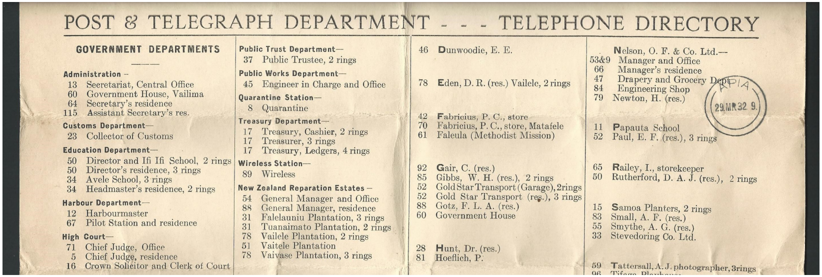
Dictionaries - a definition

A dictionary keeps an association between a set of **keys** and their **values**. For example: lets take a phonebook, names could be the keys and the numbers could be the values:



POST & TELEGRAPH DEPARTMENT - - - TELEPHONE DIRECTORY

GOVERNMENT DEPARTMENTS

Administration –
13 Secretariat, Central Office
60 Government House, Vailima
64 Secretary's residence
115 Assistant Secretary's res.

Customs Department—
23 Collector of Customs

Education Department—
50 Director and Ifi Ifi School, 2 rings
50 Director's residence, 3 rings
34 Avele School, 3 rings
34 Headmaster's residence, 2 rings

Harbour Department—
12 Harbourmaster
67 Pilot Station and residence

High Court—
71 Chief Judge, Office
5 Chief Judge, residence
16 Crown Solicitor and Clerk of Court

Public Trust Department—
37 Public Trustee, 2 rings

Public Works Department—
45 Engineer in Charge and Office

Quarantine Station—
8 Quarantine

Treasury Department—
17 Treasury, Cashier, 2 rings
17 Treasurer, 3 rings
17 Treasury, Ledgers, 4 rings

Wireless Station—
89 Wireless

New Zealand Reparation Estates –
54 General Manager and Office
88 General Manager, residence
31 Falelauniu Plantation, 3 rings
31 Tuanaimato Plantation, 2 rings
78 Vailele Plantation, 2 rings
51 Vaitele Plantation
78 Vaivase Plantation, 3 rings

46 **D**unwoodie, E. E.

78 **E**den, D. R. (res.) Vailele, 2 rings

42 **F**abricius, P. C., store
70 Fabricius, P. C., store, Matafele
61 Faleula (Methodist Mission)

92 **G**air, C. (res.)
85 Gibbs, W. H. (res.), 2 rings
52 Gold Star Transport (Garage), 2rings
52 Gold Star Transport (res.), 3 rings
88 Gotz, F. L. A. (res.)
60 Government House

28 **H**unt, Dr. (res.)
81 Hoeflich, P.

Nelson, O. F. & Co. Ltd.—
53&9 Manager and Office
66 Manager's residence
47 Drapery and Grocery Dept
84 Engineering Shop
79 Newton, H. (res.)

APIA
29.MR32 9.

11 **P**apauta School
52 Paul, E. F. (res.), 3 rings

65 **R**ailey, I., storekeeper
50 Rutherford, D. A. J. (res.), 2 rings

15 **S**amoa Planters, 2 rings
83 Small, A. F. (res.)
55 Smythe, A. G. (res.)
33 Stevedoring Co. Ltd.

59 **T**attersall, A. J. photographer, 3rings

Figure 1: Telephone directory

By Samoa Post & Telegraph Department, via Wikimedia Commons

Creating and accessing a dictionary

- There are several ways to create a dictionary. You can start off with an empty dictionary and then add key/value pairs to it:

```
name = {}              # syntax 1 (preferred)
name = dict()          # syntax 2
```

- You can create a dictionary with initial key/value pairs using the syntax:

```
name = {key: value, key: value, ..., key: value}
```

Note: the colon in between the key and the value.

```
# dictionary with initial data
phonebook = {"Allison": "(520)555-6789", "Marty": "(650)555-1234"}
```

Lookup elements in a dictionary

```
>>> phonebook["Allison"]
'(520)555-6789'
>>>
>>> phonebook["Marty"]
'(650)555-1234'
```

Modify elements in a dictionary

```
>>> phonebook["Allison"] = '(01279) 36993'
```

