Dictionaries (Chapter 6)

Dictionaries are like lists but use strings called *keys* rather than a numeric index to refer to the stored items.

In [2]:
```
exam_grades = {'alice': 100,'bob': 95,'charlie': 98}

# retrieve values by their associated key
print(f"Alice received a {exam_grades['alice']} on the exam")

# change values
exam_grades['alice'] = 105

# add items to the dictionary (same as changing a value)
exam_grades['dan'] = 85
```

```
Alice received a 100 on the exam
```

In [3]:
```
# start with an empty dictionary and add items (key-value pairs)
alien_0 = {}
alien_0['color'] = 'green'
alien_0['points'] = 5
print(alien_0)
```

```
{'color': 'green', 'points': 5}
```

Alien Code Example

In [4]:
```
alien_0 = {'x_position': 0, 'y_position': 25, 'speed': 'medium'}
print(f"Original position: {alien_0['x_position']}")

# Move the alien to the right.
# Determine how far to move the alien based on its current speed.
if alien_0['speed'] == 'slow':
    x_increment = 1
elif alien_0['speed'] == 'medium':
    x_increment = 2
else:
    # This must be a fast alien.
    x_increment = 3

# The new position is the old position plus the increment.
alien_0['x_position'] = alien_0['x_position'] + x_increment

print(f"New position: {alien_0['x_position']}")
```

```
Original position: 0
New position: 2
```

In [5]:
```
# removing items from a dictionary
favorite_foods = {
    'alice': 'apples',
    'bob': 'bananas',
    'charlie': 'cucumbers'
}
# remove charlie
del favorite_foods['charlie']

print(favorite_foods)
```

```
{'alice': 'apples', 'bob': 'bananas'}
```

Using `get()` to Access Values

In [6]:
```python
# trying to access an item that doesn't exist causes an error
favorite_foods = {
    'alice': 'apples',
    'bob': 'bananas',
    'charlie': 'cucumbers'
}
print(f"Dan's favorite food is {favorite_foods['dan']}")
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[6], line 7
      1 # trying to access an item that doesn't exist causes an error
      2 favorite_foods = {
      3     'alice': 'apples',
      4     'bob': 'bananas',
      5     'charlie': 'cucumbers'
      6 }
----> 7 print(f"Dan's favorite food is {favorite_foods['dan']}")

KeyError: 'dan'
```

In [9]:
```python
# use get() to avoid an error and provide a default value instead
print(f"Dan's favorite food is {favorite_foods.get('dan','unknown')}")
```

```
Dan's favorite food is unknown
```

Looping Through a Dictionary

In [2]:
```
favorite_foods = {
    'alice': 'apples',
    'bob': 'bananas',
    'charlie': 'cucumbers'
}
# print everyone's favorite foods
for name,food in favorite_foods.items():
    print(f"{name.title()}'s favorite food is {food}.")

print("\n")
# just print everyone's names
for name in favorite_foods.keys():
    print(name.title())

print("\n")
# just print the foods
for name in favorite_foods.values():
    print(name.title())

print("\n")
# looping through the dictionary directly gives the keys
for name in favorite_foods:
    print(name.title())
```

```
Alice's favorite food is apples.
Bob's favorite food is bananas.
Charlie's favorite food is cucumbers.


Alice
Bob
Charlie


Apples
Bananas
Cucumbers


Alice
Bob
Charlie
```

In [11]:
```python
favorite_languages = {
    'jen': 'python',
    'sarah': 'c',
    'edward': 'ruby',
    'phil': 'python',
    }

friends = ['phil', 'sarah']
for name in favorite_languages.keys():
    print(name.title())

    if name in friends:
        language = favorite_languages[name].title()
        print(f"\t{name.title()}, I see you love {language}!")
```

```
Jen
Sarah
        Sarah, I see you love C!
Edward
Phil
        Phil, I see you love Python!
```

In [3]:
```python
# use sorted() loop in a particular order
for name in sorted(favorite_foods, reverse=True):
    print(name.title())
```

```
Charlie
Bob
Alice
```

In [11]:
```python
# use set() to find unique values
colors = ['red','red','blue','green','blue']
print(set(colors))

print("\n")

# use set() along with .values() to print unique values in a dictionary
favorite_foods = {
    'alice': 'apples',
    'bob': 'bananas',
    'charlie': 'apples' # duplicate value
}
print(set(favorite_foods.values()))

# TODO print each type of fruit you need to buy on a separate line using a loop
```

```
{'blue', 'green', 'red'}


{'bananas', 'apples'}
```

