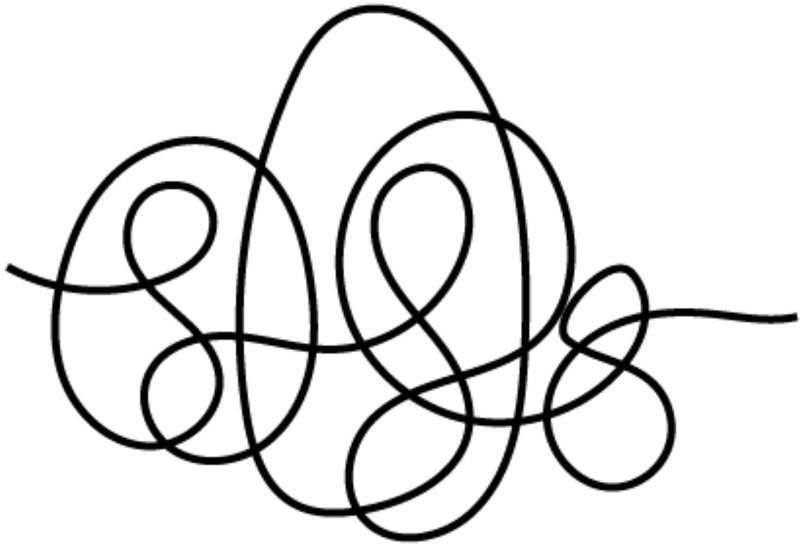
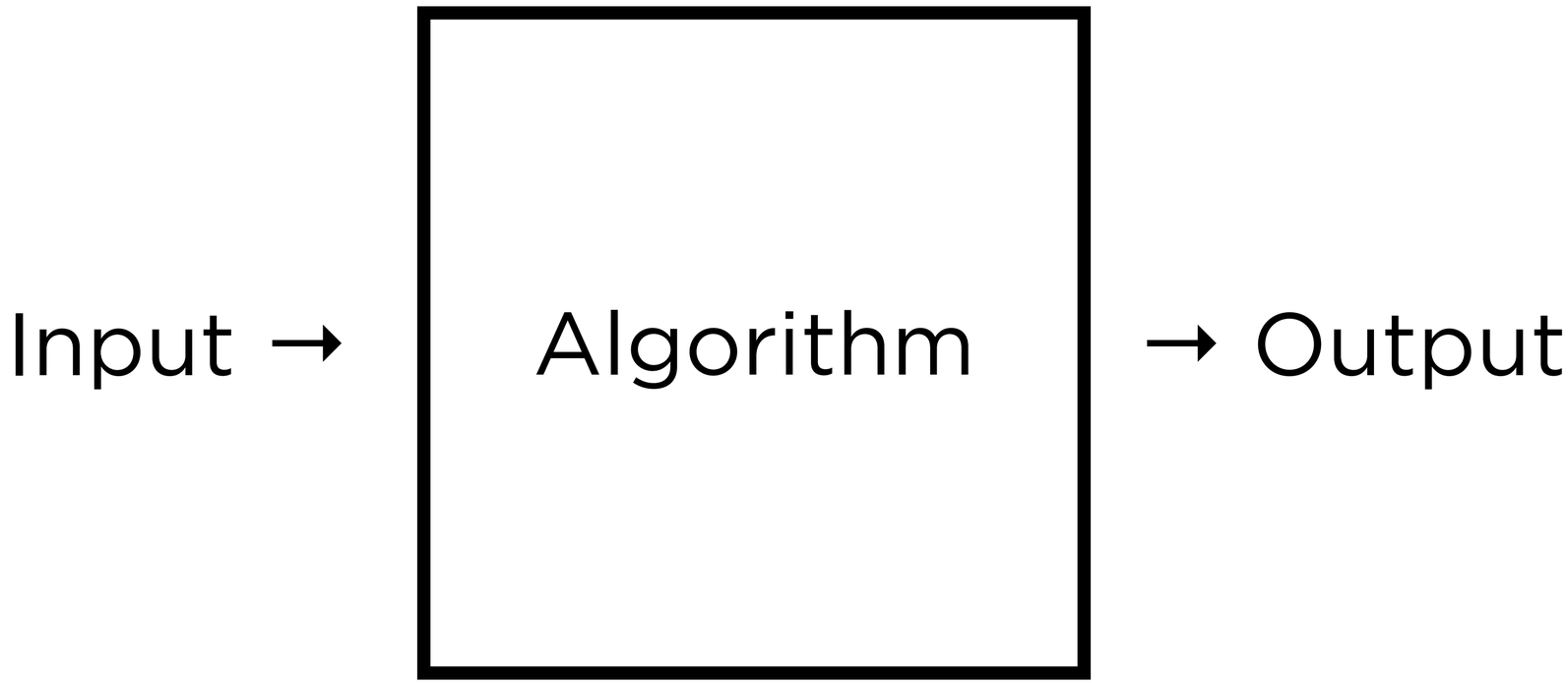


Lab 0

CS50 for MBAs


Input →
Algorithm
→ Output

- Work an example yourself
- Create an algorithm after working multiple examples
- Test your algorithm by hand
- Translate your algorithm to code
- Find bugs in your code by testing it

- Work an example yourself
- Create an algorithm after working multiple examples
- Test your algorithm by hand
- Translate your algorithm to code
- Find bugs in your code by testing it

carterzenke.me/lab

playback.py

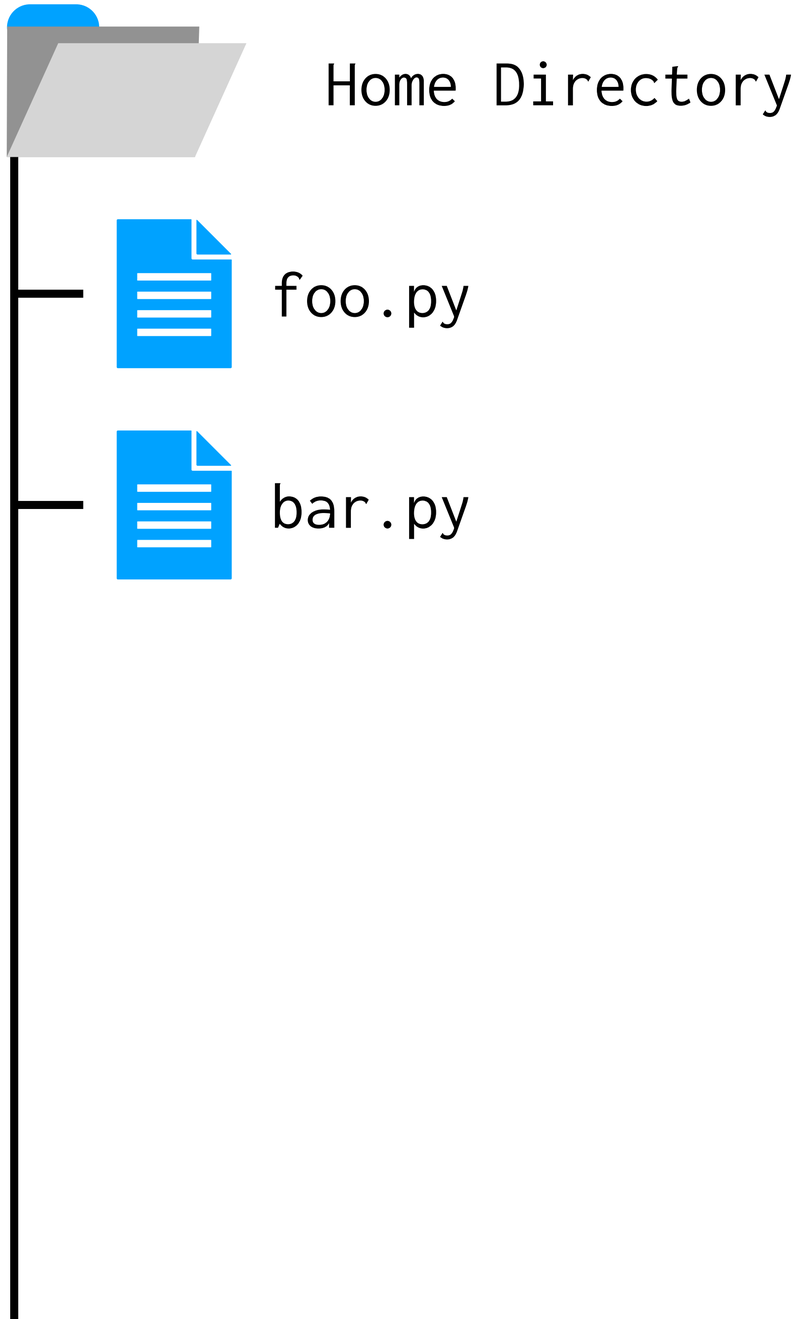
faces.py

twttr.py

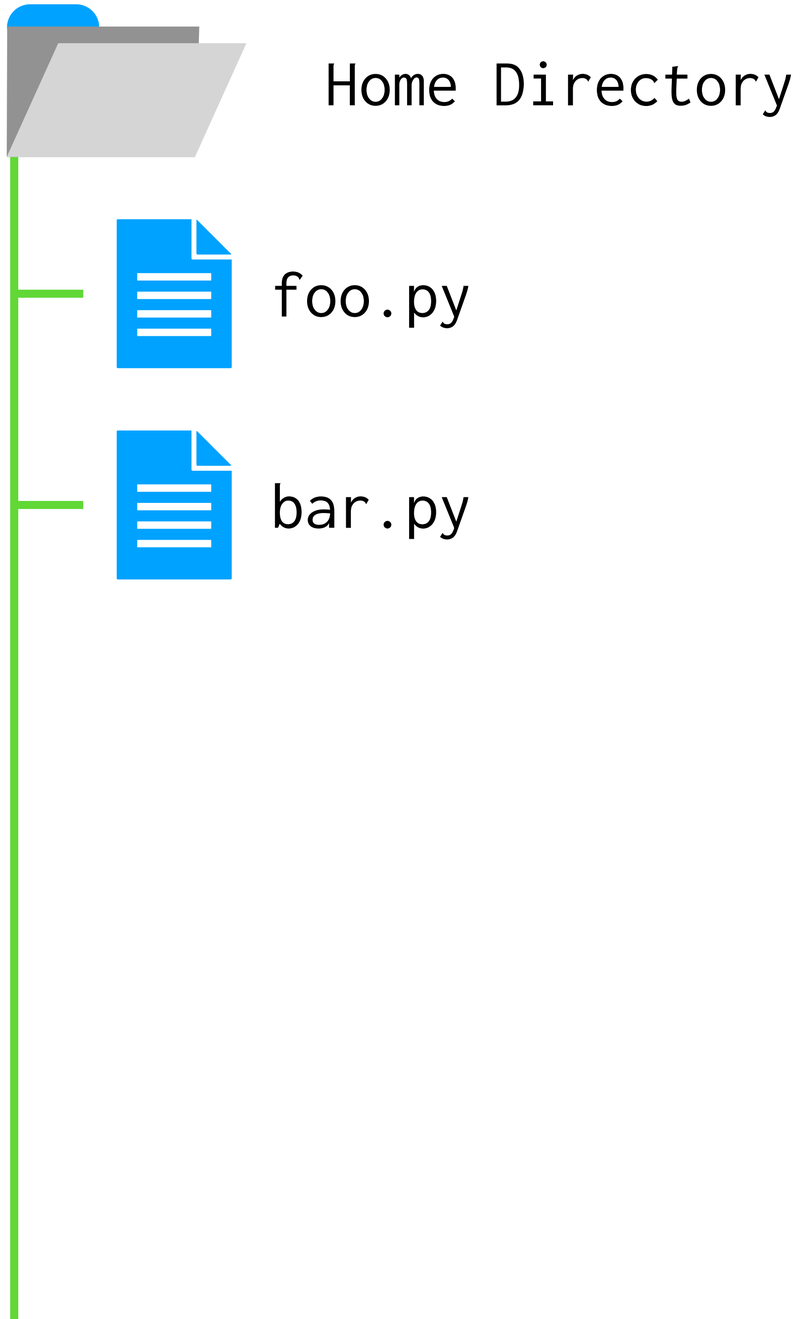
coke.py

nutrition.py

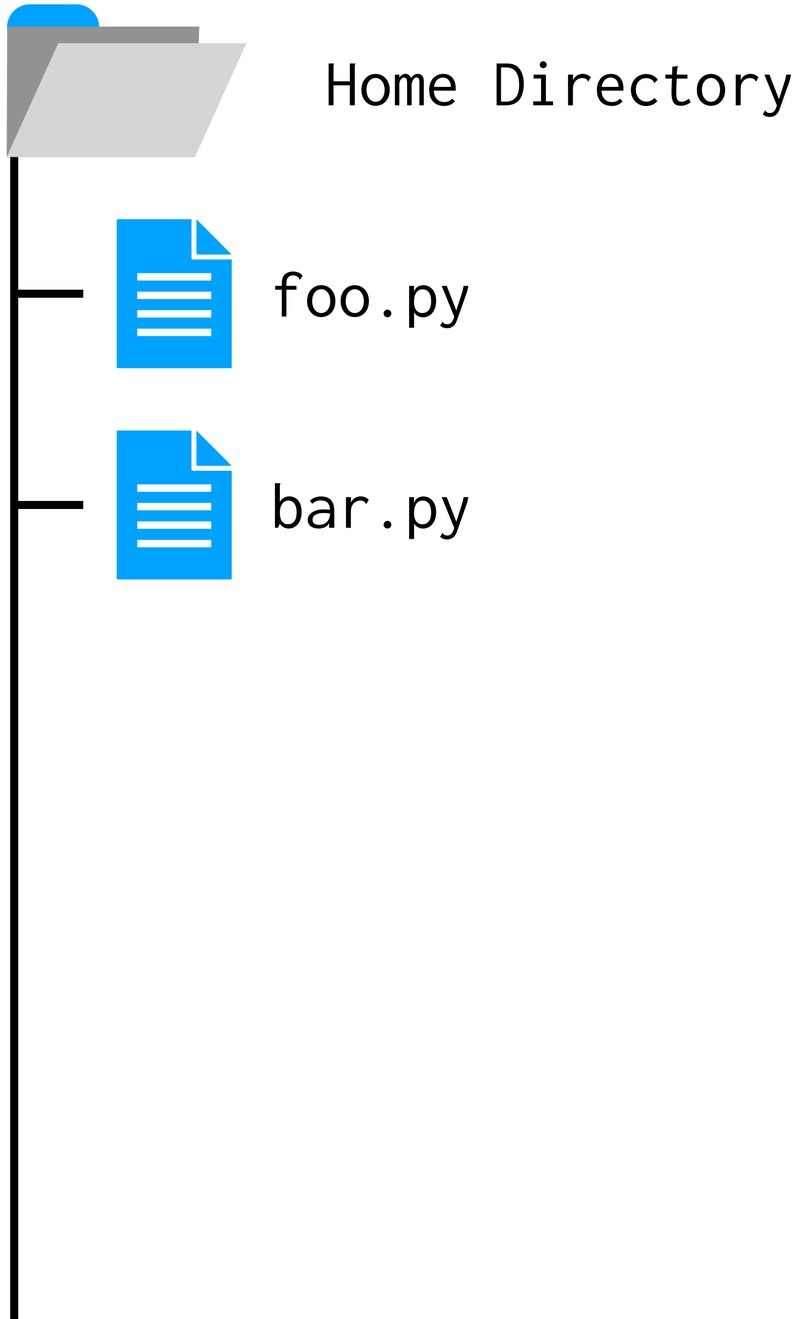

$


Home Directory
foo.py
bar.py

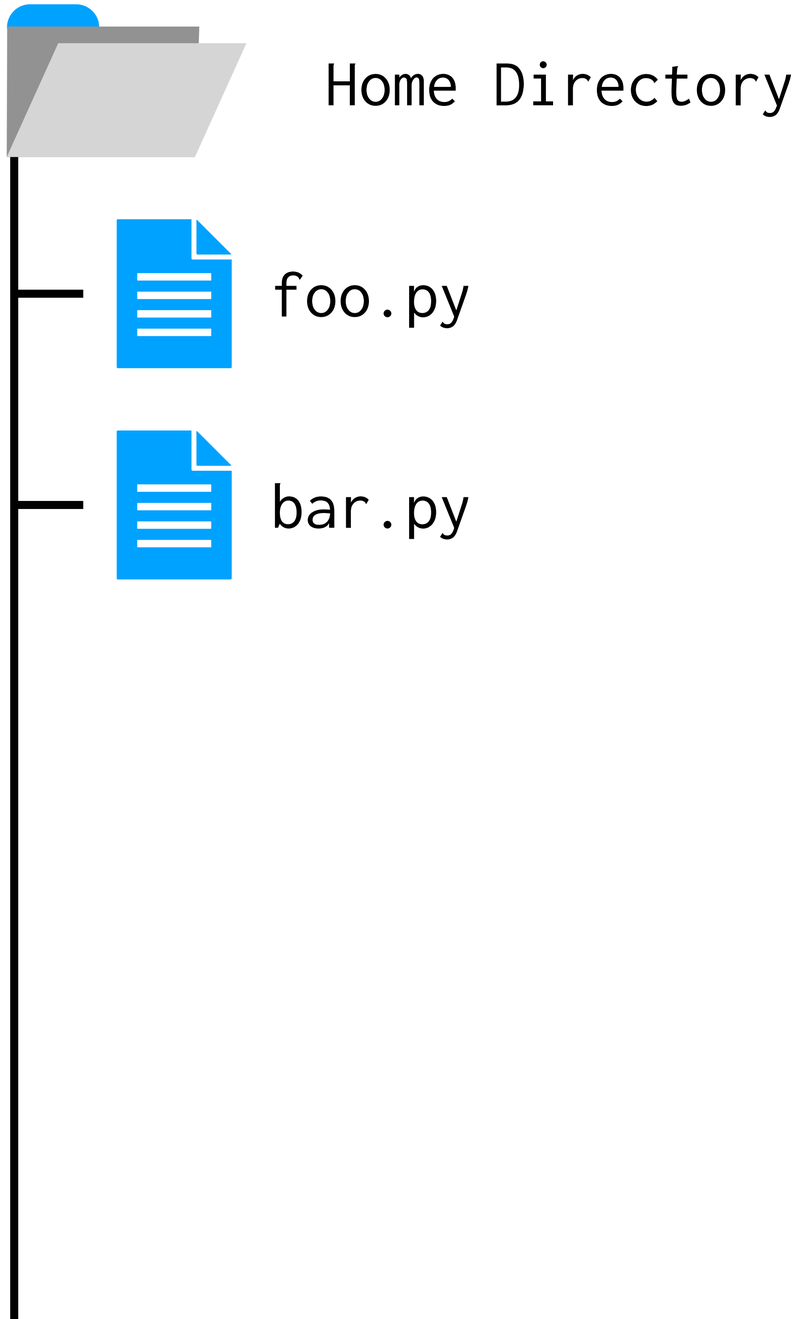

$


Home Directory
foo.py
bar.py

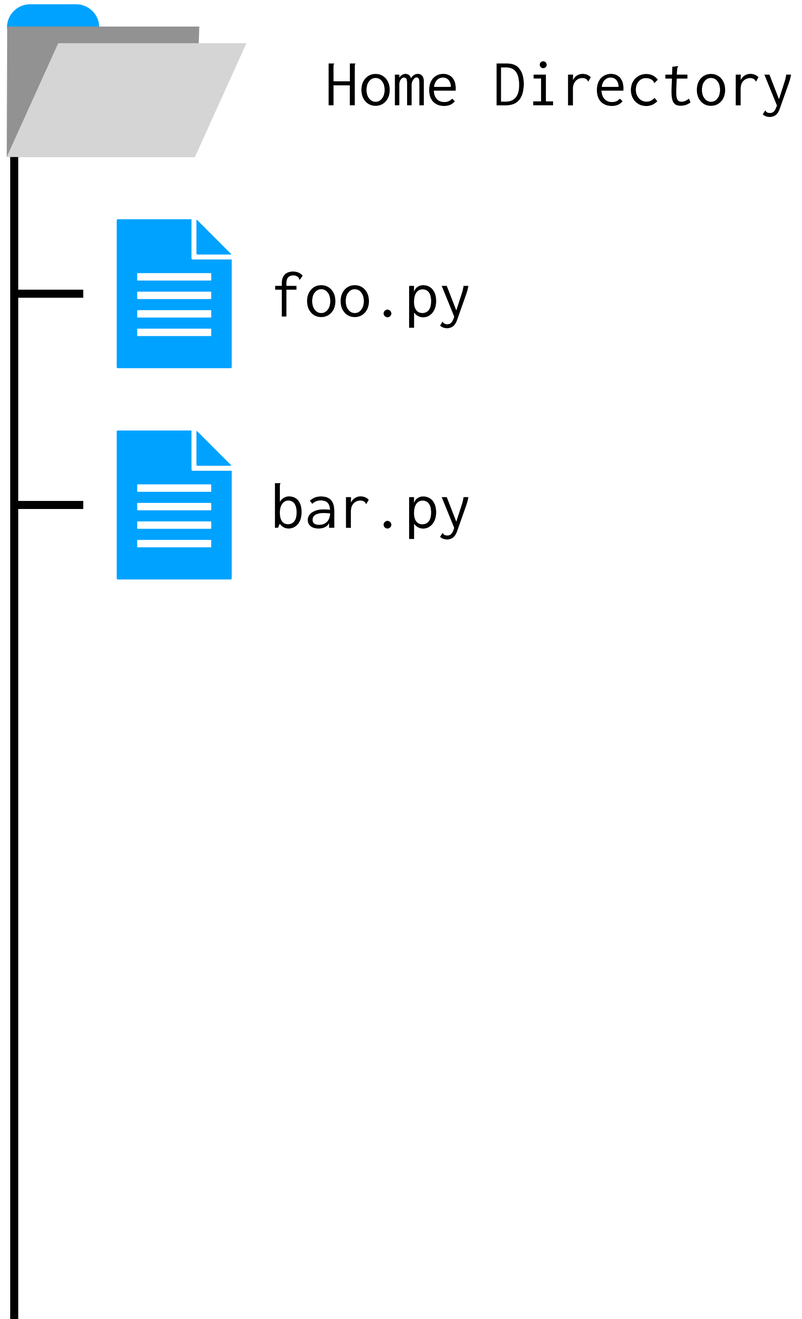
```
$ ls
```



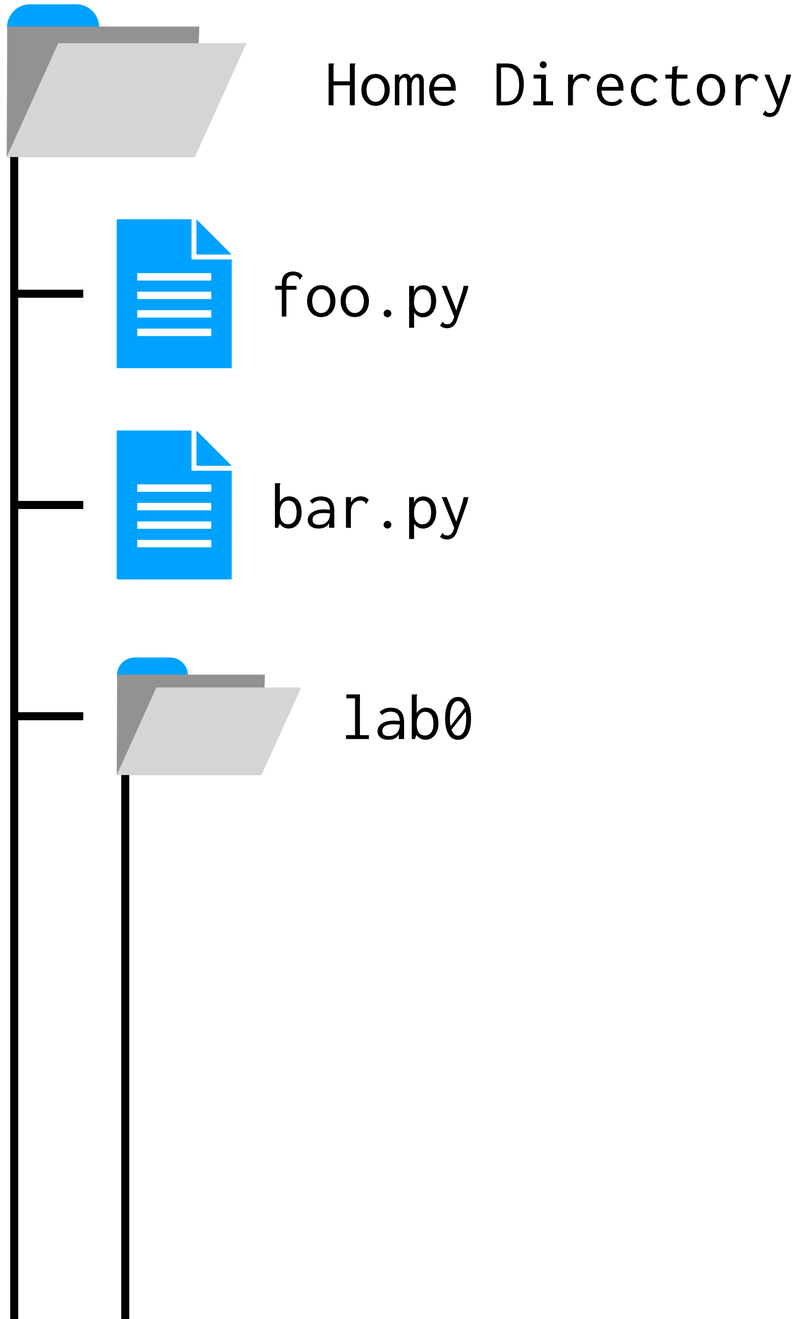
```
$ ls
foo.py  bar.py
```



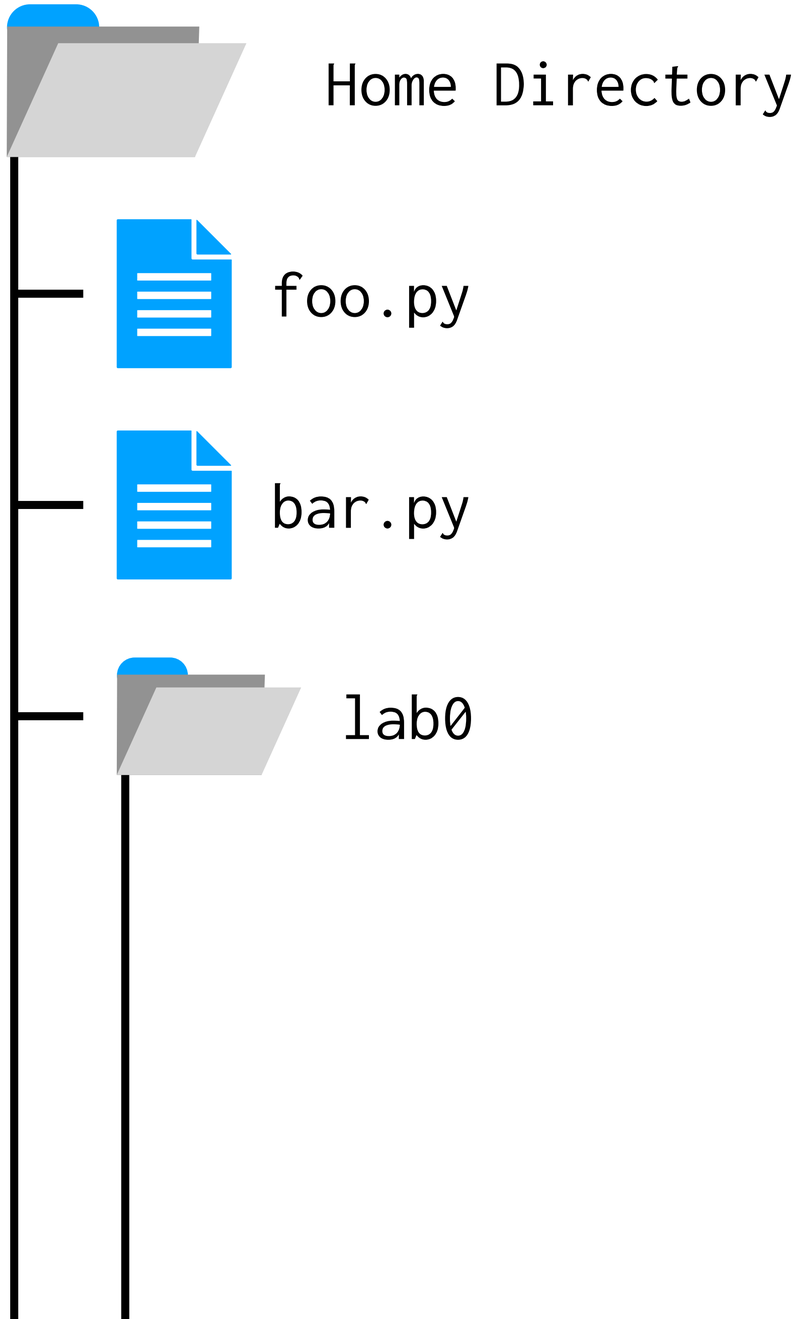
```
$  mkdir lab0
```



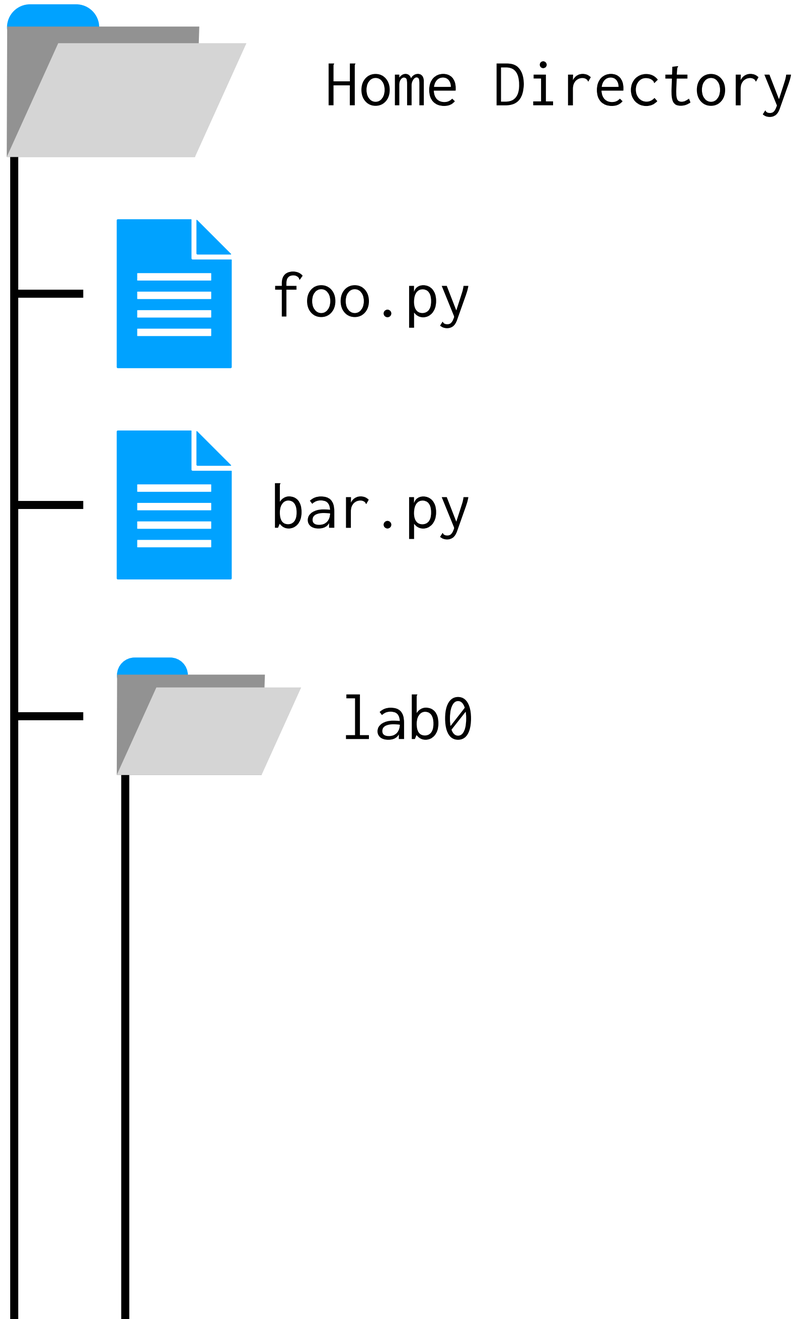
```
$  mkdir lab0
```



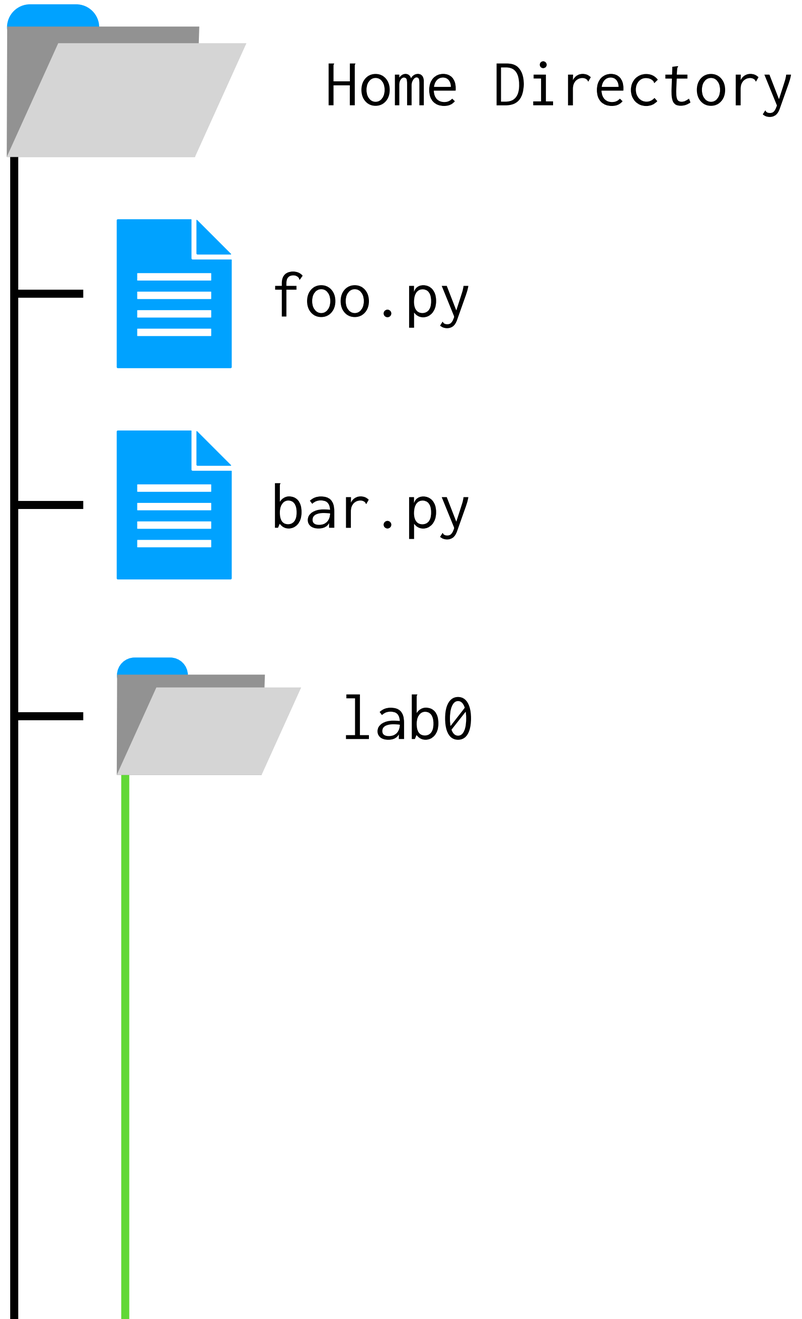
```
$ cd lab0
```



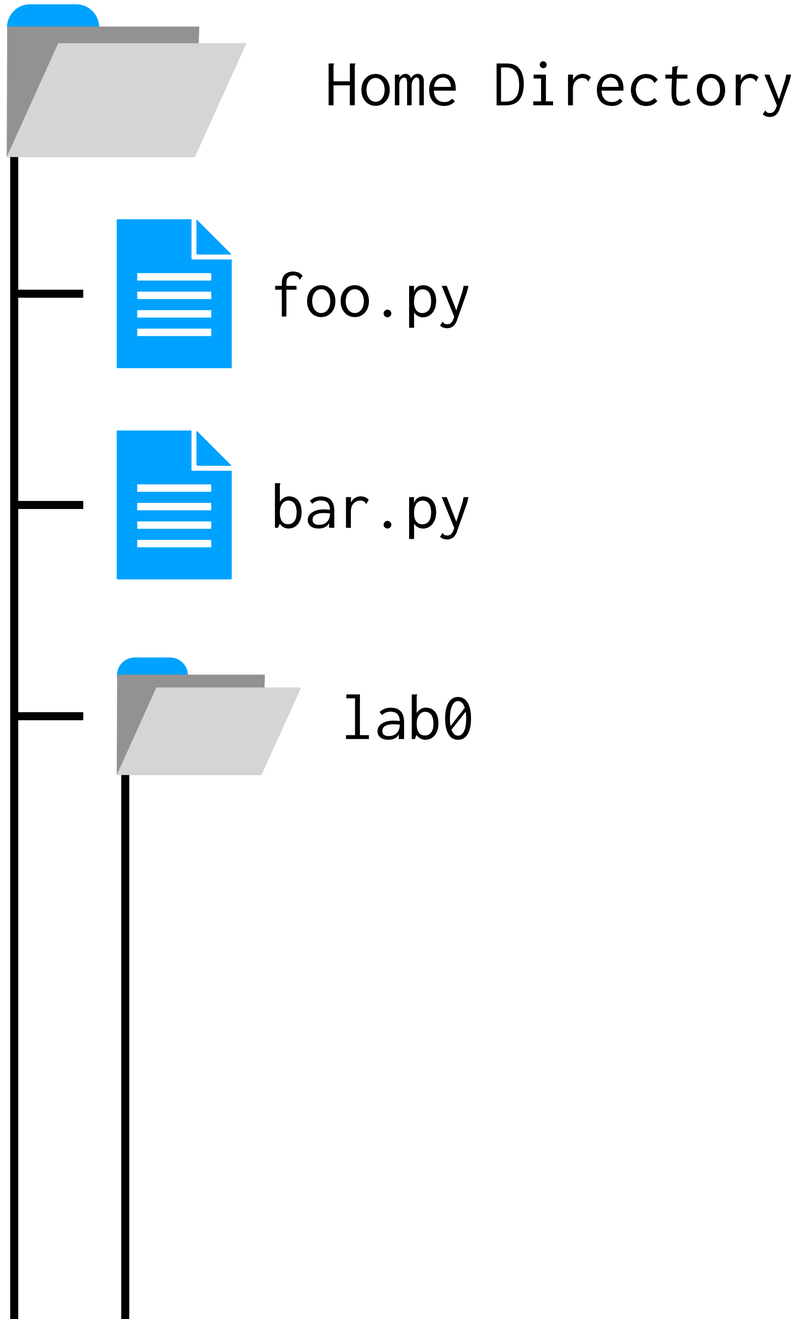
```
lab0/ $
```



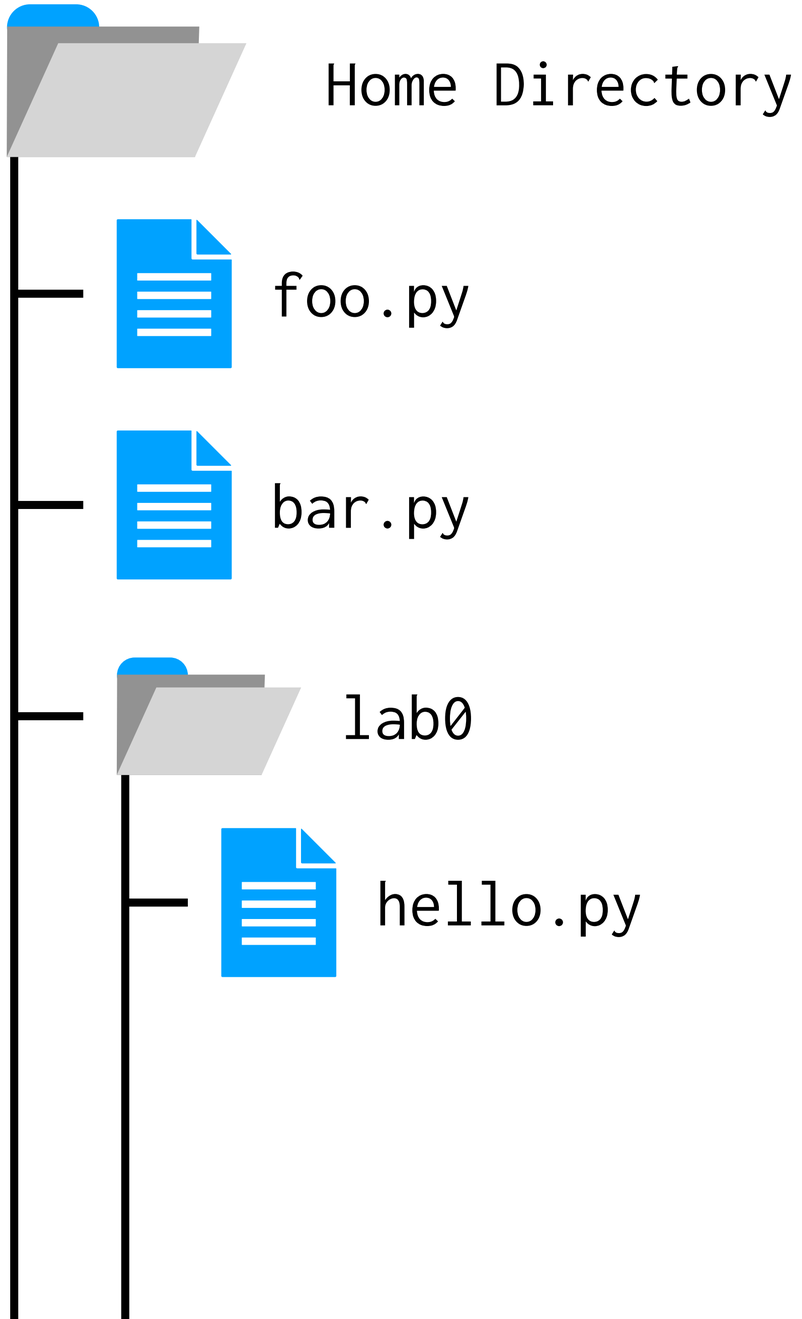
```
lab0/ $
```



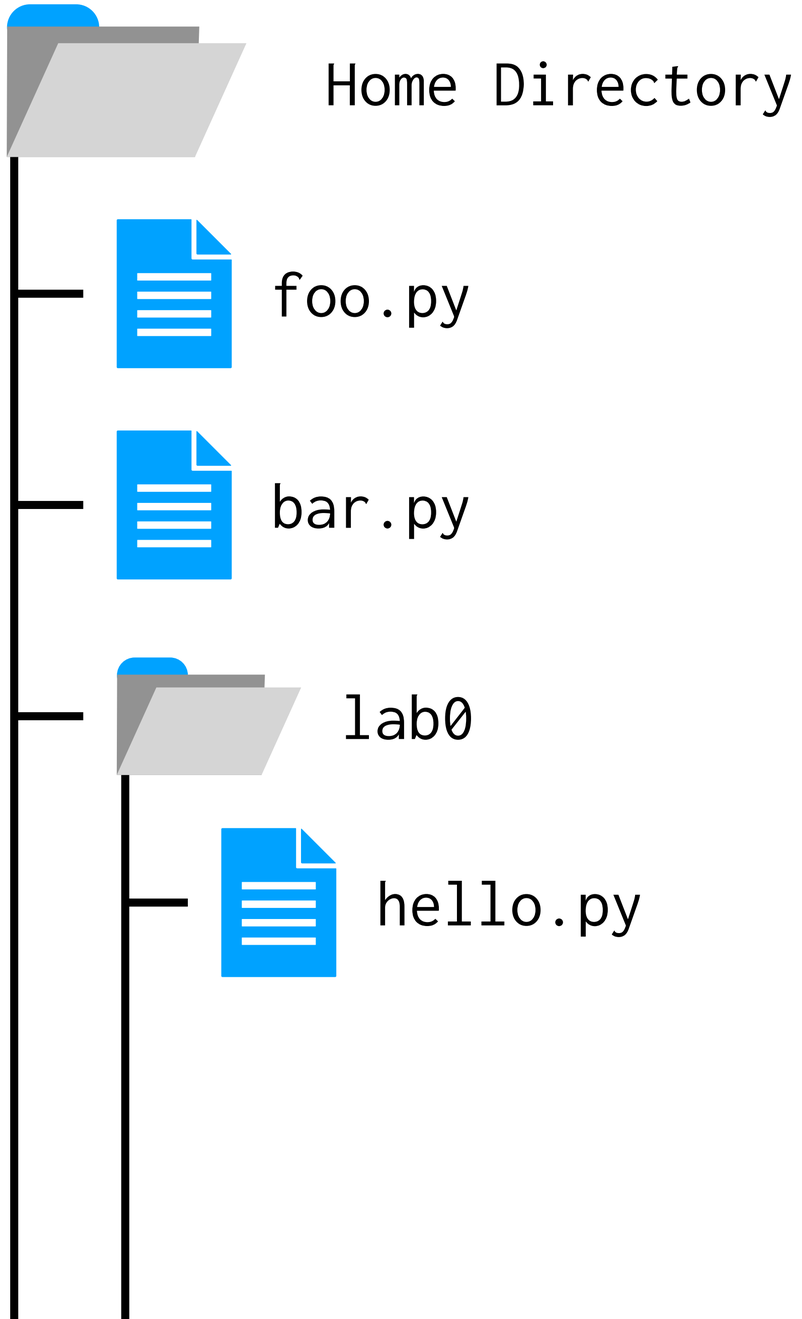
```
lab0/ $ code hello.py
```



```
lab0/ $ code hello.py
```



```
lab0/ $ code hello.py
```



Common terminal commands

- `ls` List files in current "directory" (folder)
- `mkdir NAME` Make a new directory called NAME
- `cd NAME` Change directory to one called NAME
- `cd ..` Change directory to the folder above
- `code NAME` Open a file named NAME

Common terminal commands

- `ls` List files in current "directory" (folder)
