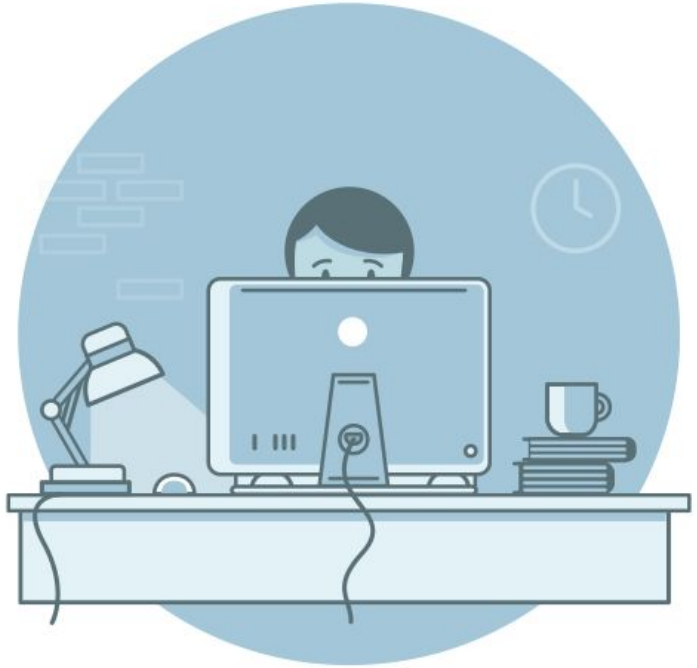
Моя IT Школа

сертифицированные IT курсы

Коллекции. Словари



Проверка домашнего задания

Дан список list=[15,48,'hello',6,19,'world’].

Все числа этого списка проверить на чётность. Если число чётное, то посчитать сумму его цифр. Если нечётное, то заменить его на 1 в списке.

Все слова: посчитать количество гласных и согласных.

Вывести всё на экран.

Решение

Моя Школа

```
list = [15, 48, 'hello', 6, 19, 'world']
l = 0
h = 0
d = 0

for i in list:
    if type(i) is int:
        if i % 2 == 0:
            i = str(i)
            for k in i:
                k = int(k)
                l += k
            print(i, "Сумма цифр: ", l, "\n")
        else:
            index = list.index(i)
            list[index] = 1
    elif type(i) is str:
        for r in i:
            if r in "aeoiu":
                h += 1
            else:
                d += 1
        print(i, "\nКоличество глассных: ", h)
        print("Количество согласных: ", d, "\n")
        h = 0
        d = 0
print(list)
```

План занятия

Моя IT Школа
сертифицированные IT курсы

1. Словари в Python
2. Методы словарей
3. Операции со словарями
4. Использование вложенных словарей
5. Встроенная функция zip()
6. Сортировка словаря

Коллекции. Словари

Словари – это встроенный тип данных, который является ассоциативным массивом или хешем и базируется на отображении пар типа **(ключ:значение)**.
Чтобы работать со словарём, его нужно создать. Создать его можно несколькими способами.
Во-первых, с помощью литерала:

```
d = {}
d = {'dict': 1, 'dictionary': 2}
print(d)
```

```
{'dict': 1, 'dictionary': 2}

Process finished with exit code 0
```

Коллекции. Словари

Моя Школа

Во-вторых, с помощью функции **dict**:

```
d = dict(short='dict', long='dictionary')
d_2 = dict([(1, 1), (2, 4)])
print(d, '\n', d_2)
```

```
{'short': 'dict', 'long': 'dictionary'}
 {1: 1, 2: 4}

Process finished with exit code 0
```

В-третьих, с помощью метода **fromkeys**:

```
d = dict.fromkeys(['a', 'b'])
d_2 = dict.fromkeys(['a', 'b'], 100)
print(d, '\n', d_2)
```

```
{'a': None, 'b': None}
 {'a': 100, 'b': 100}

Process finished with exit code 0
```

В-четвертых, с помощью генераторов словарей:

```
d = {a: a ** 2 for a in range(7)}
print(d)
```

