' * 30,
        '-' * 20,
        '-' * 15
    )
    print(line)
    print(
        '| {:^4} | {:^30} | {:^20} | {:^15} |'.format(
            "№",
            "Ф.И.О.",
            "Группа",
            "Успеваемость"
        )
    )
    print(line)
    # Вывести данные о всех студентах.
    for idx, student in enumerate(students, 1):
        print(
            '| {:>4} | {:<30} | {:<20} | {:>15} |'.format(
                idx,
                student.get('name', ''),
                student.get('group', ''),
                student.get('grade', 0)
            )
        )
    print(line)
```

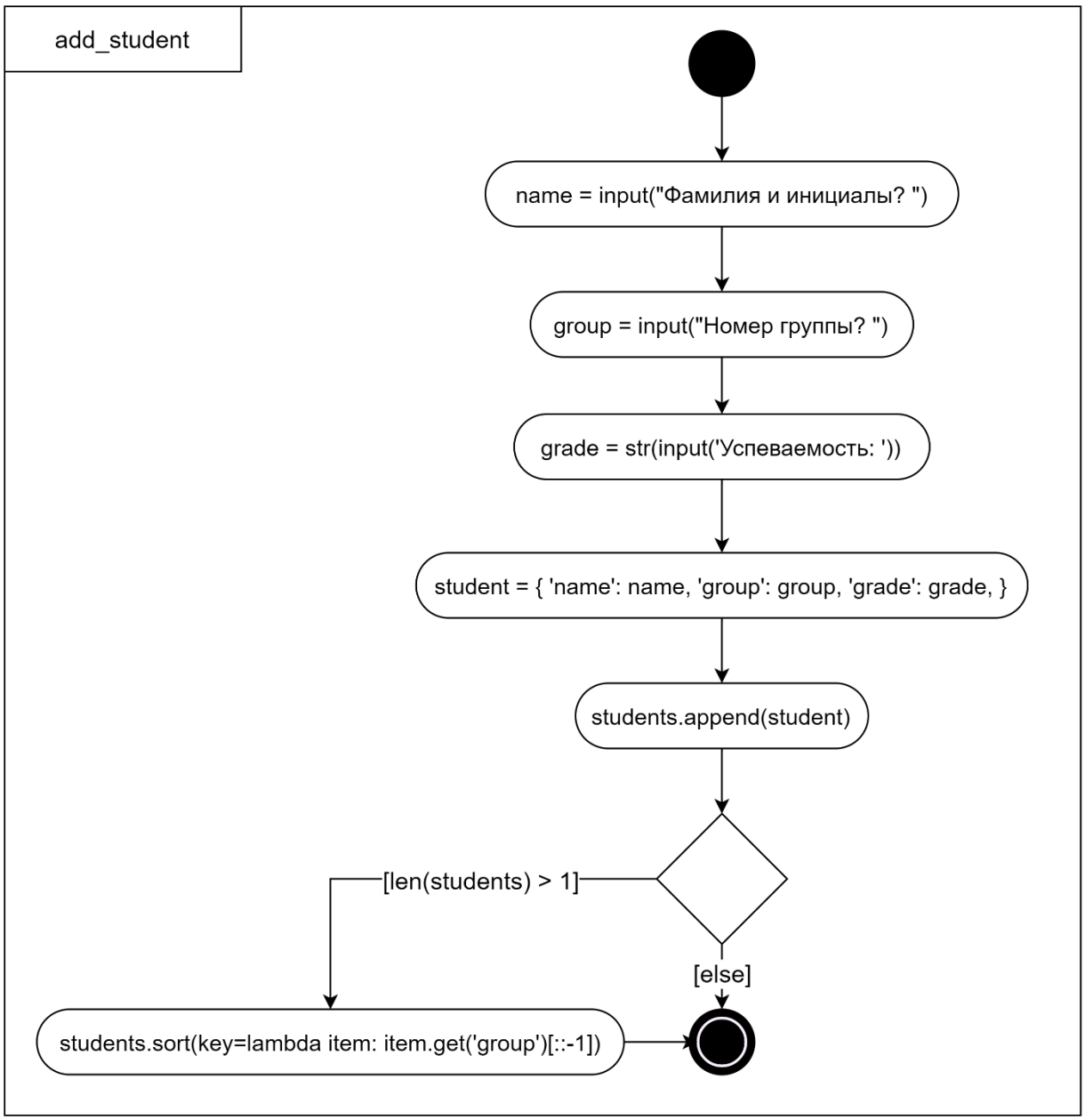
```python
def show_selected():