- `mkdir NAME` Make a new directory called NAME
- `cd NAME` Change directory to one called NAME
- `cd ..` Change directory to the directory above
- `code NAME` Open a file named NAME

hello.py

```
name = input()
print("Hello, " + name)
```

```
Carter
Hello, Carter
```

hello.py

```
name = input()
print("Hello, " + name)
```

Function

A set of instructions to perform a task.

hello.py

```
name = input("Name: ")
print("Hello, " + name)
```

```
Name: Carter
Hello, Carter
```

hello.py

```
name = input("Name: ")
print("Hello, " + name)
```

hello.py

```
name = input("Name: ")
print("Hello, " + name)
```

Variable

A name for a value that can change.

hello.py

```
name = input("Name: ")
print("Hello, " + name)
```

hello.py

```
name = input("Name: ")
name = name.capitalize()
print("Hello, " + name)
```

name

"carter"

hello.py

```
name = input("Name: ")
name = name.capitalize()
print("Hello, " + name)
```

name

"Carter"

```
name = name.capitalize()
```

```
name = name.capitalize()
```

method

```
name = name.capitalize()
```

string method

```
name = name.capitalize()
```

Method

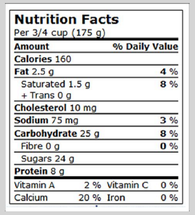
A function that acts on the object that "called" (used) it.

playback.py

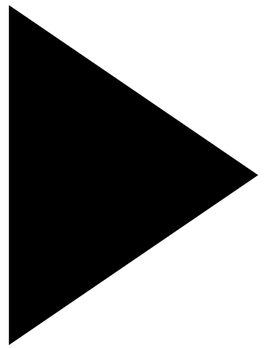
faces.py

twttr.py

coke.py

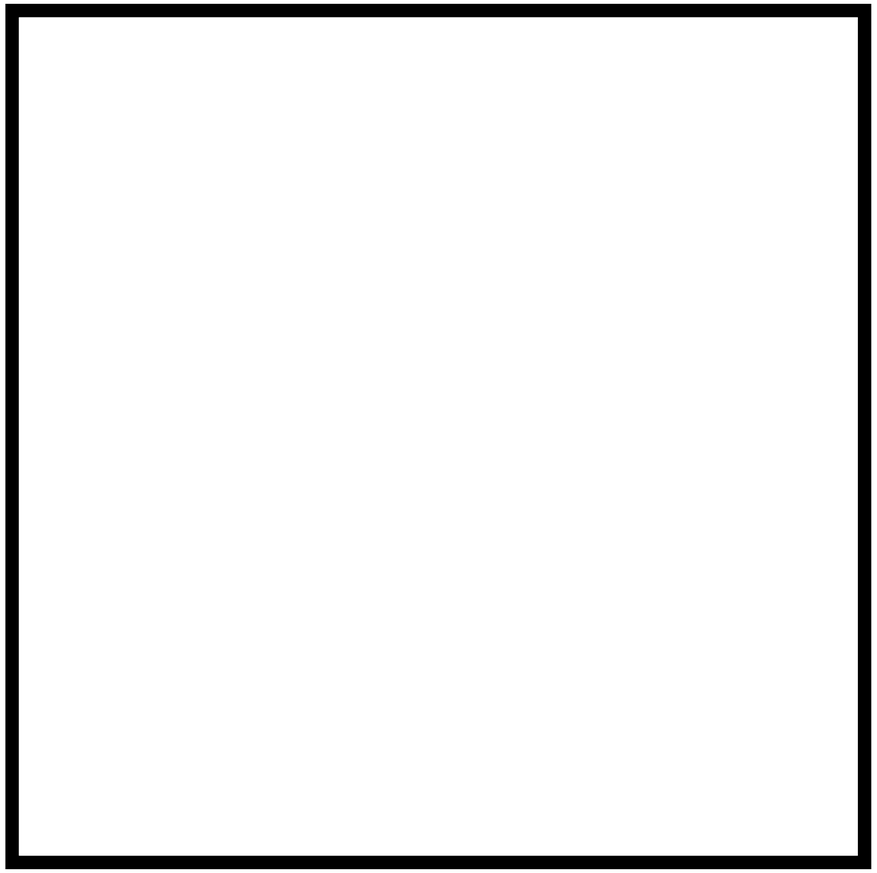


nutrition.py



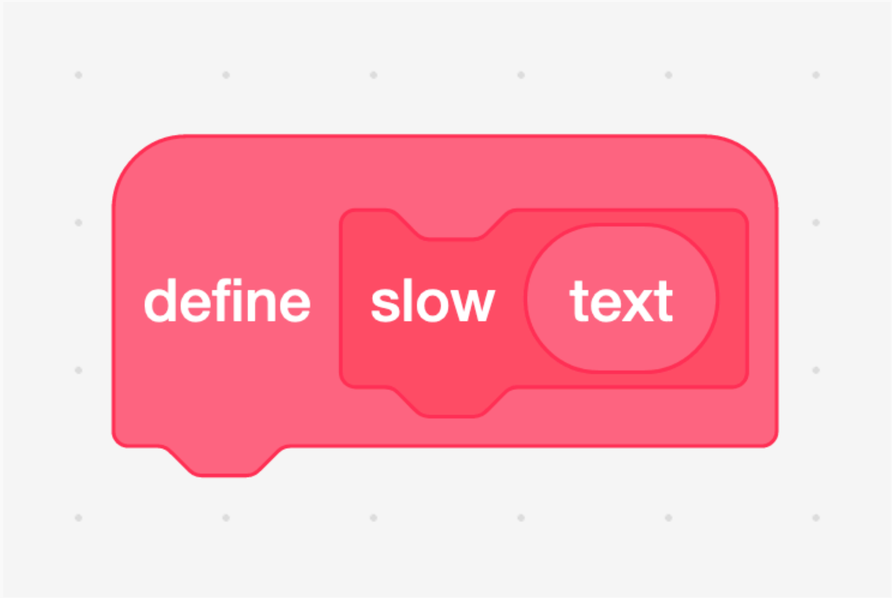
Playback

```
$ python playback.py
This is CS50.
This...is...CS50.
```

"This is CS50." →  → "This…is…CS50."

https://docs.python.org/3/library/stdtypes.html#string-methods

Abstraction




```
def slow(text):
    ...
```

Return Values

```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    text.replace(" ", "...")

main()
```

main

text

```
"This is CS50."
```

```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    text.replace(" ", "...")

main()
```

main

text

```
"This is CS50."
```

slow

text

```
"This is CS50."
```

```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    text.replace(" ", "...")

main()
```

main

text

```
"This is CS50."
```

slow

text

```
"This...is...CS50."
```

```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    text.replace(" ", "...")

main()
```

main

text

```
"This is CS50."
```

slow

```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    return text.replace(" ", "...")

main()
```

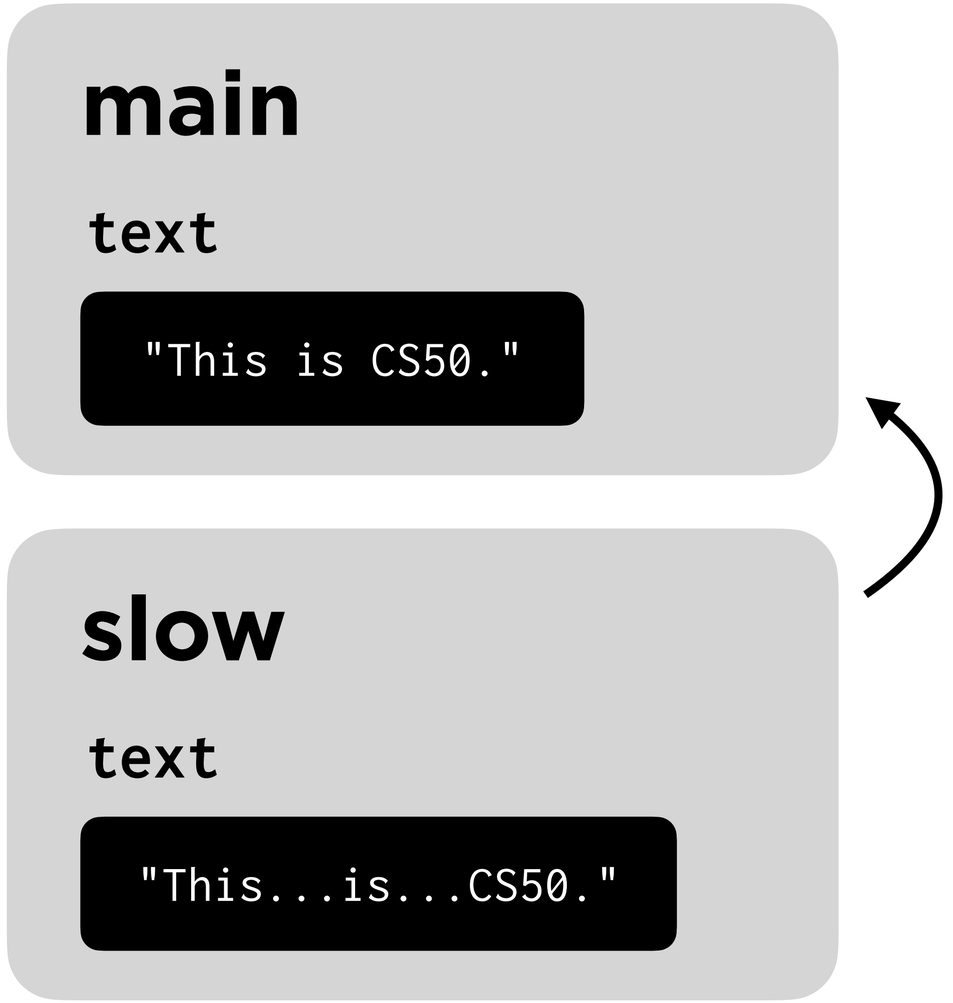
main

text

"This is CS50."

```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    return text.replace(" ", "...")

main()
```

```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    return text.replace(" ", "...")

main()
```



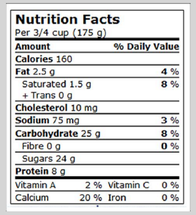
```
def main():
    text = input("")
    print(slow(text))

def slow(text):
    return text.replace(" ", "...")

main()
```

playback.py

faces.py

twttr.py

coke.py

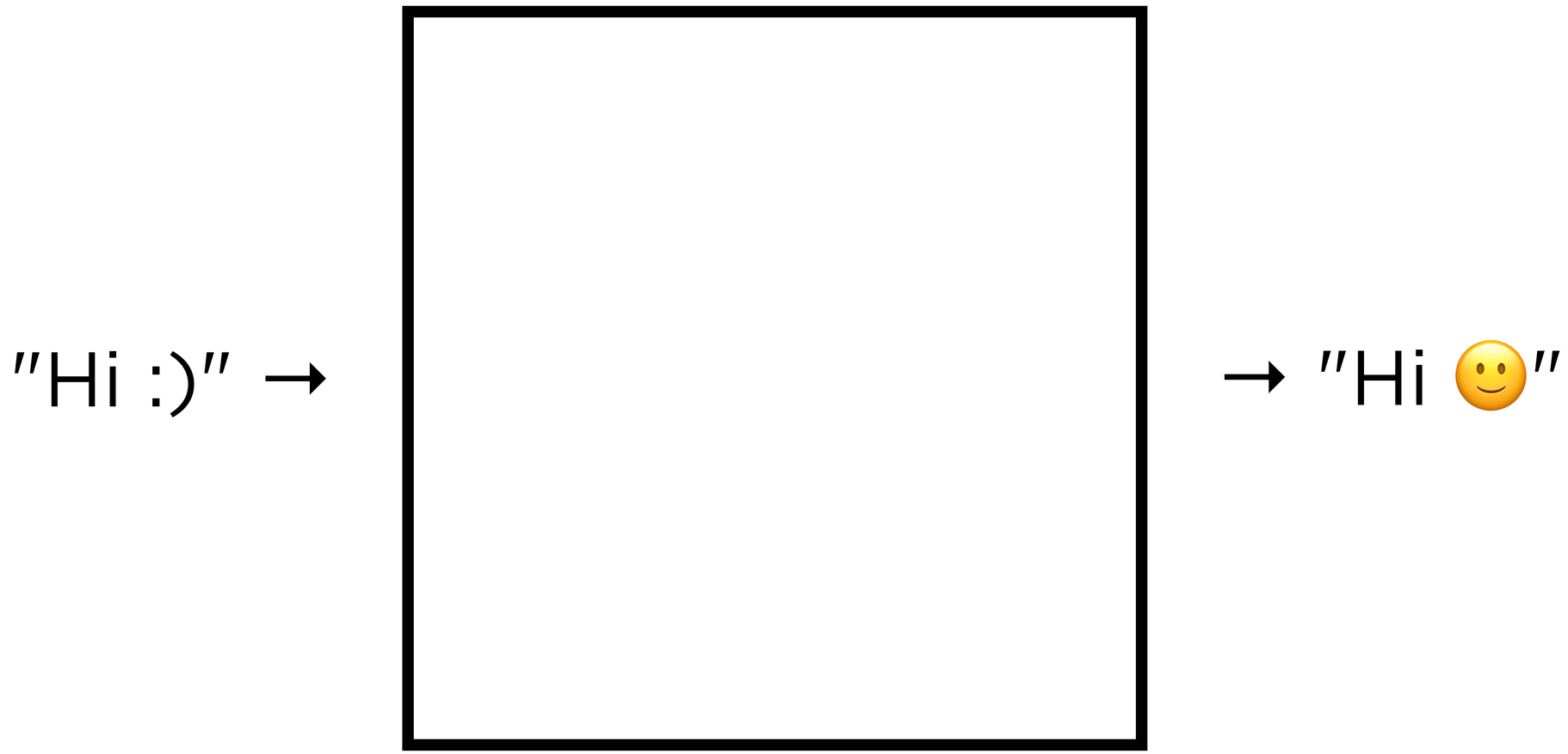


nutrition.py

Making Faces

```
$ python faces.py
Hi :)
Hi 🙂
```

```
$ python faces.py
Bye :(
Bye 😕
```


"Hi :)" →
→ "Hi 🙂"

