

```
1 # A "dictionary" is a "data structure" -- a way of organizing many values into
2 # one variable. A list is another type of data structure. Data structures
3 # typically allow us to add, remove, and find values in them. Whereas the values
4 # of a list are associated with a position (an integer index), values of a
5 # dictionary are associated with a "key".
6
7 # A Python dictionary is denoted with { }, similar to a list denoted with [ ].
8 # Instead of individual values separated by commas, we give a list of
9 # "key-value pairs". The "key" of each pair is something easy to remember,
10 # through which the dictionary can provide access to the associated value.
11
12         # This is a key : this is a value
13 contacts = {"Neal Terrell" : "562-123-4567",
14             "Anthony Giacalone" : "714-987-6543",
15             "Mehrdad Aliasgari" : "310-678-9012"}
16
17 # To retrieve a value, we "index" the dictionary with the associated key.
18 print("Neal's phone number is " + contacts["Neal Terrell"])
19 print("Mehrdad's phone number is " + contacts["Mehrdad Aliasgari"])
20
21 # Keys have to match exactly; it is a run-time error (semantic error) to index
22 # a dictionary with a key that does not exist.
23
24 # This line causes an error:
25 #print(contacts["Neal terrell"])
26
27 # We can add new items to the dictionary
28 contacts["Joshua Hayter"] = "213-555-5555"
29 contacts["Joshua Hayter"] = "213-555-1111"
30 print(contacts["Joshua Hayter"])
31 print()
32
33 # We can loop through a dictionary using a for loop, just like a list... but
34 # what we get out are the keys of the dictionary.
35 for name in contacts:
36     print(name + ": " + contacts[name])
37
38
39 # We can also loop through the dictionary's values only, using the
40 # .values() method.
41 for phone_number in contacts.values():
42     # Do something with the phone numbers.
43     print(phone_number)
44
45
```

```
1 # Rework the Baseball example to use a dictionary as the main data structure,
2 # instead of a list. Each player's tuple (a value) will be associated with
3 # the player's name (a key).
4
5 def read_players(file_name):
6     first_line = True
7     results = {} # create an empty dictionary
8     for line in open(file_name):
9         if not first_line:
10             # work here
11             split = line.split(",")
12             name = split[0].strip(' ')
13             player = (name, split[1].strip(' '), int(split[9].strip(' ')), \
14                     int(split[11].strip(' ')), int(split[19].strip(' ')), \
15                     float(split[21].strip(' ')))
16
17             # insert an association between the player's name
18             # and their statistics in the dictionary
19             results[name] = player
20         else:
21             first_line = False
22     return results
23
24 def main():
25     all_players = read_players("baseball_players.csv")
26     # Reminder: all_players is now a dictionary, with keys that are player names.
27     choice = 0
28     while choice != 4:
29         print("1. Search for player")
30         print("2. Search for team")
31         print("3. Find max homeruns")
32         print("4. Quit")
33
34         choice = int(input("Enter a choice: "))
35         if choice == 1:
36             search_for_player(all_players)
37         elif choice == 2:
38             search_for_team(all_players)
39         elif choice == 3:
40             find_max_hrs(all_players)
41
42 def print_player(player):
43     (player_name, team, hr, rbi, sb, avg) = player
44     print("{0} ({1}). {2} HR, {3} RBI, {4} SB, {5:0.3f} AVG"\
45           .format(player_name, team, hr, rbi, sb, avg))
46
47 def find_max_hrs(all_players):
48     max_hr = 0
49     max_name = ''
50     # Since we don't know the names of the players we are examining, we use
51     # .values() to loop over all players in the dictionary.
52     for player in all_players.values():
```

```
53     (player_name, team, hr, rbi, sb, avg) = player
54     if hr > max_hr:
55         max_hr = hr
56         max_name = player
57
58     print_player(max_name)
59
60 def search_for_team(all_players):
61     team_name = input("Enter a team name: ")
62     for player in all_players.values():
63         (player_name, team, hr, rbi, sb, avg) = player
64         if team_name == team:
65             print_player(player)
66
67
68 def search_for_player(all_players):
69     name = input("Enter a player's name: ")
70
71     # This is the way to search a list for the given player name. It
72     # is now unnecessary.
73     #for player in all_players:
74     #    (player_name, team, hr, rbi, sb, avg) = player
75     #    if name == player_name:
76     #        print_player(player)
77     #        break
78
79     # Instead, we just check to see if the dictionary contains the player
80     # name as a key. The "in" keyword serves this purpose.
81     if name in all_players:
82         player = all_players[name]
83         print_player(player)
84
85
86
87 main()
88
```

```
1 # Rework the Baseball example AGAIN, so that each PLAYER is also a dictionary
2 # instead of a tuple.
3
4 def read_players(file_name):
5     first_line = True
6     results = {} # create an empty dictionary
7     for line in open(file_name):
8         if not first_line:
9             # work here
10            split = line.split(",")
11
12            # Instead of packing the line into a tuple, we create a dictionary
13            # for the player's data. Later, instead of having to remember the
14            # ORDER of each stat in the player tuple, we just remember the name
15            # of the stat in the dictionary. Much easier!
16            player = {
17                "name" : split[0].strip(' '),
18                "team" : split[1].strip(' '),
19                "hr" : int(split[9].strip(' ')),
20                "rbi" : int(split[11].strip(' ')),
21                "sb" : int(split[19].strip(' ')),
22                "avg" : float(split[21].strip(' '))
23            }
24
25            results[player["name"]] = player
26        else:
27            first_line = False
28    return results
29
30 def main():
31     all_players = read_players("baseball_players.csv")
32     # Reminder: all_players is now a dictionary, with keys that are player names.
33     choice = 0
34     while choice != 4:
35         print("1. Search for player")
36         print("2. Search for team")
37         print("3. Find max homeruns")
38         print("4. Quit")
39
40         choice = int(input("Enter a choice: "))
41         if choice == 1:
42             search_for_player(all_players)
43         elif choice == 2:
44             search_for_team(all_players)
45         elif choice == 3:
46             find_max_hrs(all_players)
47
48 def print_player(player):
49     print("{0} ({1}). {2} HR, {3} RBI, {4} SB, {5:0.3f} AVG"\
50           .format(player["name"], player["team"], player["hr"],\
51                   player["rbi"], player["sb"], player["avg"]))
52
```

```
53 def find_max_hrs(all_players):
54     max_hr = -1
55
56     # Since we don't know the names of the players we are examining, we use
57     # .values() to loop over all players in the dictionary.
58     for player in all_players.values():
59         if player["hr"] > max_hr:
60             max_hr = player["hr"]
61             max_name = player["name"]
62
63
64     print("The most homeruns was {0} by {1}".format(max_hr, max_name))
65
66 def search_for_team(all_players):
67     team_name = input("Enter a team name: ")
68     for player in all_players.values():
69         if player["team"] == team_name:
70             print_player(player)
71
72
73 def search_for_player(all_players):
74     name = input("Enter a player's name: ")
75     # We just check to see if the dictionary contains the player
76     # name as a key. The "in" keyword serves this purpose.
77     if name in all_players:
78         player = all_players[name]
79         print_player(player)
80
81
82 main()
83
```