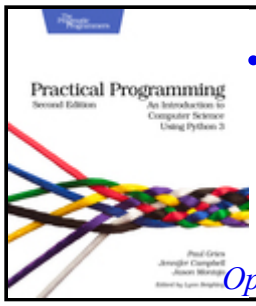
for loop over str



- Chapter 9.2 Processing Characters in Strings

Optional reading

For Loops

The general form of a for loop over a string is:

```
for variable in str:
    body
```

The variable refers to each character of the string in turn and executes the body of the loop for each character. For example:

```
>>> s = 'yesterday'
>>> for char in s:
...     print(char)
...
y
e
s
t
e
r
d
a
y
```

Accumulator pattern: numeric accumulator

Consider the code below, in which the variable `num_vowels` is an accumulator:

```
def count_vowels(s):
    """ (str) -> int

    Return the number of vowels in s. Do not treat letter y as a vowel

    >>> count_vowels('Happy Anniversary!')
    5
    >>> count_vowels('xyz')
    0
    """

    num_vowels = 0

    for char in s:
        if char in 'aeiouAEIOU':
            num_vowels = num_vowels + 1

    return num_vowels
```

The `loop` in the function above will loop over each character that `s` refers to, in turn. The body of the loop is executed for each character, and when a character is a vowel, the `if` condition is `True` and the value that `num_vowels` refers to is increased by one.

The variable `num_vowels` is an accumulator, because it accumulates information. It starts out referring to the value `0` and by the end of the function it refers to the number of vowels in `s`.

Accumulator pattern: string accumulator

In the following function, the variable `vowels` is also an accumulator:

```
def collect_vowels(s):
    """ (str) -> str

    Return the vowels from s.  Do not treat the letter
    y as a vowel.

    >>> collect_vowels('Happy Anniversary!')
    'aAiea'
    >>> collect_vowels('xyz')
    ''
    """

    vowels = ''

    for char in s:
        if char in 'aeiouAEIOU':
            vowels = vowels + char

    return vowels
```

Variable `vowels` initially refers to the empty string, but over the course of the function it accumulates the vowels from `s`.

Jennifer Campbell • Paul Gries
University of Toronto