    # Инициализировать счетчик.
    count = 0
    # Проверить сведения студентов из списка.
    for student in students:
        grade = list(map(int, student.get('grade', '').split()))
        if sum(grade) / max(len(grade), 1) >= 4.0:
            print(
                '{:>4} {}'.format('*', student.get('name', '')),
                '{:>1} {}'.format('группа №', student.get('group', ''))
            )
            count += 1
    if count == 0:
        print("Студенты с баллом 4.0 и выше не найдены.")


def main():
    # 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':
            add_student()

        elif command == 'list':
            show_list()
```

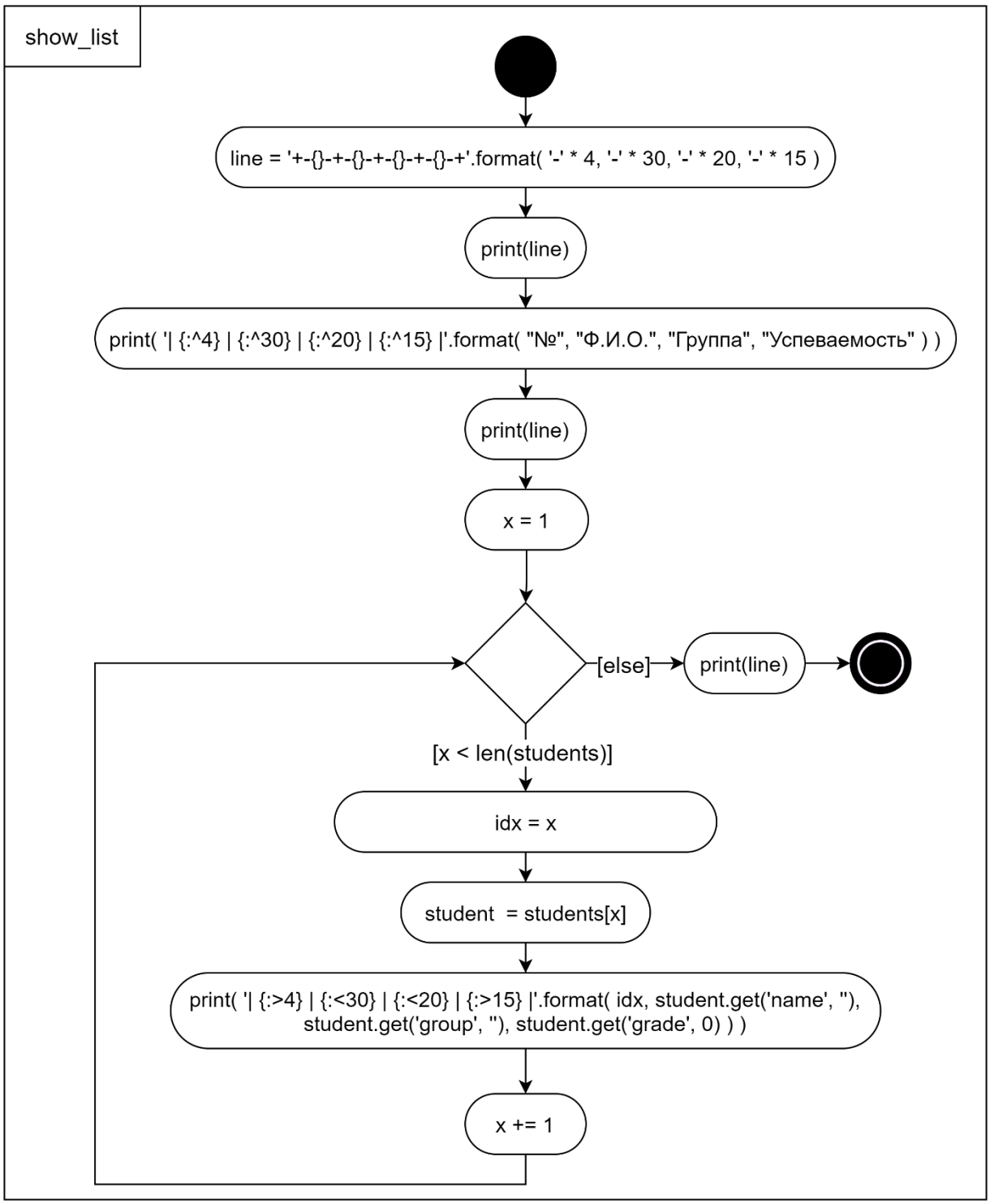
```
        elif command.startswith('select'):
            show_selected()

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


if __name__ == '__main__':