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36}

Process finished with exit code 0
```

Коллекции. Словари

Моя Школа

В словаре, если известен ключ, то доступ к значению по эту ключу можно получить с помощью операции **[]**. В этот же способ можно изменить значение, если известен ключ, который соответствует этому значению.
Теперь попробуем добавить записей в словарь и извлечь значения ключей:

```
d = {1: 2, 2: 4, 3: 9}
d[4] = 4 ** 2

print(d[1])
print(d)
```

```
2
{1: 2, 2: 4, 3: 9, 4: 16}

Process finished with exit code 0
```

Методы словарей

Моя IT Школа

Метод	Результат
dict.clear()	Очищает словарь.
dict.copy()	Возвращает копию словаря.
classmethod 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, не бросает исключение, а возвращает default (по умолчанию None)
dict.items()	Возвращает пары (ключ, значение).

Методы словарей

Моя IT Школа

Метод	Результат
dict.keys()	Возвращает ключи в словаре.
dict.pop(key[, default])	Удаляет ключ и возвращает значение. Если ключа нет, возвращает default (по умолчанию бросает исключение).
dict.popitem()	Удаляет и возвращает последнюю пару (ключ, значение). Если словарь пуст, бросает исключение KeyError. Помните, что словари не упорядочены.
dict.setdefault(key[, default])	Возвращает значение ключа, но если его нет, не бросает исключение, а создает ключ со значением default (по умолчанию None).
dict.update([other])	Обновляет словарь, добавляя пары (ключ, значение) из other. Существующие ключи перезаписываются. Возвращает None (не новый словарь!).
dict.values()	Возвращает значения в словаре.

Коллекции. Словари


```
Months = { 1:'Jan', 2:'Feb', 3:'Mar', 4:'Apr',
           5:'May', 6:'Jun', 7:'Jul', 8:'Aug',
           9:'Sep', 10:'Oct', 11:'Nov', 12:'Dec' }
count = len(Months)
print(count)
```

```
12

Process finished with exit code 0
```

Операция del. Удаление элемента по ключу

Операция **del** предназначена для удаления элемента из словаря на основе заданного ключа **key**. Общая форма использования операции, следующая:

del D[key]

D – заданный словарь;

key – ключ в словаре, элемент которого нужно удалить.

Если указать несуществующий ключ **key**, то будет сгенерировано исключение KeyError.

Коллекции. Словари

Моя Школа

```python
# Операция del - удаление элемента из словаря

# Исходный словарь
Salary = {'Director': 120800.0,
          'Secretary': 101150.25,
          'Locksmith': 188200.00}

print(Salary)
# Удалить элемент по ключу 'Secretary'
del Salary['Secretary']
print(Salary)

# Попытка удалить несуществующий ключ
# del Salary[5] - так нельзя, генерируется исключение KeyError: 5
# del Salary['None'] - тоже запрещено
```

```
{'Director': 120800.0, 'Secretary': 101150.25, 'Locksmith': 188200.0}
{'Director': 120800.0, 'Locksmith': 188200.0}

Process finished with exit code 0
```

Использование вложенных словарей

Моя Школа

```
Position = {'Manager': {'Director':'Petrov',
                        'Deputy Director':'Peter'},
           'Teacher': {'Specialist':'Masha',
                       'Methodist':'Dima',
                       'Senior Lecturer':'Ivan'}}

count1 = len(Position)
print(Position['Manager']['Director'])
print(Position['Teacher']['Methodist'])


print(Position,'len:', len(Position),'\n',
      Position['Manager'],'len:', len(Position['Manager']),'\n',
      Position['Teacher'],'len:', len(Position['Teacher']), '\n')
```

```
Petrov
Dima
{'Manager': {'Director': 'Petrov', 'Deputy Director': 'Peter'}, 'Teacher': {'Specialist': 'Masha', 'Methodist': 'Dima', 'Senior Lecturer':
'Ivan'}} len: 2
 {'Director': 'Petrov', 'Deputy Director': 'Peter'} len: 2
 {'Specialist': 'Masha', 'Methodist': 'Dima', 'Senior Lecturer': 'Ivan'} len: 3


Process finished with exit code 0
```

Операция in. Определение наличия ключа в словаре

Чтобы определить, существует ли заданный ключ в словаре, в Python используется операция **in**. Общая форма использования операции **in** следующая:

f_is = key in D

D – исходный словарь;

key – ключ, наличие которого в словаре D нужно определить;

f_is – результат логического типа. Если **f_is**=**True**, то ключ **key** присутствует в словаре. Если **f_is=False**, то ключа нету в словаре.

Коллекции. Словари

Моя IT Школа

В примере используется операция **in** для того, чтобы определить есть ли в словаре ключ Salary, который нужно удалить. Операция используется в условном операторе **if**.

