

Data structures

`list` and `dict` for storing multiple values

- add
- delete
- join
- search
- loop
- sort

list: a list of (any) values

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]  
numbers = [1, 2, 3, 4, 5]  
any_type_you_want = [1, "meow", 3.14, True]
```

<https://docs.python.org/3/tutorial/datastructures.html#more-on-lists>

Access item using index

```
# Access
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]

print(cities)      # ["Montreal", "Toronto", "Vancouver", "Detroit"]
print(cities[0])   # Montreal
print(cities[1])   # Toronto
print(cities[2])   # Vancouver
print(cities[3])   # Detroit
print(cities[4])   # ???

# Update
cities[0] = "New York"

print(cities)      # ["New York", "Toronto", "Vancouver", "Detroit"]
```

Add item to list - **append**

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]  
cities.append("New York")  
print(cities) # ["Montreal", "Toronto", "Vancouver", "Detroit", "New York"]
```

Delete item from list - `del`

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]  
del cities[0]  
print(cities) # ["Toronto", "Vancouver", "Detroit"]  
del cities[0]  
print(cities) # ["Vancouver", "Detroit"]
```

Join two lists - `+`, `extend`

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]
cities2 = ["New York", "Boston", "Chicago"]

cities3 = cities + cities2
print(cities3)

cities.extend(cities2)
print(cities)
```

Search: Is this **in** the list or **not in** the list?

Membership operators

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]

if "Montreal" in cities:
    print("Montreal is in the list")

if "New York" not in cities:
    print("New York is not in the list")
```


`len()`, `min()`, `max()`, `sum()`

```
numbers = [1, 2, 3, 4, 5]

print(len(numbers))      # 5
print(min(numbers))      # 1
print(max(numbers))      # 5
print(sum(numbers))      # 15
```

<https://docs.python.org/3/library/functions.html>

Loop over list

1. `for`

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]  
  
for city in cities:  
    print(city)
```

2. `len()` & `range()`

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]  
length = len(cities)  
  
for i in range(length):  
    print(cities[i])
```

Sort list - `sort`

```
cities = ["Montreal", "Toronto", "Vancouver", "Detroit"]  
  
cities.sort()  
print(cities)  
  
cities.sort(reverse=True)  
print(cities)
```



Namebook

- prompt the user to enter a name
- if the name exists in the namebook, print "Name found"
- if the name not exist, add the name to the book and print "Name added"
- print the updated list of names

```
Enter a name: john  
Name added  
["Alice", "Bob", "Charlie", "John"]
```

```
Enter a name: alice  
Name found  
["Alice", "Bob", "Charlie", "John"]
```

dictionary: a collection of key-value pairs

```
cities = {  
    "key": "value",  
    "name": "Montreal",  
    "state": "QC",  
    "country": "CA"  
}
```

<https://docs.python.org/3/tutorial/datastructures.html#dictionaries>

Access item in dict using key

```
cities = {  
    "name": "Montreal",  
    "state": "QC",  
    "country": "CA"  
}  
print(cities["name"])      # Montreal  
print(cities["state"])     # QC  
print(cities["country"])   # CA  
  
# Update  
cities["name"] = "New York"  
  
# Add  
cities["continent"] = "NA"
```

Delete item from dict - `del`

```
cities = {  
    "name": "Montreal",  
    "state": "QC",  
    "country": "CA"  
}  
  
del cities["state"]  
  
print(cities)
```

Join two dicts (unique keys) - `|`, `update`

```
cities = {
    "name": "Montreal",
    "state": "QC",
    "country": "CA"
}

cities2 = {
    "name2": "New York",
    "state2": "NY",
    "country2": "US"
}

cities3 = cities | cities2
print(cities3)

cities.update(cities2)
print(cities)
```


`keys()`, `values()`, `items()` return list-like objects

```
cities = {
    "name": "Montreal",
    "state": "QC",
    "country": "CA"
}

print(cities.keys())
# output: dict_keys(['name', 'state', 'country'])

print(cities.values())
# output: dict_values(['Montreal', 'QC', 'CA'])

print(cities.items())
# output: dict_items([('name', 'Montreal'), ('state', 'QC'), ('country', 'CA')])
```

Search: **in**, **not in**

```
cities = {  
    "name": "Montreal",  
    "state": "QC",  
    "country": "CA"  
}  
  
if "name" in cities.keys():  
    print("name is in the keys of the dict")  
  
if "Detroit" not in cities.values():  
    print("Detroit is not in the values of the dict")
```

`len()`, `min()`, `max()`, `sum()`

```
numbers = {  
    "a": 1,  
    "b": 2,  
    "c": 3,  
    "d": 4,  
    "e": 5  
}  
  
print(len(numbers))      # 5  
print(min(numbers))      # a (key)  
print(max(numbers.values())) # 5 (value)  
print(sum(numbers.values())) # 15
```

loop over dict

```
cities = {  
    "name": "Montreal",  
    "state": "QC",  
    "country": "CA"  
}  
  
for key in cities: # equivalent to cities.keys()  
    print(key)  
  
for key in cities: # equivalent to cities.keys()  
    print(key, cities[key])  
  
for value in cities.values():  
    print(value)  
  
for key, value in cities.items():  
    print(key, value)
```

Sort dict - sorted

```
cities = {  
    "name": "Montreal",  
    "state": "QC",  
    "country": "CA"  
}  
  
sorted(cities.items())
```



Count words

- Given a list of words, write a program that creates a dictionary where the keys are the words and the values are the frequency of each word.