Nesting

In [12]:
```
# a list of dictionaries
users = [{'name':'Alice', 'email':'alice@gmail.com'},
         {'name':'Bob', 'email':'bob123@hotmail.com'},
         {'name':'Charlie','email':'charlz@yahoo.com'}]
# print all of the user names:
for user in users:
    print(user['name'])

# a dictionary of lists
course_roster={
    'section1': ['alice','bob','charlie'],
    'section2': ['dan','edward','frank']
}
# print all of the students
all_students = []
for students in course_roster.values():
    all_students += students
print(all_students)
```

```
Alice
Bob
Charlie
['alice', 'bob', 'charlie', 'dan', 'edward', 'frank']
```

In [13]:
```
# Make an empty list for storing aliens.
aliens = []

# Make 30 green aliens.
for alien_number in range(30):
    new_alien = {'color': 'green', 'points': 5, 'speed': 'slow'}
    aliens.append(new_alien)

for alien in aliens[:3]:
    if alien['color'] == 'green':
        alien['color'] = 'yellow'
        alien['speed'] = 'medium'
        alien['points'] = 10

# Show the first 5 aliens.
for alien in aliens[:5]:
    print(alien)
print("...")
```

```
{'color': 'yellow', 'points': 10, 'speed': 'medium'}
{'color': 'yellow', 'points': 10, 'speed': 'medium'}
{'color': 'yellow', 'points': 10, 'speed': 'medium'}
{'color': 'green', 'points': 5, 'speed': 'slow'}
{'color': 'green', 'points': 5, 'speed': 'slow'}
...
```

Dictionary in a Dictionary

In [14]:
```
users = {
    'aeinstein': {
        'first': 'albert',
        'last': 'einstein',
        'location': 'princeton',
        },

    'mcurie': {
        'first': 'marie',
        'last': 'curie',
        'location': 'paris',
        },

    }

for username, user_info in users.items():
    print(f"\nUsername: {username}")
    full_name = f"{user_info['first']} {user_info['last']}"
    location = user_info['location']

    print(f"\tFull name: {full_name.title()}")
    print(f"\tLocation: {location.title()}")
```

```
Username: aeinstein
        Full name: Albert Einstein
        Location: Princeton

Username: mcurie
        Full name: Marie Curie
        Location: Paris
```

Homework Problems

6-1. Person: Use a dictionary to store information about a person you know. Store their first name, last name, age, and the city in which they live. You should have keys such as `first_name`, `last_name`, and `city`. Print each piece of information stored in your dictionary.

In [21]:
```
friend = {
    'first_name':'Avery',
    'last_name': 'Williams',
    'city': 'ATL',
}

print(f"first name: {friend['first_name']}")
print(f"last name: {friend['last_name']}")
print(f"city: {friend['city']}")
print(f"{friend}")
```

```
first name: Avery
last name: Williams
city: ATL
{'first_name': 'Avery', 'last_name': 'Williams', 'city': 'ATL'}
```

6-2. Favorite Numbers: Use a dictionary to store people's favorite numbers. Think of five names, and use them as keys in your dictionary. Think of a favorite number for each person, and store each as a value in your dictionary. Print each person's name and their favorite number.

In [26]:
```
fav_num = {
    'Erin':5,
    'Avery':47,
    'Josh':1,
    'Leah':2,
    'Bob':7,
}

for friends,num in fav_num.items():
    print(f"{friends}'s favorite number is {num}.")
```

```
Erin's favorite number is 5.
Avery's favorite number is 47.
Josh's favorite number is 1.
Leah's favorite number is 2.
Bob's favorite number is 7.
```

6-3. Glossary: A Python dictionary can be used to model an actual dictionary. However to avoid confusion let's call it a glossary.

- Think of five programming words you've learned about in the previous chapters. Use these words as the keys in your glossary, and store their meanings as values.
- Print each word and its meaning as neatly formatted otuput. You might print the word followed by a colon and then its meaning, or print the word on one line and then print its meaning indented on a second line. Use the newline character (`\n`) to insert a blank line between each word-meaning pair in your output.

In [30]:
```
coding_glossary = {
    'list':'a collection of items in a particular order',
    'if statements':'allows the use of conditionals',
    'loop':'allows ability to filter through a inputs or a list',
    'slicing':'helps to divide a list',
    'sort':'helps you to organize a list',
}
for term,defin in coding_glossary.items():
    print(f"{term}; {defin}.")
```

```
list; a collection of items in a particular order.
if statements; allows the use of conditionals.
loop; allows ability to filter through a inputs or a list.
slicing; helps to divide a list.
sort; helps you to organize a list.
```

6-4. Glossary 2: Clean up the code from above by replacing your series of `print()` calls with a loop that runs through the dictionary's keys and values. When you're sure that your loop works, add five more Python terms to your glossary. When you run your program again, these new words and meanings should automatically be included in the output.