    # Список студентов.
    students = []
    show_commands()

    main()
```

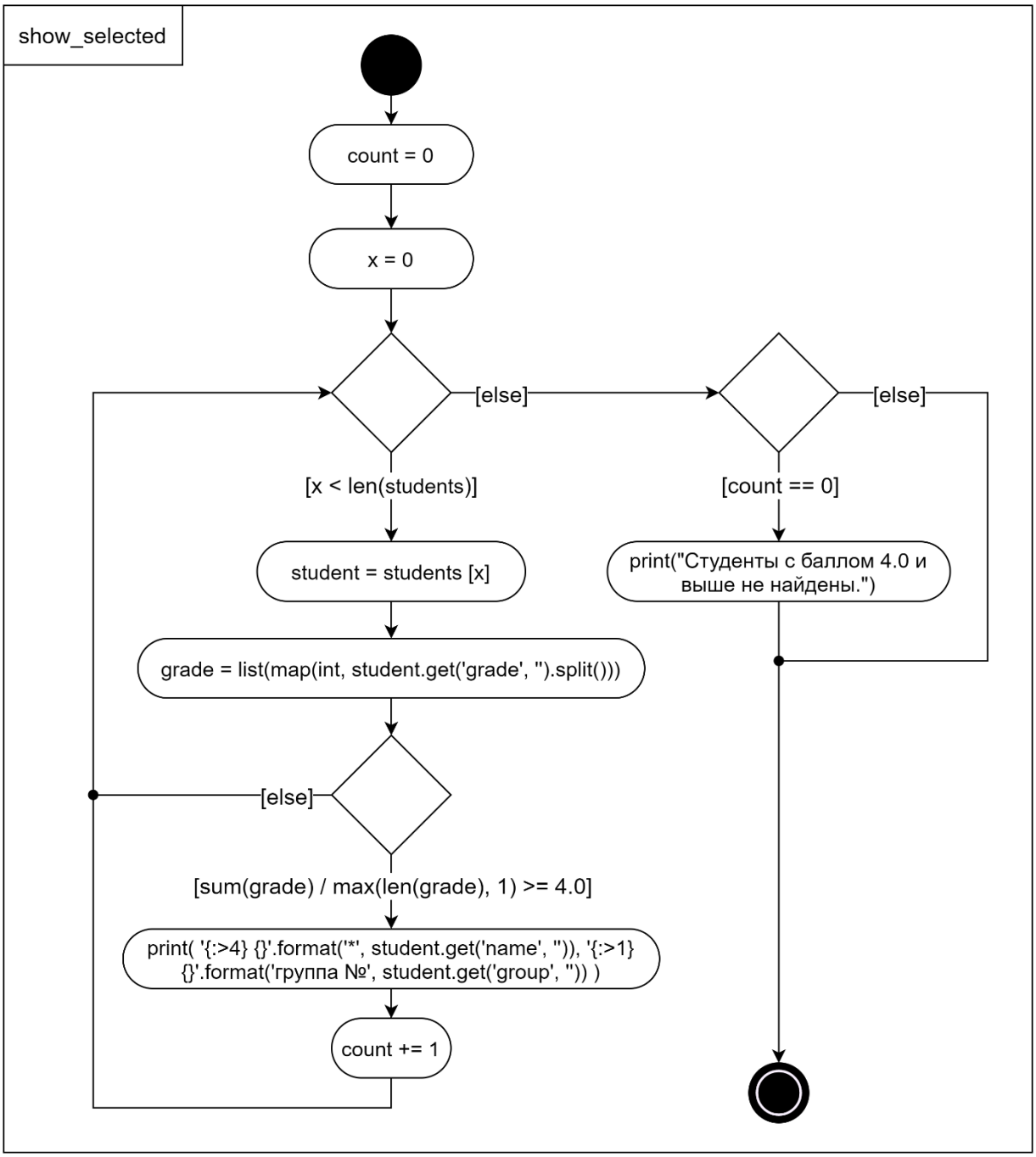
```
C:\Users\admin\Desktop\git\LAB_9\venv\Scripts\python.exe "C:/Users/admin/Desktop/git
Список команд:

add - добавить студента;
list - вывести список студентов;
select <средний балл> - запросить студентов с баллом выше 4.0;
exit - завершить работу с программой.
>>> add
Фамилия и инициалы? Симоненко Александр
Номер группы? 4
Успеваемость: 5
>>> add
Фамилия и инициалы? Буданов Иван
Номер группы? 1
Успеваемость: 9
>>> list
+------+--------------------------------+----------------------+-----------------+
|  №   |             Ф.И.О.             |        Группа        |  Успеваемость   |
+------+--------------------------------+----------------------+-----------------+
|    1 | Буданов Иван                   | 1                    |               9 |
|    2 | Симоненко Александр            | 4                    |               5 |
