Семинар 10

Словари

Словари

Словарь (dictionary) — это ассоциативный массив (или еще его называют хеш-таблица). Это неупорядоченное множество пар **ключ:значение** с требованием **уникальности** ключей.
В отличие от кортежей и списков, доступ к элементам словаря производится по ключу, а не по индексу, ключ может быть любого неизменяемого типа. (Какие типы неизменяемые?)

Основные операции над словарем — сохранение объекта с заданным ключом, извлечение по нему значения и перебор ключей и значений словаря.

```
myDict = {}  # или myDict = dict() - пустой словарь
myDict = {1: 2, 2: 4, 3: 9}  # статическое задание словаря
```

Простые операции над словарями

```
myDict = {'name': 'Jack', 'age': 20, (1, 2): 10} # ключи могут быть любыми

myDict['age'] = 27  # обновление данных
print(myDict)  # {'age': 27, 'name': 'Jack'}

myDict['address'] = 'Novaja str.'  # добавление или обновление
print(myDict)  # {'address': 'Novaja str.', 'age': 27, 'name': 'Jack'}

# будет ли некорректное завершение работы программы?
currentYear = datetime.datetime.today().year
print("%s was born in %i year" % (myDict['name'], currentYear - myDict['age']))
# а так?
# print("%s was born in %i year" % (myDict['name'], currentYear - myDict['SomeOtherAge'])) # KeyError!

age = currentYear - myDict.get('SomeOtherAge', -1)  # -1 значение, которое возвращается, если ключа нет
print("%s was born in %i year" % (myDict['name'], age))  # Почти хорошо, разве что Джек родился в будущем
```

Методы словарей

dict.**clear**() - очищает словарь.

dict.**copy**() - возвращает копию словаря, myDict1 = myDict2 - только создает еще одну ссылку-псевдоним

dict.**fromkeys**(seq[, value]) - создает словарь с ключами из seq и значением value (по умолчанию None).

dict.**get**(key[, default]) - возвращает значение ключа, но если его нет - возвращает default (или None).

dict.**items**() – возвращает iterable пар (ключ, значение).

dict.**keys**() - возвращает iterable последовательность ключей в словаря.

dict.**pop**(key[, default]) - **удаляет** ключ и **возвращает** значение. Если ключа нет, возвращает default (по умолчанию **бросает исключение**).

dict.**setdefault**(key[, default]) - возвращает значение ключа, но если его нет, не бросает исключение, а создает ключ с значением default (по умолчанию None).

dict.**update**([other]) - обновляет словарь, добавляя пары (ключ, значение) из other в dict. Существующие ключи перезаписываются.

dict.**values**() - возвращает iterable значений в словаре.

Задача 1

```
# Подсчитать, сколько раз в тексте употреблялись имена собственные и какие это имена
# Вывести в упорядоченном виде от большего числа употреблений к меньшему.
# Для простоты в тексте имена встречаются только в именительном падеже
text = input("Введите текст:\n")
names = {}
# поиск имен и добавление в словарь
for word in text.split():
    if word[0].isupper():
        if word in names:
            names[word] += 1
        else:
            names[word] = 1
# сортировка
result = sorted(names.items(), key=lambda x: x[1], reverse=True)
print(result)  # result - список кортежей (key, value)
```

Особенности работы со словарями

Вместо:

```
if word in names:
    names[word] += 1
else:
    names[word] = 1
```

Можно использовать:

```
names[word] = names.setdefault(word, 0) + 1
```

Задача 2

```
# В файле хранится генеалогическое дерево собак за несколько поколений
# Формат хранения Кличка-родитель1-родитель2-возраст
# Для введенных 2х имен определить являются ли они прямыми потомками


file = open("z2.txt", 'r')
dogs = {}
for line in file.readlines():
    dogInfo = line.split("-")
    if len(dogInfo) == 4:
        dogs[dogInfo[0]] = {"parents": tuple(dogInfo[1:3]),
                            "age": int(dogInfo[3].strip())}
print(dogs)
file.close()
```

Задача 2

```
dog1 = input("Введите кличку собаки1:")
dog2 = input("Введите кличку собаки2:")
# dog1 = "Bob"
# dog2 = "Annie"
if dog1 not in dogs:  # Проверка существования ключа
    print(dog1, "нет в списке")
    exit(0)
if dog2 not in dogs:
    print(dog2, "нет в списке")
    exit(0)
if dog2 in dogs[dog1]['parents']:
    print(dog2, "является родителем", dog1)
elif dog1 in dogs[dog2]['parents']:
    print(dog1, "является родителем", dog2)
else:
    print("Прямое родство не обнаужено")
```

Задача 3

```
# На полках в магазине храняться различные товары.
# Полка может выдержать N кг товара (задается константой).
# Для заданных в файле товаров посчитать выдержат ли их соответствующие полки.
# Формат файла "Товар: полка, количество, вес 1 упаковки"
```