Abstraction

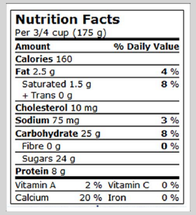
```
def convert(text):
    ...
    return ...
```

playback.py

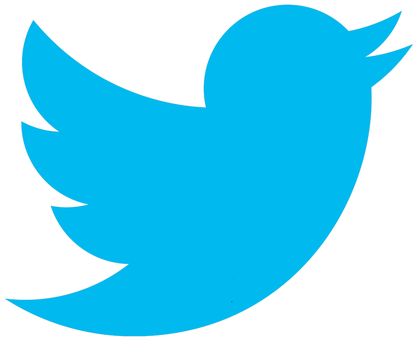
faces.py

twttr.py

coke.py



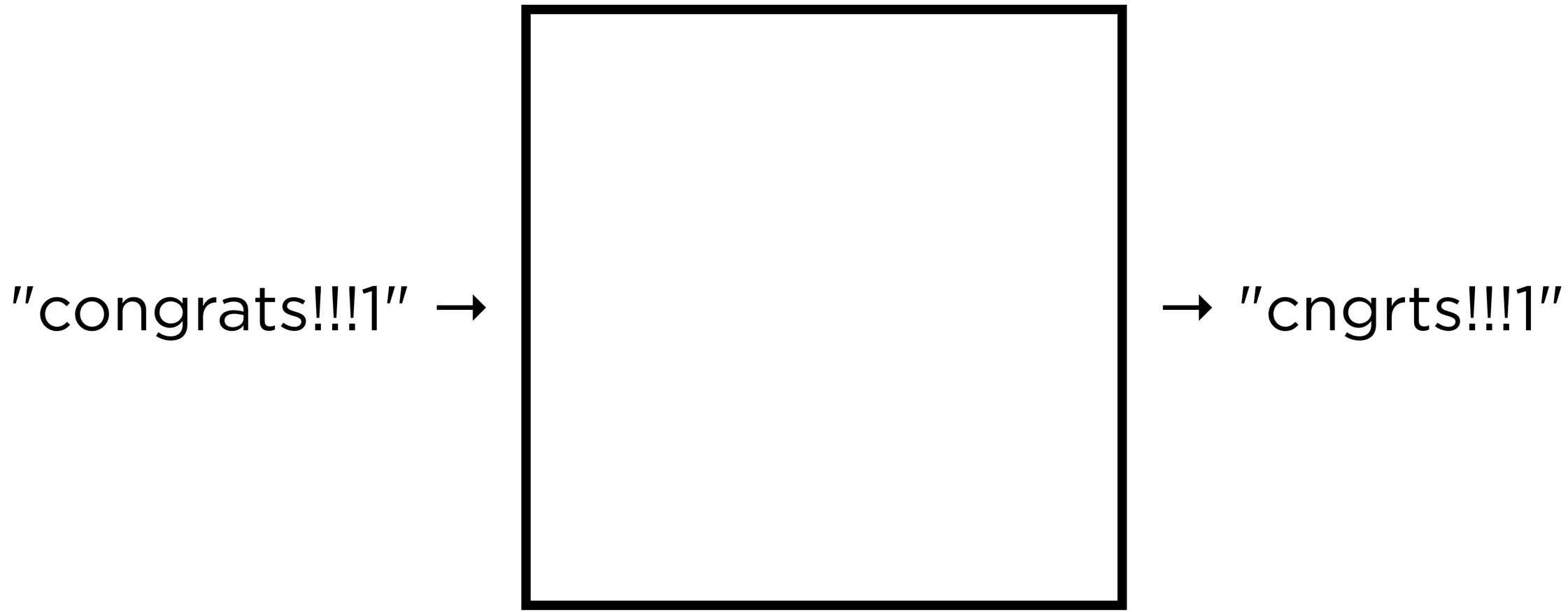
nutrition.py



Just setting up my twttr

```
$ python twttr.py
Twitter
Twttr
```

```
$ python twttr.py
congrats!!!1
cngrts!!!1
```


"congrats!!!1" →
→ "cngrts!!!1"

```
text
```

```
text.replace("a", "")
```

```
text.replace("a", "").replace("e", "")
```

```
text.replace("a", "").replace("e", "").replace("i", "")
```

Pseudocode

```
# Get input from the user
# For each character
    # If character is NOT a vowel
        # Print the character
```

```
# Get input from the user
# For each character
    # If character is NOT a vowel
        # Print the character
```

Abstraction

```
def is_vowel(character):
    return ...
```

Lists

A E I O U

'A' 'E' 'I' 'O' 'U'

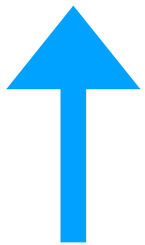
['A', 'E', 'I', 'O', 'U']

in

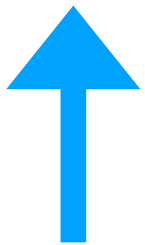
```
'I' in ['A', 'E', 'I', 'O', 'U']
```

```
'I' in ['A', 'E', 'I', 'O', 'U']
```

```
'I' in ['A', 'E', 'I', 'O', 'U']
```



```
'I' in ['A', 'E', 'I', 'O', 'U']
```

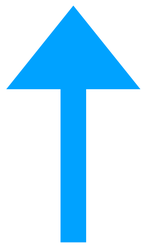


```
'I' in ['A', 'E', 'I', 'O', 'U']
```

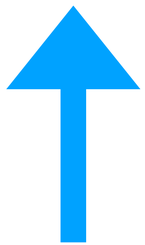
```
'B' in ['A', 'E', 'I', 'O', 'U']
```

```
'B' in ['A', 'E', 'I', 'O', 'U']
```

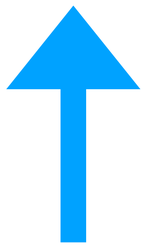
```
'B' in ['A', 'E', 'I', 'O', 'U']
```



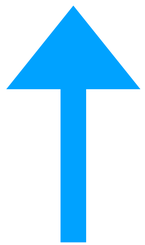
```
'B' in ['A', 'E', 'I', 'O', 'U']
```

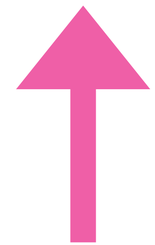


```
'B' in ['A', 'E', 'I', 'O', 'U']
```



```
'B' in ['A', 'E', 'I', 'O', 'U']
```





```
'B' in ['A', 'E', 'I', 'O', 'U']
```

```
# Get input from the user
# For each character
    # If character is NOT a vowel
        # Print the character
```

```
for c in text:
    print(c)
```

"In the great

green room"

```
for c in text:
    print(c)
```

c

"In the great

green room"

```
for c in text:
    print(c)
```

c

"In the great

green room"

```
for c in text:
    print(c)
```

c

"In the great

green room"

```
for c in text:
    print(c)
```

c

"In the great

green room"

```
for c in text:
    print(c)
```

c

"In the great

green room"

```
for c in text:
    print(c)
```

c

"In the great

green room"

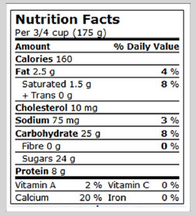
```
# Get input from the user
# For each character
    # If character is NOT a vowel
        # Print the character
```

playback.py

faces.py

twttr.py

coke.py



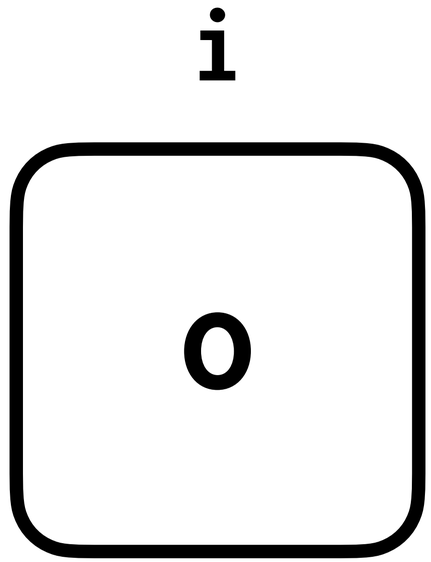
nutrition.py

Coke Machine

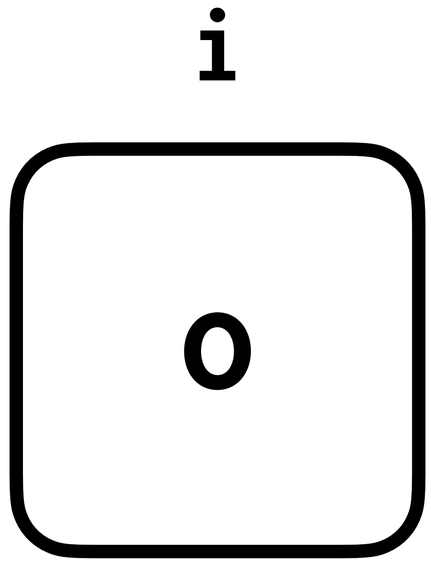
While Loops

```
i = 0
while i < 3:
    i += 1
```

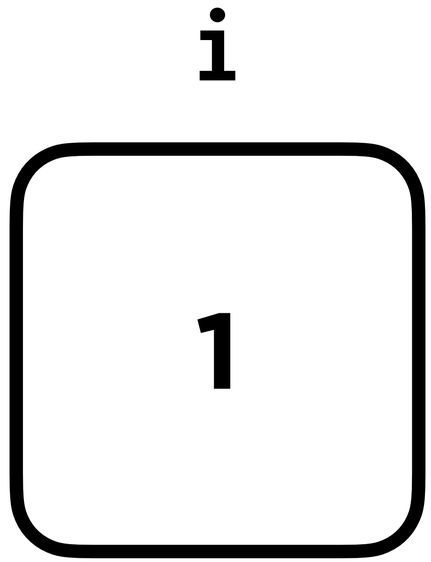
```
i = 0
while i < 3:
    i += 1
```



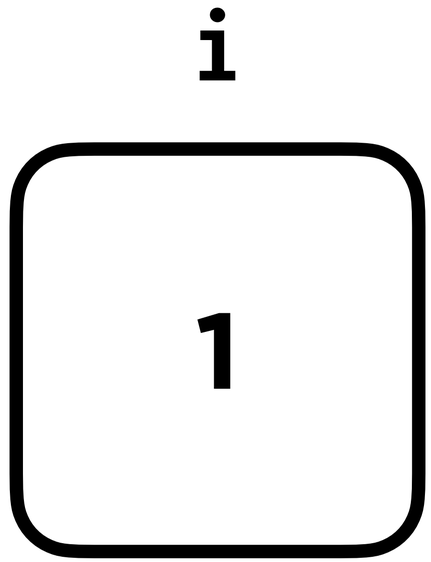
```
i = 0
while i < 3:
    i += 1
```



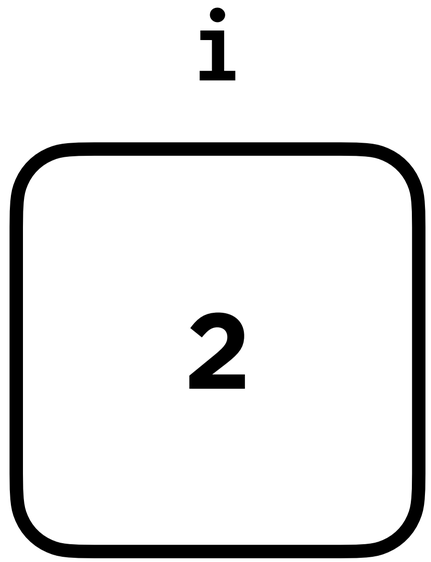
```
i = 0
while i < 3:
    i += 1
```



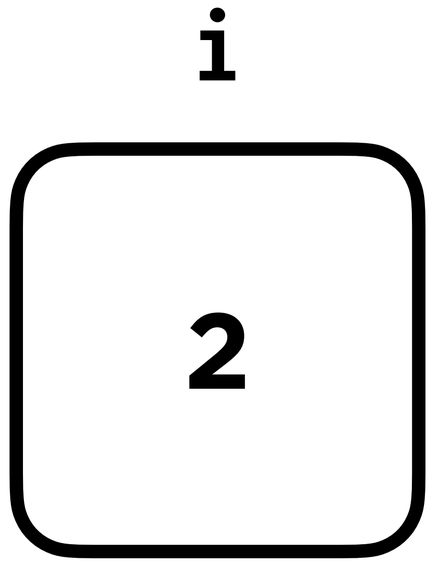
```
i = 0
while i < 3:
    i += 1
```



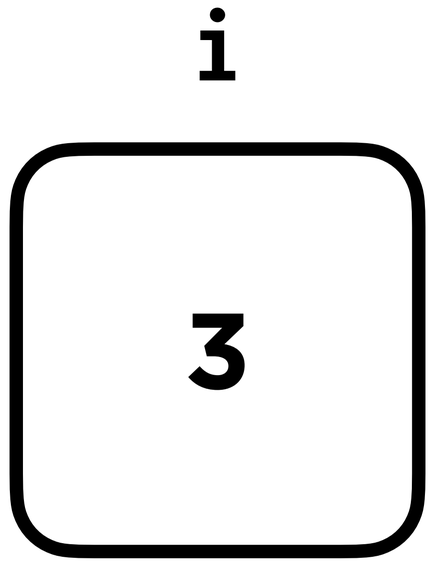
```
i = 0
while i < 3:
    i += 1
```



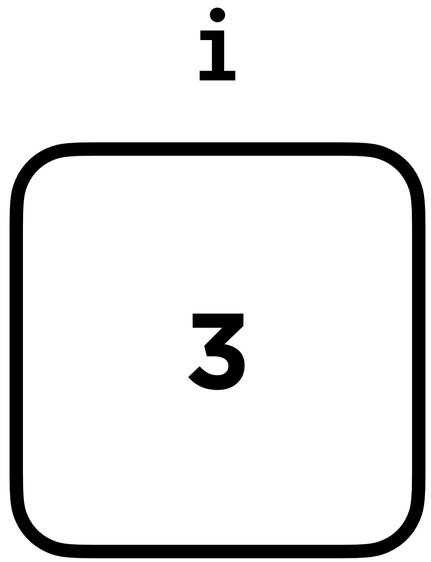
```
i = 0
while i < 3:
    i += 1
```



```
i = 0
while i < 3:
    i += 1
```



```
i = 0
while i < 3:
    i += 1
```



Pseudocode

```
# While amount owed is > 0
    # Accept coin from user
    # Check if valid coin
        # Subtract coin from amount owed
```

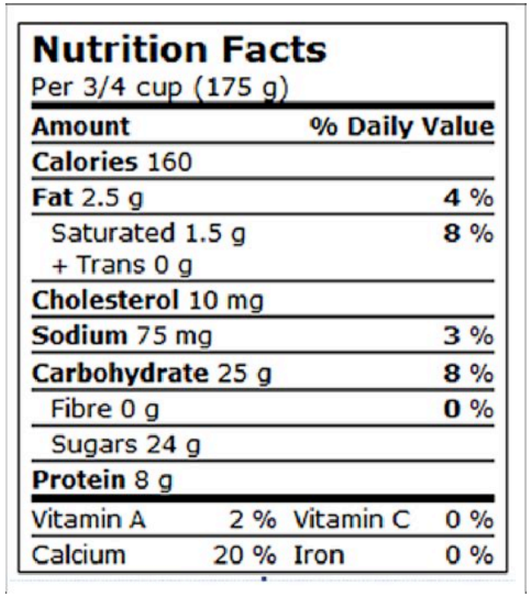
playback.py

faces.py

twttr.py

coke.py

nutrition.py



Nutrition Facts

Per 3/4 cup (175 g)

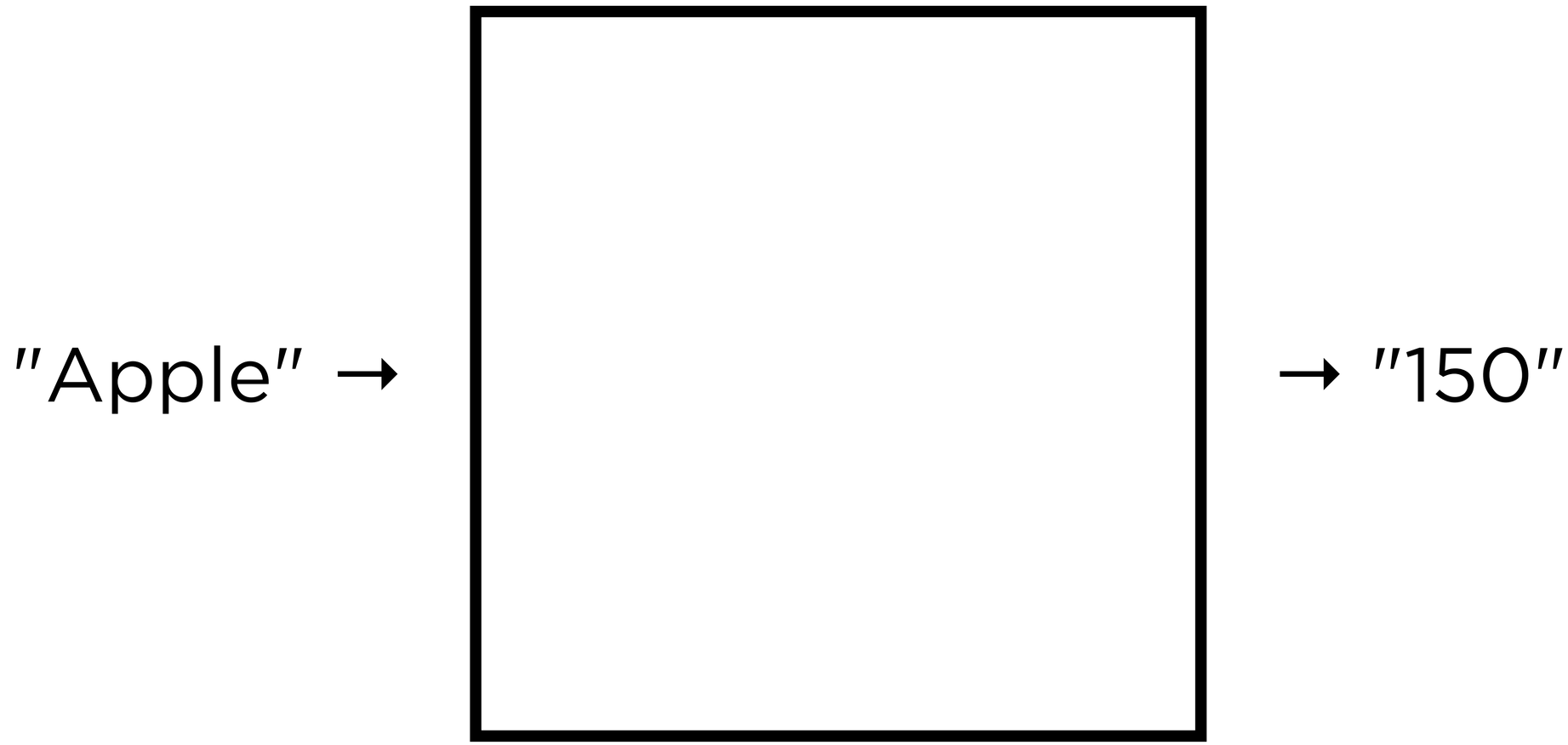
Amount	% Daily Value
Calories 160	
Fat 2.5 g	**4** %
Saturated 1.5 g + Trans 0 g	**8** %
Cholesterol 10 mg	
Sodium 75 mg	**3** %
Carbohydrate 25 g	**8** %
Fibre 0 g	**0** %
Sugars 24 g	
Protein 8 g	

Vitamin A	2 %	Vitamin C	0 %
Calcium	20 %	Iron	0 %

Nutrition

```
$ python nutrition.py
Strawberries
Calories: 50
```

```
$ python nutrition.py
Apple
Calories: 130
```


"Apple" →
→ "150"

Dictionaries

```
fruits = {
    "apple": 130,
    "strawberries": 50
}
```

Key	Value
apple	130
strawberries	50

```
fruits = {
    "apple": 130,
    "strawberries": 50
}

fruits["strawberries"]
```

Key	Value
apple	130
strawberries	50

```
fruits = {
    "apple": 130,
    "strawberries": 50
}

fruits["apple"]
```

Key	Value
apple	130
strawberries	50

Exceptions

```
fruits = {
    "apple": 130,
    "strawberries": 50
}

fruits["chocolate"]
```

Key	Value
apple	130
strawberries	50

```
fruits = {
    "apple": 130,
    "strawberries": 50
}

fruits["chocolate"]
```

Key	Value
apple	130
strawberries	50

KeyError

```
fruits = {
    "apple": 130,
    "strawberries": 50
}

try:
    fruits["chocolate"]
except KeyError:
    print("Not here!")
```

Key	Value
apple	130
strawberries	50

Submission

- **Submit code files to Gradescope** by Thursday, February 2, 3:10 PM.
- Graded based on completion, but please double check to be sure your files are named correctly:
  - `playback.py` **not** `playback (1).py`