```
# Операция in - определение наличия ключа в словаре
# Исходный словарь
Salary = {'Director': 120800.0,
          'Secretary': 101150.25,
          'Locksmith': 188200.00}
print(Salary)

# Удалить элемент по ключу 'Secretary' с проверкой
key = 'Secretary'
if key in Salary:
    del Salary['Secretary']
    print(Salary)

# Попытка удалить несуществующий ключ
# если ключа нету, то исключение KeyError не генерируется
key2 = 5
if key2 in Salary:
    del Salary[key2]
```

```
{'Director': 120800.0, 'Secretary': 101150.25, 'Locksmith': 188200.0}
{'Director': 120800.0, 'Locksmith': 188200.0}

Process finished with exit code 0
```

Операция not in. Определение отсутствия ключа в словаре

Операция **not in** возвращает результат всегда противоположный операции **in**. Общая форма операции **not in** следующая:

f_is = key not in D

D – исходный словарь;

key – ключ, наличие которого в словаре D нужно определить;

f_is – результат логического типа. Если **f_is = True**, то ключа **key** нету в словаре **D**. Если **f_is = False**, то ключ **key** присутствует в словаре **D**.

Коллекции. Словари

Моя Школа

Пример. В примере демонстрируется использование операции not in для определения того, присутствует ли в словаре ключ, который был введен из клавиатуры.

```
# Операция not in - определение отсутствия ключа в словаре
# Формирование словаря слов с их числовым эквивалентом

# 1. Сформировать пустой словарь
Words = dict()# Words = {}

# 2. Ввести количество слов в словаре
count = int(input("Количество слов в словаре: "))

# 3. Цикл добавления слов
i=0
while i<count:
    print("Ввод слов")
    wrd = str(input("Слово:"))
    value = int(input("Значение: "))

    # Если ключа wrd нет в словаре, то добавить пару [wrd:value]
    if wrd not in Words:
        Words[wrd] = value
    i=i+1
# Вывести сформированный словарь
print(Words)
```

```
Количество слов в словаре: 2
Ввод слов
Слово:one
Значение: 1
Ввод слов
Слово:two
Значение: 2
{'one': 1, 'two': 2}

Process finished with exit code 0
```

Встроенная функция zip().

Моя IT Школа

Функция **zip()** позволяет создать словарь путем объединения списков ключей и значений.

```
# Словари. Функция zip()

# Создание словаря из списков ключей и значений
Numbers = dict(zip([1, 2, 3], ['One', 'Two', 'Three']))
print(Numbers)
```

```
{1: 'One', 2: 'Two', 3: 'Three'}

Process finished with exit code 0
```

Обход словаря с помощью цикла for.

Моя Школа

В примере демонстрируется обход словаря с помощью цикла for и вывод всех пар **(ключ: значение).**

```
# Работа со словарями
# Обход словаря с помощью цикла for

# Исходный словарь
Months = {1:'Jan', 2:'Feb', 3:'Mar',
          4:'Apr', 5:'May', 6:'Jun',
          7:'Jul', 8:'Aug', 9:'Sep',
          10:'Oct', 11:'Nov', 12:'Dec'}

# Цикл for обхода словаря
# в цикле mn - ключ, Months[mn] - значение
for mn in Months:
    print(mn, ': ', Months[mn])
```

```
1 :  Jan
2 :  Feb
3 :  Mar
4 :  Apr
5 :  May
6 :  Jun
7 :  Jul
8 :  Aug
9 :  Sep
10 :  Oct
11 :  Nov
12 :  Dec

Process finished with exit code 0
```

Сортировка словаря

Моя IT Школа

Как известно, ключи в словаре сохраняются в произвольном порядке. Если возникает необходимость отсортировать словарь по ключам, то для этого можно использовать метод **sort()**, который используется для списков. Для этого предварительно нужно конвертировать представление ключей в список.