Gotcha - Dictionary keys are unique

- The keys in a dictionary are unique and if you add a key, value pair where the key already exists, the original key, value pair will be overwritten.
- You will not receive an error or a warning. Here is an example using the Python shell, you can assume that the dictionary has been set up.

```
>>> # replacing a value in a dictionary
>>> phonebook["Allison"]
'(520)555-6789'
>>> phonebook["Allison"] = "(444)555-8800"
>>> phonebook["Allison"]
'(444)555-8800'
```

Gotcha - More than one key can be associated with the same value

- It is perfectly legal to have two or more keys that refer to the same value. Using the previous example:

```
>>> # dictionary where two keys pair with the same value
>>> phonebook
{'Allison': '(520)555-6789', 'Marty': '(650)555-1234'}
>>> phonebook["Yana"] = "(650)555-1234"   # duplicate value
>>> phonebook["Marty"]
'(650)555-1234'
>>> phonebook["Yana"]
'(650)555-1234'
>>> phonebook
{'Allison': '(520)555-6789', 'Marty': '(650)555-1234',
  'Yana': '(650)555-1234'}
```

Gotcha - Removing dictionary items

- To remove an item from a dictionary, call the `pop()` method with the key as the argument. Here is an example using the Python shell.

```
>>> phonebook = {}                          # create empty dict
>>> phonebook["Allison"] = "(520)555-6789"  # store a pair
>>> phonebook["Marty"]   = "(650)555-1234"  # store another pair
>>> phonebook.pop("Allison")               # delete entry with key "All
'(520)555-6789'
>>> phonebook["Allison"]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'Allison'
>>>
```

- You can see that if a key doesn't exist, you will get a KeyError exception. We will look at exceptions in a future lecture.

Traversing a dictionary

- Just like sets, you cannot access the elements in a dictionary by position.
- This means that traversing a dictionary is done in much the same way as a set.

```
for name in phonebook:
  print(name, phonebook[name])
```

Dictionary operations

Operation	Description
`dict[key]`	returns the value associated with the given key; raises `KeyError` if not found
`dict[key] = value`	sets the value associated with the given key; replaces if already found
`del dict[key]`	removes the given key and its paired value; raises `KeyError` if not found
`key in dict`	returns `True` if the given key is found
`key not in dict`	returns `True` if the given key is *not* found
`len(dict)`	number of key/value pairs
`str(dict)`	returns string representation such as `"{'a':1, 'b':2}"`
`dict.clear()`	removes all key/value pairs
`dict.get(key,default)`	returns the value associated with the given key; returns `default` if not found
`dict.items()`	returns the contents of the dictionary as a sequence of (key, value) tuples
`dict.keys()`	returns the keys in the dictionary as a sequence
`dict.pop(key)`	returns the value associated with the given key, and removes that key/value pair
`dict.update(dict2)`	adds all key/value pairs from another dictionary, replacing if keys are already present
`dict.values()`	returns the values in the dictionary as a sequence

Question 1

- Given the following dictionary definition

```
favoriteFoods = {"Peg": "burgers", "Cy": "hotdogs",\
   "Bob": "apple pie"}
```

  Which code segment correctly prints the dictionary, both the key and value, in alphabetical order by the person's name?

  (i)
```
print(favoriteFoods)
```

  (ii)
```
for name in sorted(favoriteFoods) :
print(name, favoriteFoods[name])
```

  (iii)
```
for name in (favoriteFoods) :
    print(name, favoriteFoods[name])
```

  (iv)
```
for name in sorted(favoriteFoods) :
    print(favoriteFoods[name])
```

Question 2

- Given the dictionary `periodicTable` that contains the periodic table of the elements, which of the following correctly prints the values stored in the table, one per line?

(i)
```
print(periodicTable)
```

(ii)
```
print(values(periodTable))
```

(iii)
```
for value in periodicTable :
    print(value)
```

(iv)
```
for value in periodicTable.values() :
    print(value)
```

p 486, Horstmann, Cay S., Rance Necaise. Python for Everyone, Interactive Edition, 2nd Edition. Wiley, 2016-05-09. VitalBook file.

Answer 1

(ii)
```
for name in sorted(favoriteFoods) :
    print(name, favoriteFoods[name])
```

Answer 2

(iv)
```
for value in periodicTable.values() :
    print(value)
```