```
Enter a sentence: the quick brown fox jumps over the lazy dog  
{'the': 2, 'quick': 1, 'brown': 1, 'fox': 1, 'jumps': 1, 'over': 1, 'lazy': 1, 'dog': 1}
```

	List	Dict
Access	<code>cities[0]</code>	<code>cities["key"]</code>
Update	<code>cities[0] = "new item"</code>	<code>cities["existing key"] = "new value"</code>
Add	<code>cities.append("new item")</code>	<code>cities["new key"] = "new value"</code>
Delete	<code>del cities[0]</code>	<code>del cities["key"]</code>

	List	Dict
Join	<code>cities3 = cities1 + cities2</code>	<code>cities3 = cities1 cities2</code>
Search	<code>"item" in cities</code>	<code>"key" in cities.keys()</code> <code>"value" in cities.values()</code>
Loop	<code>for item in cities:</code>	<code>for key in cities.keys():</code> <code>for value in cities.values():</code> <code>for item in cities.items():</code>
Sort	<code>cities.sort()</code>	<code>sorted(cities)</code>

More on data structures

- list of dict: `[{...}, {...}, {...}]`
- list of list: `[[...], [...], [...]]`
- dict of list: `{"key1": [...], "key2": [...], "key3": [...]}`
- dict of dict: `{"key1": {...}, "key2": {...}, "key3": {...}}`

list of dict

```
cities = [  
    {"name": "Montreal", "state": "QC", "country": "CA"},  
    {"name": "Toronto", "state": "ON", "country": "CA"},  
    {"name": "Vancouver", "state": "BC", "country": "CA"},  
    {"name": "Detroit", "state": "MI", "country": "US"}  
]
```

`cities`

```
[  
    {...}, # index 0  
    {...}, # index 1  
    {...}, # index 2  
    {...}  # index 3  
]
```

`cities[0]`

```
{"name": "Montreal", "state": "QC", "country": "CA"}
```

`cities[0]["state"]`

```
"QC"
```

list of dict

```
cities = [  
    {"name": "Montreal", "state": "QC", "country": "CA"},  
    {"name": "Toronto", "state": "ON", "country": "CA"},  
    {"name": "Vancouver", "state": "BC", "country": "CA"},  
    {"name": "Detroit", "state": "MI", "country": "US"}  
]  
print(type(cities))  
  
for city in cities: # [{...}, {...}, {...}, {...}]  
    print(type(city))  
    print(city["name"], city["state"], city["country"])
```

list of list

```
cities = [  
    ["Montreal", "QC", "CA"],  
    ["Toronto", "ON", "CA"],  
    ["Vancouver", "BC", "CA"],  
    ["Detroit", "MI", "US"]  
]  
print(type(cities))  
  
for city in cities: # [[...], [...], [...], [...]]  
    print(type(city))  
    print(city[0], city[1], city[2])  
  
# change the state of Montreal to NY  
cities[0][1] = "NY"  
print(cities[0][1])
```

dict of list

```
cities = {
    "name": ["Montreal", "Toronto", "Vancouver", "Detroit"],
    "state": ["QC", "ON", "BC", "MI"],
    "country": ["CA", "CA", "CA", "US"]
}
print(type(cities))

for col, rows in cities.items():
    print(type(rows))

    for row in rows:
        print(col + ": " + row)

# change the state of Montreal to NY
cities["state"][0] = "NY"
print(cities["state"][0])
```

dict of dict

```
cities = {
    "Montreal": {"state": "QC", "country": "CA"},
    "Toronto": {"state": "ON", "country": "CA"},
    "Vancouver": {"state": "BC", "country": "CA"},
    "Detroit": {"state": "MI", "country": "US"}
}
print(type(cities))

for city, info in cities.items():
    print(type(info))
    print(city, info["state"], info["country"])

# change the state of Montreal to NY
cities["Montreal"]["state"] = "NY"
print(cities["Montreal"]["state"])
```

🤔 How to access the following items?

```
cities = [  
    {"name": "Montreal", "state": "QC", "country": "CA"},  
    {"name": "Toronto", "state": "ON", "country": "CA"},  
    {"name": "Vancouver", "state": "BC", "country": "CA"},  
    {"name": "Detroit", "state": "MI", "country": "US"}  
]
```

- {"name": "Vancouver", "state": "BC", "country": "CA"}
- "Vancouver"
- "Montreal"


```
cities = {  
    "name": ["Montreal", "Toronto", "Vancouver", "Detroit"],  
    "state": ["QC", "ON", "BC", "MI"],  
    "country": ["CA", "CA", "CA", "US"]  
}
```

- ["Montreal", "Toronto", "Vancouver", "Detroit"]
- "Montreal"

```
cities = {
    "location": {
        "Montreal": {"state": "QC", "country": "CA"},
        "Toronto": {"state": "ON", "country": "CA"},
        ...
    },
    "stats": {
        "Montreal": [
            {"year": 2013, "population": 2000000, "area": 431.5},
            {"year": 2014, "population": 1980000, "area": 431.5}
        ],
        "Toronto": [
            {"year": 2013, "population": 2800000, "area": 630.2},
            ...
        ]
    }
}
```

- {"year": 2013, "population": 2000000, "area": 431.5}
- "QC"
- 2000000



1. **Team up! (Write down the email addresses; 4-5 members)**
2. Pseudocode the program (`main`) and functions together
3. Divide the tasks (functions) among team members
4. Complete the assigned functions individually
5. Complete `main` together

```
Enter a city name: new york  
New york is not available in data
```

```
Enter a city name: montreal  
name: Montreal  
state: QC
```

```
...  
Total number of keys: 8
```