**Федеральное государственное образовательное**

**бюджетное учреждение высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ**

**ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**ИНСТИТУТ ЗАОЧНОГО И ОТКРЫТОГО ОБРАЗОВАНИЯ**

**ДЕПАРТАМЕНТ АНАЛИЗА ДАННЫХ,**

**ПРИНЯТИЯ РЕШЕНИЙ И ФИНАНСОВЫХ ТЕХНОЛОГИЙ**

**ПРАКТИЧЕСКАЯ РАБОТА № 4**

**по дисциплине «Алгоритмы и структуры данных в языке Python»**

**Тема: словари**

Направление подготовки - 09.03.03 «Прикладная информатика»

Студент(ка) **Белоусов А.А.**

(Ф.И.О.)

Курс1 № группы **ЗБ-ПИ1-1**

Преподаватель **Петросов Д.А.**

(Ф.И.О.)

Москва 2020

Задания

**Задание 1**

Создать словарь адресной книги, содержащей ФИО и адрес. Заполнить его 50 элементами, реализовать поиск по адресу.

**Задание 2**

Создать словарь телефонного справочника. Заполнить его 50 элементами, реализовать поиск по телефону.

**Задание 3**

Реализовать проверку на существование записи в предыдущих заданиях с возможностью дополнения.

**Задание 4**

Создать словарь на свободную тему, включающий в себя кортеж в качестве ключа. Реализовать поиск по данному ключу.

Код программы

**import** random  
  
  
**def** getNames(count):  
 maleNames = frozenset((**'Виталий'**, **'Евгений'**, **'Пётр'**, **'Владимир'**))  
 maleSurnames = frozenset((**'Жуков'**, **'Смолов'**, **'Матвеев'**, **'Елиссеев'**, **'Петров'**))  
 malePatronymics = frozenset((**'Анатольевич'**, **'Владимирович'**, **'Петрович'**, **'Геннадьевич'**))  
  
 femaleNames = frozenset((**'Евгения'**, **'Варвара'**, **'Елизавета'**, **'Екатерина'**))  
 femaleSurnames = frozenset((**'Петрова'**, **'Скворцова'**, **'Князева'**, **'Голубикина'**))  
 femalePatronymics = frozenset((**'Петровна'**, **'Владимировна'**, **'Павловна'**, **'Витальевна'**))  
  
 count = min(count, len(maleNames) \* len(maleSurnames) \* len(malePatronymics) + len(femaleNames) \* len(  
 femaleSurnames) \* len(femalePatronymics))  
  
 males = [**' '**.join([name, surname, patronymic]) **for** name **in** maleNames **for** surname **in** maleSurnames **for** patronymic **in** malePatronymics]  
  
 females = [**' '**.join([name, surname, patronymic]) **for** name **in** femaleNames **for** surname **in** femaleSurnames **for** patronymic **in** femalePatronymics]  
  
 result = males + females  
 random.shuffle(result)  
 **return** result[:count]  
  
  
**def** getАddresses(count):  
 streets = frozenset((**"ул. Автомобилистов"**, **"пр. Петрова"**, **"ул. Московская"**, **"пл. Садовая"**, **"ул. Философская"**,  
 **"ул. Питоническая"**, **"ул. Словарная"**))  
  
 count = min(count, len(streets) \* 20 \* 20)  
  
 result = [**', '**.join([street, **" д. "** + str(house), **" кв. "** + str(flat)]) **for** street **in** streets **for** house **in** range(1, 21) **for** flat **in** range(1, 21)]  
 random.shuffle(result)  
 **return** result[:count]  
  
  
**def** getTelNumbers(count):  
 regioncodes = frozenset((**"499"**, **"812"**, **"424"**, **"845"**, **"485"**))  
  
 count = min(count, len(regioncodes) \* ((10 \*\* 7) - 1))  
  
 local\_numbers = list(  
 map(**lambda** number: str(number).rjust(7, **'0'**), random.sample(range(1, 10 \*\* 7), min(count, (10 \*\* 7) - 1))))  
 result = [**' '**.join([**"+7"**, code, **'-'**.join([local[:3], local[3:5], local[5:]])]) **for** code **in** regioncodes **for** local **in** local\_numbers]  
 random.shuffle(result)  
 **return** result[:count]  
  
