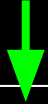




CL07 - Dictionaries

Dictionaries

Keys



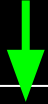
Values



Flavor	Num Orders
"chocolate"	12
"vanilla"	8
"strawberry"	5

Dictionaries

Keys



Values



Flavor	Num Orders
"chocolate"	12
"vanilla"	8
"strawberry"	5

Lists

Indexes



Values



0	12
1	8
2	5

Also called: Map, Hashmap, Key-Value Store

Syntax

Data type:

name: dict[<key type>, <value type>]

temps: dict[str, float]

Construct an empty dict:

dict() or {}

Construct a populated dict:

temps: dict[str, float] = {"Florida": "72", "Raleigh": "56"}

Do it yourself!

Create a dictionary called ice_cream that stores the following orders

Keys	Values
chocolate	12
vanilla	8
strawberry	5

Adding elements

We use subscription notation.

`<dict name>[<key>] = <value>`

`temps["DC"] = 52`

Do it yourself!

Add 3 orders of "mint" to your ice_cream dictionary.

Removing elements

Similar to lists, we use pop()

`<dict name>.pop(<key>)`

`temps.pop("Florida")`

Do it yourself!

Remove the orders of "mint" from
ice_cream.

Access + Modify

To access a value,
use subscription notation:

```
<dict name>[<key>]  
temps["DC"]
```

To modify, also use subscription notation:

```
<dict name>[<key>] = new_value  
temps["DC"] = 53 or temps["DC"] += 1
```

Do it yourself!

Print out how many orders there
are of "chocolate".
Update the number of orders of
Vanilla to 10.

Length of dictionary

`len(<dict name>)`

`len(temps)`

Do it yourself!

Print out the length of ice_cream.

What exactly is this telling you?

Check if key in dictionary

`<key> in <dict name>`

`"DC" in temps`

`"Florida" in temps`

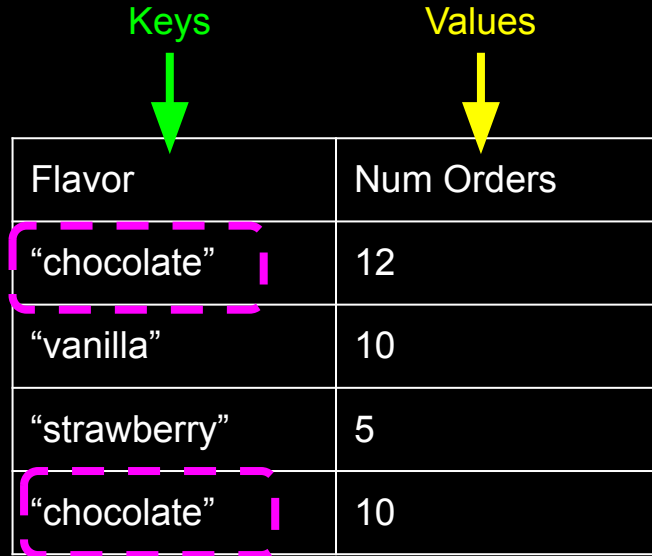
Do it yourself!

Check if both the flavors "mint" and "chocolate" are in ice_cream.

Write a conditional that behaves the following way:
If "mint" is in ice_cream, print out how many orders of "mint" there are.
If it's not, print "no orders of mint".

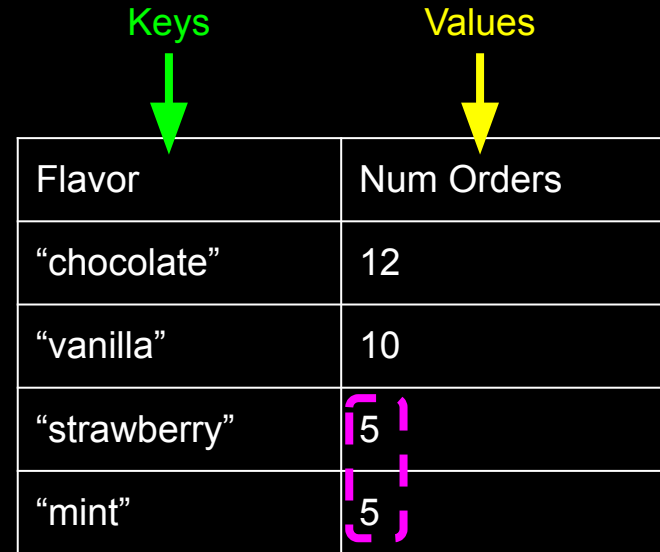
Important Note: Can't Have Multiple of Same Key

(Duplicate values are okay.)



A diagram illustrating a valid dictionary structure. A green arrow labeled 'Keys' points to the 'Flavor' column, and a yellow arrow labeled 'Values' points to the 'Num Orders' column. The table has four rows. The first two rows are grouped by a dashed pink bracket on the left, indicating they share the same key. The last two rows are also grouped by a dashed pink bracket on the left, indicating they share the same key. The keys are 'chocolate', 'vanilla', 'strawberry', and 'chocolate'. The values are 12, 10, 5, and 10.

Flavor	Num Orders
"chocolate"	12
"vanilla"	10
"strawberry"	5
"chocolate"	10



A diagram illustrating an invalid dictionary structure. A green arrow labeled 'Keys' points to the 'Flavor' column, and a yellow arrow labeled 'Values' points to the 'Num Orders' column. The table has five rows. The first three rows are grouped by a dashed pink bracket on the left, indicating they share the same key. The last two rows are also grouped by a dashed pink bracket on the left, indicating they share the same key. The keys are 'chocolate', 'vanilla', 'strawberry', and 'mint'. The values are 12, 10, 5, and 5.

Flavor	Num Orders
"chocolate"	12
"vanilla"	10
"strawberry"	5
"mint"	5

“for” Loops

“for” loops iterate over the **keys** by default

Do it yourself!

Use a for loop to print:
chocolate has 12 orders.
vanilla has 10 orders.
strawberry has 5 orders.

```
for key in ice_cream:  
    print(key)
```

```
for key in ice_cream:  
    print(ice_cream[key])
```

Flavor	Num Orders
“chocolate”	12
“vanilla”	10
“strawberry”	5

Dicts in Memory

```
1 jerseys: dict[int, str] = {1: "Alyssa", 2: "Shefali"}  
2 √ for number in jerseys:  
3   |     print(jerseys[number])
```

To do:

CQ05 + LS17

Instructions on Site

(Feel free to raise your hand with questions!)