PYTHON FOR COMPUTATIONAL PROBLEM SOLVING

Dictionaries in python

Prof. Sindhu R Pai
PCPS Theory Anchor - 2024
Department of Computer Science and Engineering

Dictionary

- Dictionary is a data structure that **organizes data into key and value pairs**
- Every value has a **certain unique key mapped** to it
- It is **mutable** (values can be changed after creation)

Dictionary Creation

```
>>> phonebook={} # Creation of empty dictionary
>>> phonebook={"Johan":938477565} # Dictionary with one key-value pair
>>> phonebook={"Johan":938477565,"Jill":938547565} # Dictionary with two key-value pair
```

Dictionary

To access values in the dictionary, we use the keys like so:

```
>>> eng2french={ 1:’un’,
                 2:’deux’,
                 3:’trios’,
                 4:’quatre’,
                 5:’cinq }
 >>> print(eng2french[2]) #will give you ‘deux’
```

Dictionary

- Each **key is of any immutable type** associated with a single value.

```
>>> d={1:"one",2:"two",3:"three",4:"four"}
>>> d1={[1,2]:"hello"}
        Traceback (most recent call last):  File "<stdin>", line 1, in
        <module> TypeError: unhashable type: 'list'
```

- The **values can be of any type**.

```
>>> d={1:"one", 2:[23,33] , 3:"three", 4:"four"}
```

- If you assign a value to a key, then later in the same dictionary have the same key assigned to a new value, the **previous value will be overwritten**.

```
>>> d={1:"one",2:"two",3:"three"}
>>> d={1:"one",2:"two",3:"three",1:"five"}
>>> d
    {1: 'five', 2: 'two', 3: 'three'}
```

PYTHON FOR COMPUTATIONAL PROBLEM SOLVING

Dictionary

- The items (key-value pair) in dictionary are unordered, which means that the order isn’t decided by the programmer but rather the interpreter.

- The ordering is based on a concept called “**hashing**” .

```
>>> d={'w':11,'a':33,'e':44}
>>> f= {'e':44,'w':11,'a':33}
>>> print(d==f)
      True
```

Common operations on dictionaries

- len()
- min()
- max()

Note: Dictionaries do not support ‘+’ and ‘*’ operations

Dictionary Functions

- **get():** returns the value for a given key, if present.

```
>>> print(phonebook.get('Jill’))
      938547565
```

- **items(...)**

  D.items() -> a set-like object providing a view on D’s items.

```
>>> phonebook.items()
      dict_items([('Johan', 938477565), ('Jill', 938547565)])
```

Dictionary

- **keys(...)**

D.keys() -> a set-like object providing a view on D’s keys.

```
>>> phonebook.keys()
    dict_keys(['Johan', 'Jill’])
```

- **pop(...)**

D.pop(key) -> v, remove specified key and return the corresponding value. If key is not found, otherwise KeyError is raised.

```
>>> phonebook.pop(‘Jill’)
    938547565
```

- **popitem(...)**

D.popitem() -> (k, v); remove and return some (key, value) pair as a 2-tuple, but raise KeyError if D is empty.

```
>>> person = {'name': 'Phill', 'age': 22, 'salary': 3500.0}
>>> result = person.popitem()
>>> print('Return Value = ', result)
     Return Value =  ('salary', 3500.0)
>>> print('person = ', person)
     person =  {'name': 'Phill', 'age': 22}
```

- **setdefault(...)**

D.setdefault(key,value) -> if the key is in the dictionary, returns its value. If the key is not present, insert the key with a specified value and returns that same value.

```
>>> person = {'name': 'Phil', 'age': 22}
>>> phone = person.setdefault(‘phone’, 90909090)
>>> print('person = ',person)
>>> print(‘phone = ‘,phone)
```

Output:

```
person = {‘name’: ‘Phil’, ‘age’: 22, ‘phone’: 90909090}
phone = 90909090
```

- **update(...)**

  D.update() -> updates content of D with key-value pairs from a dictionary/iterable that it is given

```
>>> marks = {'Physics':67, 'Maths':87}

>>> internal_marks = {'Practical':48}

>>> marks.update( [(‘Chemistry’, 90), (‘Python’, 100)] )

>>> marks.update(internal_marks)

>>> print(marks)

    {'Physics': 67, 'Maths': 87, ‘Chemistry’: 90, ‘Python’: 100, 'Practical': 48}
```

PYTHON FOR COMPUTATIONAL PROBLEM SOLVING

Dictionary

- **values(...)**

D.values() -> returns a view object that displays a list of all the values in the dictionary.

```
>>> marks = {'Physics':67, 'Maths':87}
>>> print(marks.values())
    dict_values([67, 87])
```

Use of for and while loops for dictionary

1. for loop

```
Ex: (1)  dict = {'a': 'pencil', 'b': ‘eraser', 'c': 'sharpner’}
         for key, value in dict.items():
                  print(key, value)
Ex: (2) dict = {'a': 'juice', 'b': 'grill', 'c': 'corn’}
         for key in dict:
                  print(key, dict[key])
```

Use of for and while loops for dictionary

2. while loop

Ex:

```
books={"learning python": "Mark Lutz", "think python": "Allen B. Downey",
           "Fluent Python": "Luciano Ramalho"}
key=list(books) #converts keys into a list
i=0
while i<len(key):
        print(key[i],":",books[key[i]])
        i+=1
```

PES UNIVERSITY
CELEBRATING 50 YEARS

THANK YOU

Department of Computer Science and Engineering

Dr. Shylaja S S, Director, CCBD & CDSAML, PESU
Prof. Sindhu R Pai – sindhurpai@pes.edu
Prof. Chitra G M
Prof. Mohan Kumar AV