  
**def** addItem(dict, key, value):**if** key **in** dict.keys():  
 print(**"Запись уже существует"**)  
 **else**:  
 dict.setdefault(key, value)  
 print(**"Запись: "** + key + **" = "** + value + **" добавлена"**)  
  
  
**def** main():  
 print(**"Задание №1"**)  
 addressBook = dict(zip(getNames(50), getАddresses(50)))  
 print(list(addressBook.items())[24])  
 print(addressBook.get(list(addressBook.keys())[24], **"Запись не найдена"**))  
 print(addressBook.get(**"Вероника Гагарина Витальевна"**, **"Запись не найдена"**))  
 print()  
  
 print(**"Задание №2"**)  
 phoneBook = dict(zip(getNames(100), getTelNumbers(100)))  
 print(list(phoneBook.items())[70])  
 print(phoneBook.get(list(phoneBook.keys())[70], **"Запись не найдена"**))  
 print(phoneBook.get(**"Павел Василий Васильевич"**, **"Запись не найдена"**))  
 print()  
  
 print(**"Задание №3"**)  
 addItem(addressBook, list(addressBook.keys())[18], **"ул. Множеств, д. 7, кв. 17"**)  
 addItem(addressBook, **"Вероника Гагарина Витальевна"**, **"ул. Множеств, д. 7, кв. 17"**)  
 print(addressBook.get(**"Вероника Гагарина Витальевна"**, **"Запись не найдена"**))  
 print()  
 addItem(phoneBook, list(phoneBook.keys())[15], **"+7 999 999-99-99"**)  
 addItem(phoneBook, **"Павел Василий Васильевич"**, **"+7 999 999-99-99"**)  
 print(phoneBook.get(**"Павел Василий Васильевич"**, **"Запись не найдена"**))  
 print()  
  
 print(**"Задание №4"**)  
 processors = {(**"Ice Lake"**, **"1068G7"**): (4, 8), (**"Ice Lake"**, **"1005G1"**): (2, 4), (**"Zen"**, **"Ryzen 5 1600X"**): (6, 12),  
 (**"Zen"**, **"Ryzen 7 1700"**): (8, 16), (**"Kaby Lake"**, **"7640X"**): (4, 4)}  
  
 print(getCPUsbyFamily(processors, **"Ice Lake"**))  
 print(getCPUsbyFamily(processors, **"Zen 2"**))  
 print()  
 print(getCPUbyModel(processors, **"Ice Lake"**))  
 print(getCPUbyModel(processors, **"1068G7"**))  
 print(getCPUbyModel(processors, **"Ryzen 7 1700"**))  
 print(getCPUbyModel(processors, **"Ryzen 5 1700"**))  
  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 main()

Результат выполнения программы

**Задание №1**

('Екатерина Петрова Витальевна', 'пл. Садовая, д. 7, кв. 7')

пл. Садовая, д. 7, кв. 7

Запись не найдена

**Задание №2**

('Виталий Смолов Геннадьевич', '+7 499 779-31-41')

+7 499 779-31-41

Запись не найдена

**Задание №3**

Запись уже существует

Запись: Вероника Гагарина Витальевна = ул. Множеств, д. 7, кв. 17 добавлена

ул. Множеств, д. 7, кв. 17

Запись уже существует

Запись: Павел Василий Васильевич = +7 999 999-99-99 добавлена

+7 999 999-99-99

**Задание №4**

{('Ice Lake', '1068G7'): (4, 8), ('Ice Lake', '1005G1'): (2, 4)}

{}

None

(('Ice Lake', '1068G7'), (4, 8))

(('Zen', 'Ryzen 7 1700'), (8, 16))

None

Вывод

В ходе выполнения практической работы были получены навыки работы со словарями в языке Python, а также опыт генерации списков со случайными значениями.