+------+--------------------------------+----------------------+-----------------+
>>> select 2
   * Буданов Иван группа № 1
   * Симоненко Александр группа № 4
>>> exit
```

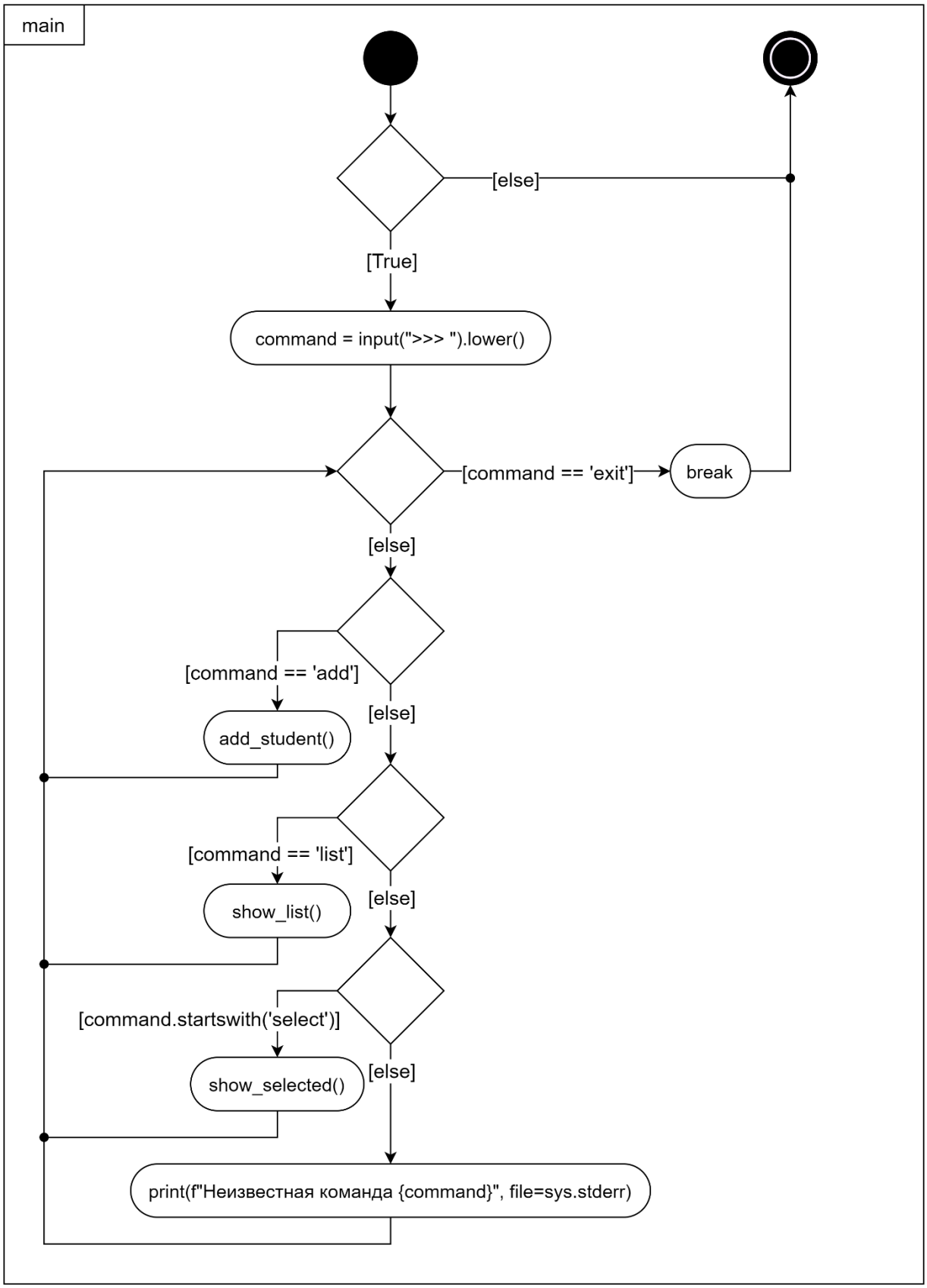

import sys
def show_commands()
def add_student()
def show_list()
def show_selected()
def main()
[else]
[__name__ == '__main__]
students = []
show_commands()
main()


show_commands
print("Список команд")
print("Список команд")
print("list - вывести список студентов;")
print("select <средний балл> - запросить студентов с баллом выше 4.0;")
print("exit - завершить работу с программой.")


add_student
name = input("Фамилия и инициалы? ")
group = input("Номер группы? ")
grade = str(input('Успеваемость: '))
student = { 'name': name, 'group': group, 'grade': grade, }
students.append(student)
[len(students) > 1]
[else]
students.sort(key=lambda item: item.get('group')[::-1])


show_list
line = '+-{}-+-{}-+-{}-+-{}-+'.format('-' * 4, '-' * 30, '-' * 20, '-' * 15)
print(line)
print('| {:^4} | {:^30} | {:^20} | {:^15} |'.format("№", "Ф.И.О.", "Группа", "Успеваемость"))
print(line)
x = 1
[else]
print(line)
[x < len(students)]
idx = x
student = students[x]
print('| {:>4} | {:<30} | {:<20} | {:>15} |'.format(idx, student.get('name', ''),
student.get('group', ''), student.get('grade', 0)))
x += 1


show_selected
count = 0
x = 0
[else]
[else]
[x < len(students)]
[count == 0]
student = students [x]
print("Студенты с баллом 4.0 и выше не найдены.")
grade = list(map(int, student.get('grade', '').split()))
[else]
[sum(grade) / max(len(grade), 1) >= 4.0]
print('{:>4} {}'.format('*', student.get('name', '')), '{:>1} {}'.format('группа №', student.get('group', '')))
count += 1


main
[else]
[True]
command = input(">>> ").lower()
[command == 'exit']
break
[else]
[command == 'add']
add_student()
[else]
[command == 'list']
show_list()
[else]
[command.startswith('select')]
show_selected()
[else]
print(f"Неизвестная команда {command}", file=sys.stderr)