```
# Словари. Сортировка по ключам

# Исходный словарь
A = {'f':10, 'a':2, 'c':17}

# Сортировка по ключам
# 1. Получить представление ключей
ak = A.keys()

# 2. Конвертировать представления ak в список
list_ak = list(ak)

# 3. Отсортировать список ключей - функция sort()
list_ak.sort()

# 4. Вывести перечень (ключ:значение)
# в отсортированном порядке по ключам,
# сформировать новый словарь
B = {}
for k in list_ak:
    print('(', k, ': ', A[k], ')')
    B[k] = A[k]

print(B)
```

Задание №1

Создайте словарь person, в котором будут присутствовать ключи name, age, city.

Выведите значение возраста из словаря person.

Решение

Моя Школа

```python
# 1. Выведите значение возраста из словаря person.
person = {"name": "Kelly", "age":25, "city": "New york"}
print(person['age'])
```

```
25

Process finished with exit code 0
```

Задание №2

Значениями словаря могут быть и списки.

Создайте словарь с ключами BMW, Tesla и списками из 3х моделей в качестве значений.

Выведите первое и последнее значения каждого из ключей.

Решение

Моя Школа

```
# 2. Значениями словаря могут быть и списки.
# Допишите словарь с ключами BMW, Tesla
# и списками из 3х моделей в качестве значений.
models_data = {"BMW": ["Model_1","Model_2","Model_3"],
               "Tesla": ["Modes S", "Model A", "Model B"]}
print(models_data["Tesla"][0], models_data["Tesla"][2])
print(models_data["BMW"][0], models_data["BMW"][2])
```

```
Modes S Model B
Model_1 Model_3

Process finished with exit code 0
```

Задание №3

Исправьте ошибки в коде, чтобы получить требуемый вывод. (Вывод True)

d1 = {"a": 100. "b": 200. "c":300}

d2 = {a: 300. b: 200, d:400}

print(d1["b"] == d2["b"])

Решение

Моя Школа

```python
# 3. Исправьте ошибки в коде, что бы получить требуемый вывод.
d1 = {"a": 100, "b": 200, "c": 300}
d2 = {"a": 300, "b": 200, "d": 400}

print(d1["b"] == d2["b"])
```

```
True

Process finished with exit code 0
```

Задание №4

Дан словарь с числовыми значениями. Необходимо их все перемножить и вывести на экран.

Решение

Моя Школа

```
my_dictionary = {'data1': 375, 'data2': 567, 'data3': 37, 'data4': 21}
result = 1
for key in my_dictionary:
    result = result * my_dictionary[key]
print(result)
```

```
165209625

Process finished with exit code 0
```

Задание №5

Даны два списка одинаковой длины. Необходимо создать из них словарь таким образом, чтобы элементы первого списка были ключами, а элементы второго — соответственно значениями нашего словаря.

Решение

Моя Школа

```
keys = ['red', 'green', 'blue']
values = ['#FF0000', '#008000', '#0000FF']
color_dictionary = dict(zip(keys, values))
print(color_dictionary)
```

```
{'red': '#FF0000', 'green': '#008000', 'blue': '#0000FF'}

Process finished with exit code 0
```

Задание №6

Создайте словарь из строки 'pythonist' следующим образом: в качестве ключей возьмите буквы строки, а значениями пусть будут числа, соответствующие количеству вхождений данной буквы в строку.

Решение

Моя Школа

```
str1 = 'pythonist'
my_dict = {i: str1.count(i) for i in str1}
print(my_dict)
```

```
{'p': 1, 'y': 1, 't': 2, 'h': 1, 'o': 1, 'n': 1, 'i': 1, 's': 1}

Process finished with exit code 0
```

Домашнее задание

Задание №1

У вас есть словарь, где ключ – название продукта.
Значение – список, который содержит цену и кол-во товара.
Выведите через ‘’–’’ название – цену – количество.
С клавиатуры вводите название товара и его кол-во. n – выход из программы. Посчитать цену выбранных товаров и сколько товаров осталось в изначальном списке.