In [31]:
```
#already did loop, so just adding five more

coding_glossary = {
    'list':'a collection of items in a particular order',
    'if statements':'allows the use of conditionals',
    'loop':'allows ability to filter through a inputs or a list',
    'slicing':'helps to divide a list',
    'sort':'helps you to organize a list',
    'string':'stores letter values, cannot be used for mathematical operations',
    'conditionals':'==,<,>,<=,>=,!= not equal',
    'keys': 'the variable names for stuff in dictionaries',
    'values': 'stored in keys in dictionaries',
    'items': 'the combination of keys and values in a dictionary'
}
for term,defin in coding_glossary.items():
    print(f"{term}; {defin}.")
```

```
list; a collection of items in a particular order.
if statements; allows the use of conditionals.
loop; allows ability to filter through a inputs or a list.
slicing; helps to divide a list.
sort; helps you to organize a list.
string; stores letter values, cannot be used for mathematical operations.
conditionals; ==,<,>,<=,>=,!= not equal.
keys; the variable names for stuff in dictionaries.
values; stored in keys in dictionaries.
items; the combination of keys and values in a dictionary.
```

6-5. Rivers: Make a dictionary containing three major rivers and the country each river runs thorugh. One key-value pair might be `'nile': 'egypt'` .

- Use a loop to print a sentence about each river, such as *The Nile runs through Egypt*.
- Use a loop to print the name of each river included in the dictionary.
- Use a loop to print the name of each country included in the the dictionary.

In [34]:
```
rivers = {
    'Amazon':'Brazil',
    'Mekong':'Vietnam',
    'Yangtze':'China',
}
for river,location in rivers.items():
    print(f"The {river} river runs through {location}.")
```

```
The Amazon river runs through Brazil.
The Mekong river runs through Vietnam.
The Yangtze river runs through China.
```

6-6. Polling: Use the code in *favorite_languages.py* (page 97).

- Make a list of people who should take the favorite languages poll. Include some names that are already in the dictionary and some that are not.
- Loop through the list of people who should take the poll. If they have already taken the poll, print a message thanking them for responding. If they have not yet taken the poll, print a message inviting them to take the poll.

In [*]:
```
favorite_languages = {
    'jen':'python',
    'sarah':'c',
    'edward':'ruby',
    'phil':'python',
}
print(f"{favorite_languages}")
people = ['jen', 'josh', 'phil', 'kyra']
print(people)
#for name in favorite_languages.keys():
   # for person in people:
       # if person in people == name:
       #     print(f"{person} thank you for your response.")

       # else:
       #     print(f"{person} please consider taking the poll!")

#i = 0
#while i < 4:
#    if people(i) == name in favorite_languages.keys():
#        print('thanks!')
#i = i+1

print(f"thank you {favorite_languages.get('jen','unknown')} for taking the survey")
```

6-7. People: Start with the program you wrote for Exercise 6-1. Make two new dictionaries representing different people, and store all three dictionaries in a list called `people`. Loop through your list of people. As you loop through the list, print everything you know about each person.

In [ ]:
```

```

6-8. Pets: Make several dictionaries, where each dictionary represents a different pet. In each dictionary, include the kind of animal and the owner's name. Store these dictionaries in a list called `pets`. Next, loop through your list and as you do, print everything you know about each pet.

In [ ]:

6-9. Favorite Places: Make a dictionary called `favorite_places`. Think of three names to use as keys in the dictionary, and store one to three favorite places for each person. Loop through the dictionary, and print each person's name and their favorite place.

In [ ]:

6-10. Favorite Numbers: Modify your program from Exercise 6-2 so each person can have more than one favorite number. Then print each person's name along with their favorite numbers.

In [ ]:

6-11 Cities: Make a dictionary called `cities`. Use the names of three cities as keys in your dictionary. Create a dictionary of information about each city and include the country that the city is in, its approximate population, and one fact about that city. The keys for each city's dictionary should be something like `country`, `population`, and `fact`. Print the name of each city and all of the information you have stored about it.

In [ ]:

6-12. Extensions: We're now working with examples that are complex enough that they can be extended in any number of ways. Use one of the example programs from this chapter, and extend it by adding new keys and values, changing the context of the program or improving the formatting of the output.

In [ ]: