Chapter 5

Functions -- QuickStart

Functions

- From Mathematics we know that functions perform some operation and return one value.
- They "encapsulate" the performance of some particular operation, so it can be used by others (for example, the `sqrt()` function)

Why have them?

- Support divide-and-conquer strategy
- Abstraction of an operation
- Reuse. Once written, use again
- Sharing. If tested, others can use
- Security. Well tested, then secure for reuse
- Simplify code. More readable.

Mathematical Notation

- Consider a function which converts temperatures in Celsius to temperatures in Fahrenheit.
  - Formula: F = C * 1.8 + 32.0
  - Functional notation:

    F ~ celsius_to_Fahrenheit(C) where

        celsius_to_Fahrenheit(C) = C * 1.8 + 32.0

Python Invocation

- Math: F = celsius_to_Fahrenheit(C)
- Python, the invocation is much the same

```
F = celsius_to_Fahrenheit(cel_float)
```

Terminology: `cel_float` is the ***argument***

Function defintion

- Math: g(C) = C*1.8 + 32.0
- Python

```
def celsius_to_Fahrenheit(param_float):
    return param_float * 1.8 + 32.0
```

- Terminology: `param_float` is the ***parameter***

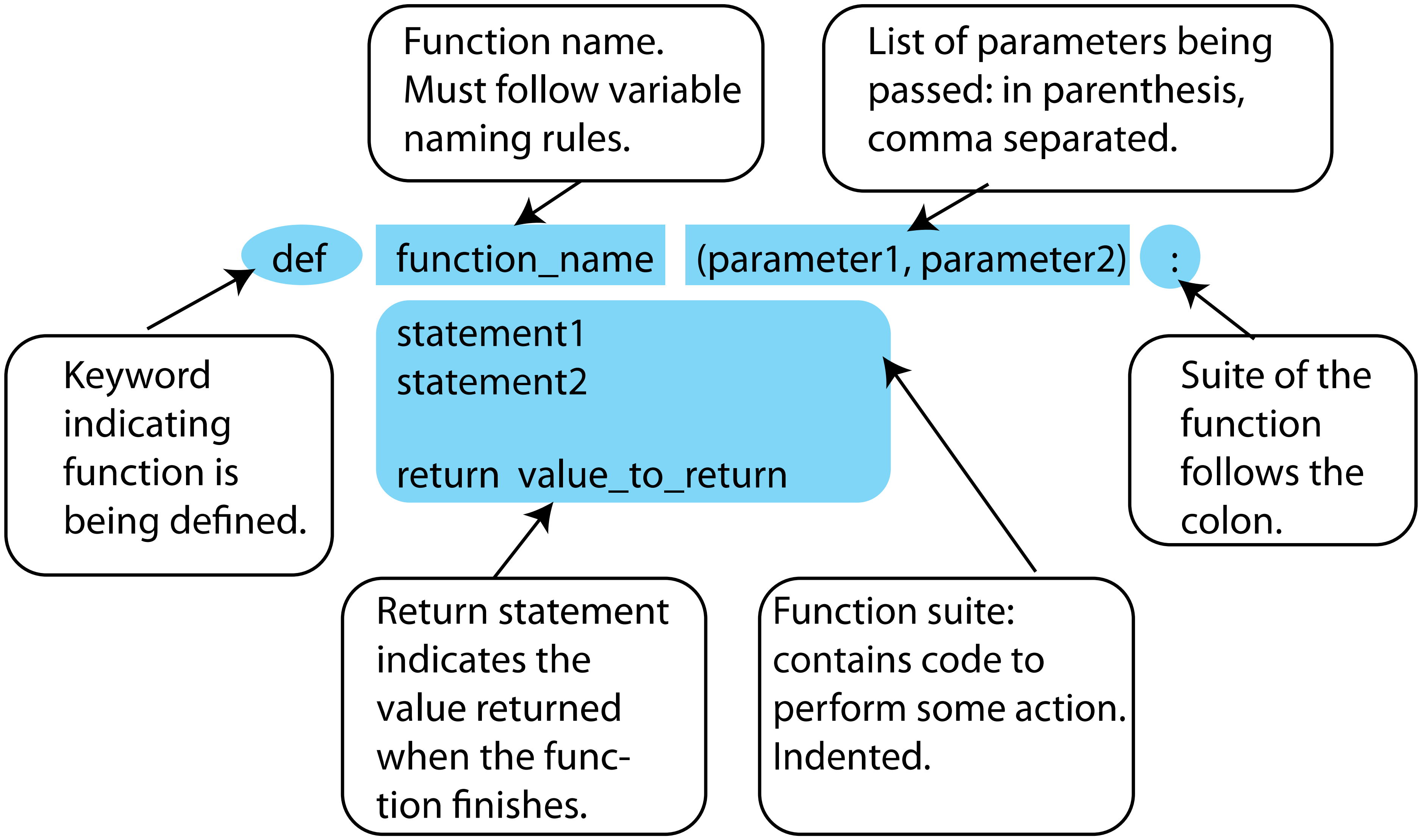

Function name. Must follow variable naming rules.
List of parameters being passed: in parenthesis, comma separated.
def
function_name
(parameter1, parameter2)
:
statement1
statement2
return value_to_return
Keyword indicating function is being defined.
Suite of the function follows the colon.
Return statement indicates the value returned when the function finishes.
Function suite: contains code to perform some action. Indented.

FIGURE 6.1 Function parts.

return statement

- The `return` statement indicates the value that is returned by the function
- The statement is optional (the function can return nothing). If no `return`, function is often called a procedure.

Code Listing 6.1

Temp convert

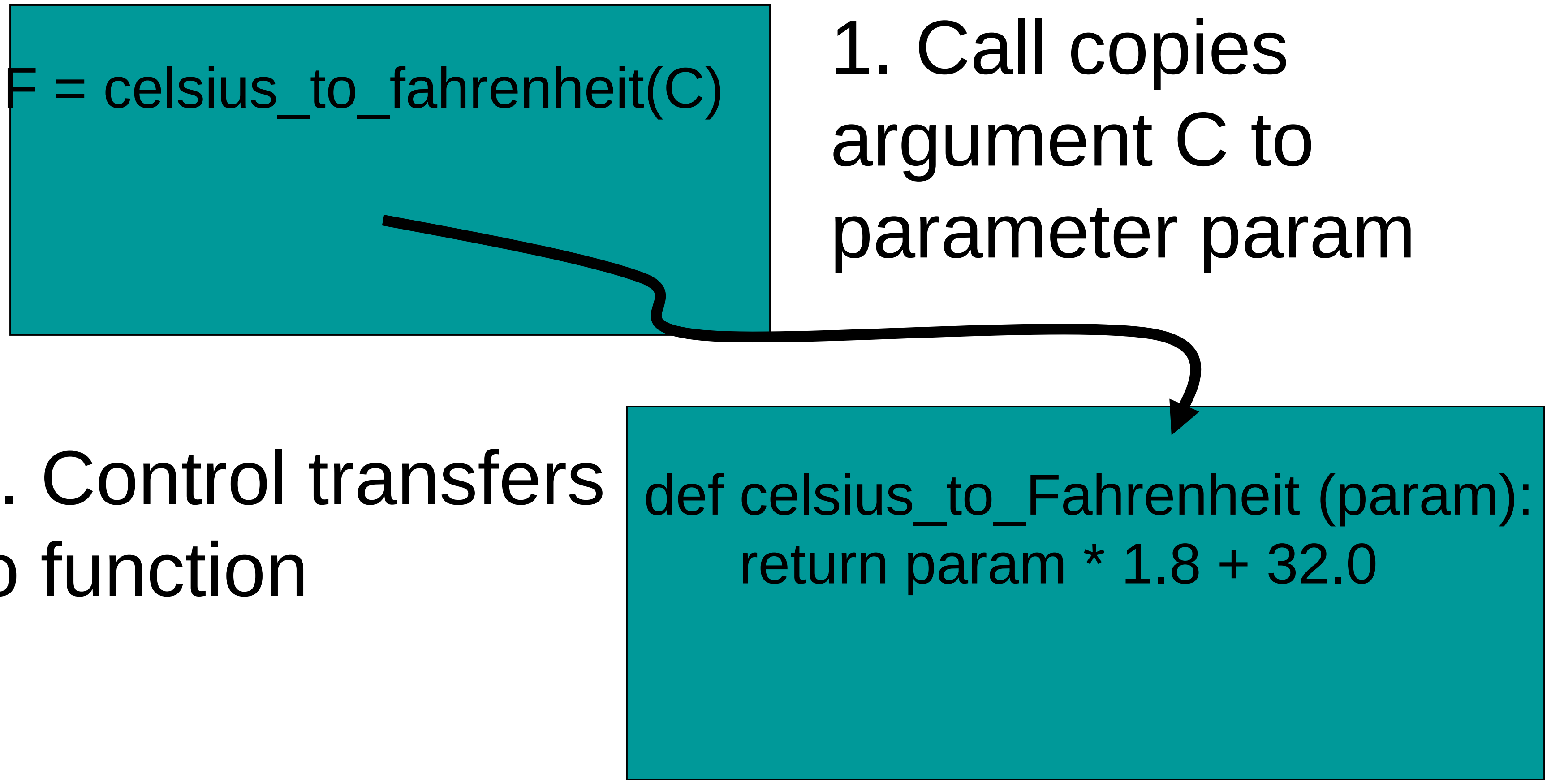
```python
# Temperature conversion

def celsius_to_fahrenheit(celsius_float):
    """ Convert Celsius to Fahrenheit."""
    return celsius_float * 1.8 + 32
```

Triple quoted string in function

- A triple quoted string just after the def is called a ***docstring***
- docstring is documentation of the function's purpose, to be used by other tools to tell the user what the function is used for. More on that later

Operation



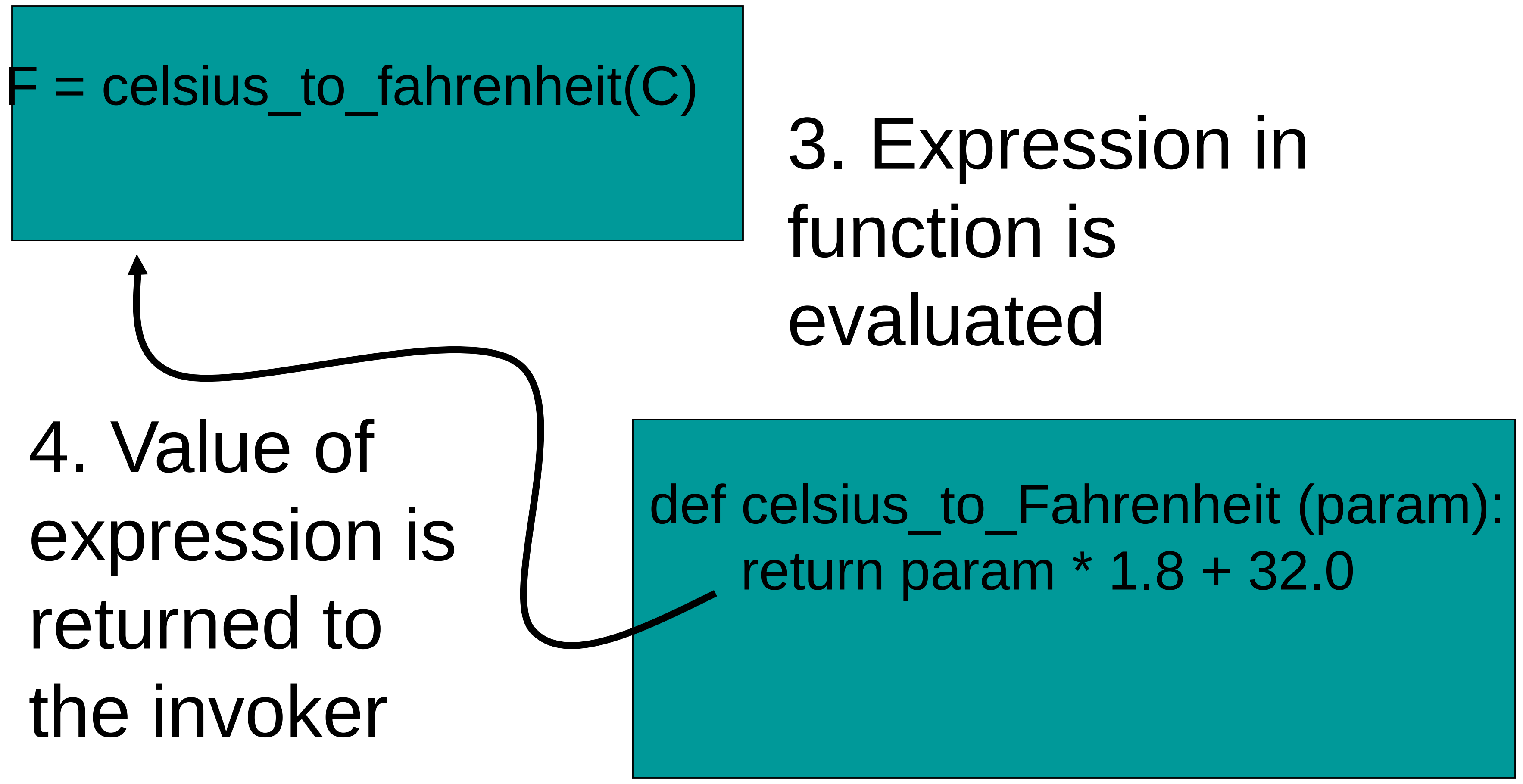
```
F = celsius_to_fahrenheit(C)
```

1. Call copies argument C to parameter param

```
def celsius_to_Fahrenheit (param):
    return param * 1.8 + 32.0
```

2. Control transfers to function

Operation (con't)



```
F = celsius_to_fahrenheit(C)
```

3. Expression in function is evaluated

4. Value of expression is returned to the invoker

```
def celsius_to_Fahrenheit (param):
        return param * 1.8 + 32.0
```

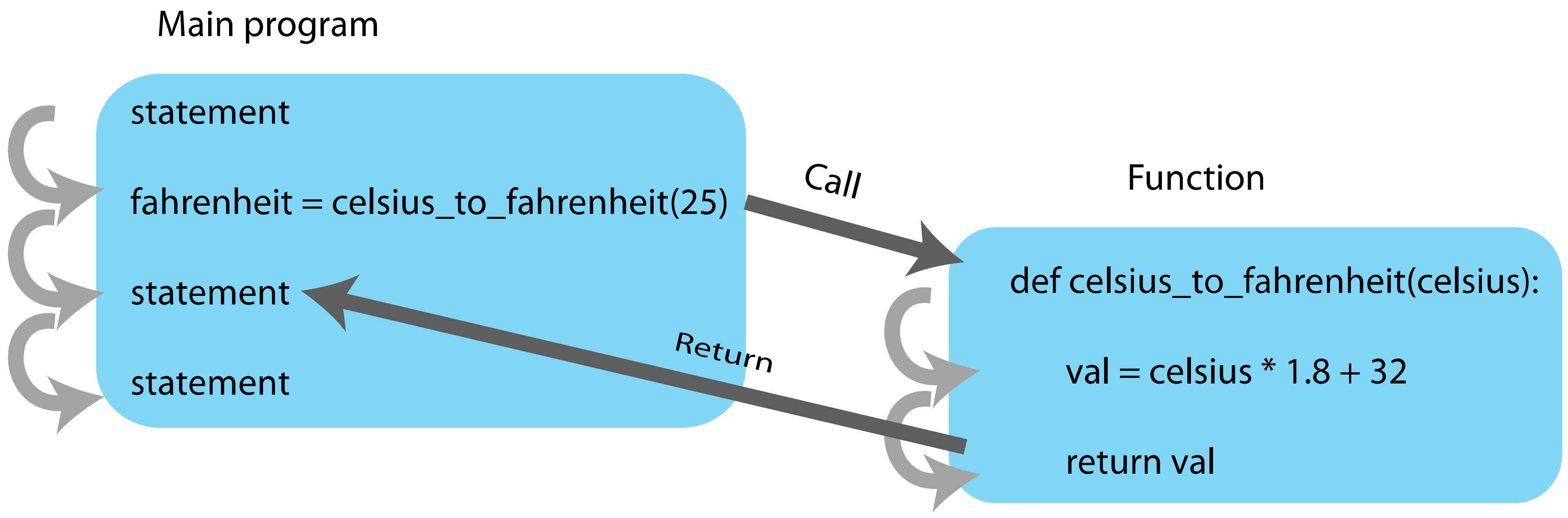


FIGURE 6.2 Function flow of control.

Code Listing 6.2

Full Temp Program

```python
# Conversion program

def celsius_to_fahrenheit(celsius_float):
    """ Convert Celsius to Fahrenheit."""
    return celsius_float * 1.8 + 32

# main part of the program
print("Convert Celsius to Fahrenheit.")
celsius_float = float(input("Enter a Celsius temp: "))
# call the conversion function
fahrenheit_float = celsius_to_fahrenheit(celsius_float)
# print the returned value
print(celsius_float," converts to ",fahrenheit_float," Fahrenheit")
```

Code Listing 6.3

re-implement len

```python
def length(a_str):
    """Return the length of a_str"""
    count = 0
    for char in a_str:
        count += 1
    return count
```

Code Listing 6.4

Count letters in string

check membership in lowercase

- `import string`
- use `string.ascii_lowercase`, string of lowercase english letters
  - 'abcdefghijklmnopqrstuvwxyz'
- check if each char is a member (using in operator) of `string.ascii_lowercase`
- `char.lower()` before membership (catch Capital Letters that way)

```python
import string

def letter_count(a_str):
    """Return the count of letters in a_str."""
    count = 0
    for char in a_str:
        if char.lower() in string.ascii_lowercase:
            count += 1
    return count
```

Word Puzzle

- Find an English language word that has the vowels 'a', 'e', 'i', 'o', and 'u' in sequence

Reading a file of Text

Remember how to work with text files

•The open function takes a string (a file name) and a mode ('r' for reading) and returns a file object.

•You can use a for loop on the file object to fetch one line of text at a time (a line ends with a carriage return)

Code Listing 6.5

Open a file to read

Need a list of words

We use a dictionary file (easily found on the web) of english words, one word per line

•open the file

•process each line (a single word)

•this example just prints them all

```python
# Print all words in a dictionary file that has one word per line

# open file named "dictionary.txt" for reading ('r')
data_file = open("dictionary.txt", 'r')

# iterate through the file one line at a time
for line_str in data_file:
    print(line_str)
```

Code Listing 6.6

Clean a word

clean the word

- `strip` method removes white space characters from the beginning and end of a string (can remove other chars as well)
  - beginning and end only, not the middle
  - all such characters from either end
  - file line likely has returns or tabs of spaces which might hurt compares
- `lower` method so case won't matter

```python
def clean_word(word):
    """Return word in lowercase stripped of whitespace."""
    return word.strip().lower()
```

Code Listing 6.8

Extract Vowels

collect vowels

- collect only vowels as a string, in order from the word, and compare against the reference "aeiou"
  - use `in` operator for membership
  - use `+` operator to concat vowels together

```python
def get_vowels_in_word(word):
    """Return vowels in string word—include repeats."""
    vowel_str = "aeiou"
    vowels_in_word = ""
    for char in word:
        if char in vowel_str:
            vowels_in_word += char
    return vowels_in_word
```

Code Listion 6.9

Solution to word problem

```
data_file = open("dictionary.txt", "r")

def clean_word(word):
    """Return word in lowercase stripped of whitespace."""
    return word.strip().lower()

def get_vowels_in_word(word):
    """Return vowels in string word—include repeats."""
    vowel_str = "aeiou"
    vowels_in_word = ""
    for char in word:
        if char in vowel_str:
            vowels_in_word += char
    return vowels_in_word

# main program
print("Find words containing vowels 'aeiou' in that order:")
for word in data_file:          # for each word in the file
    word = clean_word(word)  # clean the word
    if len(word) <= 6:          # if word is too small, skip it
        continue
    vowel_str = get_vowels_in_word(word)   # get vowels in word
    if vowel_str == 'aeiou':                # check if you have exactly all
vowels in order
        print(word)
```

Did functions help?

- Made our problem solving easier (solved smaller problems as functions)
- main program very readable (details hid in the functions)

How to write a function

- ***Does one thing***. If it does too many things, it should be broken down into multiple functions (refactored)
- ***Readable.*** How often should we say this? If you write it, it should be readable
- ***Reusable***. If it does one thing well, then when a similar situation (in another program) occurs, use it there as well.

More on functions

- ***Complete***. A function should check for all the cases where it might be invoked. Check for potential errors.
- ***Not too long***. Kind of synonymous with do one thing. Use it as a measure of doing too much.

Rule 8

A function should do one thing

Procedures

- Functions that have no return statements are often called *procedures*.
- Procedures are used to perform some duty (print output, store a file, etc.)
- Remember, return is not required.

Multiple returns in a function

- A function can have multiple `return` statements.
- Remember, the first `return` statement executed ends the function.
- Multiple returns can be confusing to the reader and should be used judiciously.

Reminder, rules so far

1. Think before you program!
2. A program is a human-readable essay on problem solving that also happens to execute on a computer.
3. The best way to imporve your programming and problem solving skills is to practice!
4. A foolish consistency is the hobgoblin of little minds
5. Test your code, often and thoroughly
6. If it was hard to write, it is probably hard to read. Add a comment.
7. All input is evil, unless proven otherwise.
8. A function should do one thing.