

Lecture 16

Topics: *Chapter 8. Loop Structures and Booleans*
review in-class assignment3 from previous class
8.1 *For loops: a quick review*
8.2 *Indefinite loops*
8.3 *Common loop patterns: interactive, sentinel*

8.1 For loops: a quick review

Definite Loops

Let's have a brief review of definite loops:

A Python `for` loop has this general form:

```
for <var> in <sequence>:  
    <body>
```

`<body>` is any sequence of Python statements

`<var>` is the *loop index* ,

takes on each successive value in the `sequence`, and
the statements in the `body` are executed once for each value.

`sequence` portion often consists of a *list* of values.

8.1 For loops: a quick review

Definite Loops

Example 1:

```
y = 1
for counter in [1,2,3,4]:
    y = y + counter
    print("counter={0}, y={1}".format(counter,y))
```

8.1 For loops: a quick review

Definite Loops

Example 1:

```
y = 1
for counter in [1,2,3,4]:
    y = y + counter
    print("counter={0}, y={1}".format(counter,y))
```

Example 2:

```
for i in range(10):
    x = 3.9 * x * (1-x)
    print(x)
```

begin to typing in `range(` in the interactive window - you'll see:
`range([start,] stop[, step])` -> list of integers

8.1 For loops: a quick review

Definite Loops

Example 1:

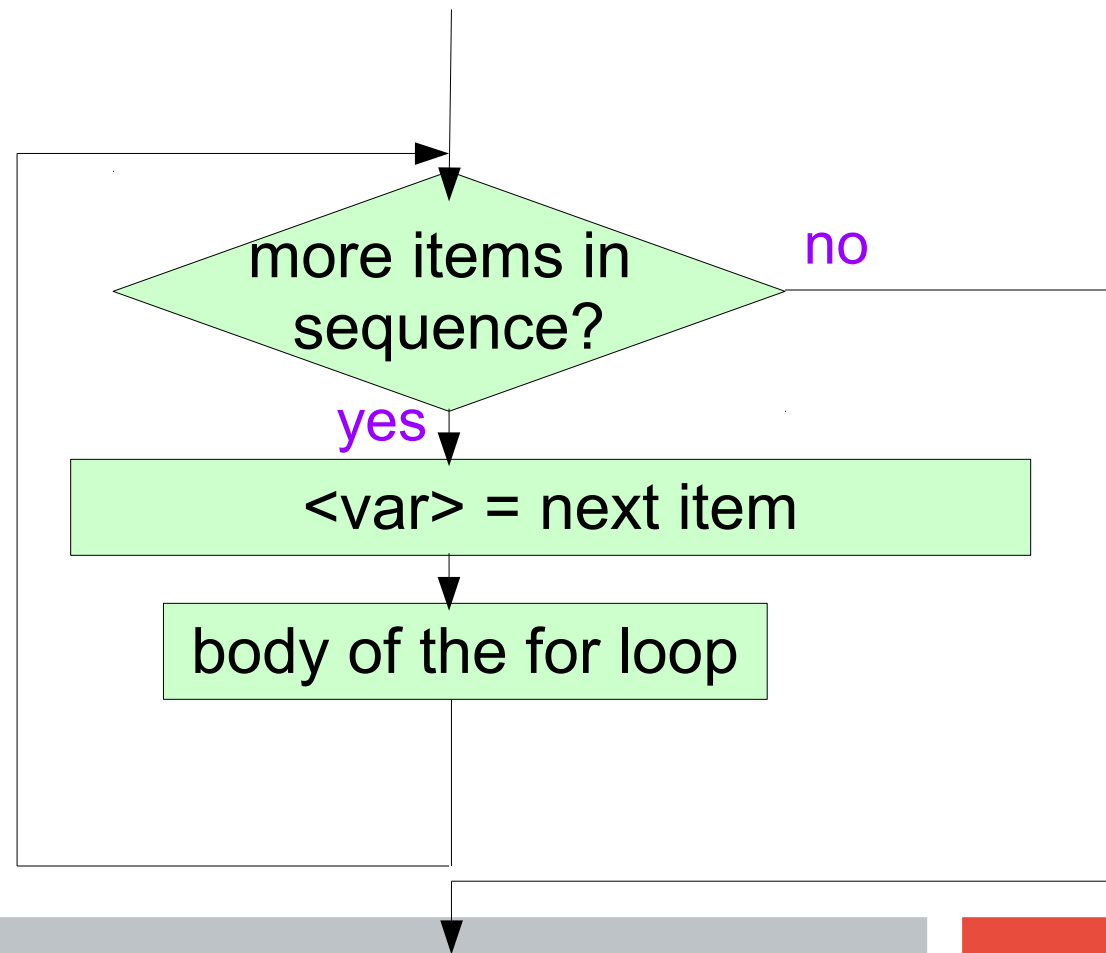
```
y = 1
for counter in [1,2,3,4]:
    y = y + counter
    print("counter={0}, y={1}".format(counter,y))
```

Example 2:

```
for i in range(10):
    x = 3.9 * x * (1-x)
    print(x)
```

8.1 For loops: a quick review

Definite Loops



8.2 Indefinite loops

Indefinite loops

Assume that we want to write the program that finds the average of inputted numbers. And I don't know in advance how many numbers will be inputted.

Can we use a for loop here?

8.2 Indefinite loops

Indefinite loops

Assume that we want to write the program that finds the average of inputted numbers. And I don't know in advance how many numbers will be inputted.

Can we use a for loop here? **No**

8.2 Indefinite loops

Indefinite loops

Let's take a look at **indefinite (conditional) loop**:

```
while <condition>:  
    <body>
```

<condition> is a boolean expression (just like in **if** statements).

The **<body>** of the loop executes repeatedly as long as the condition remains true.

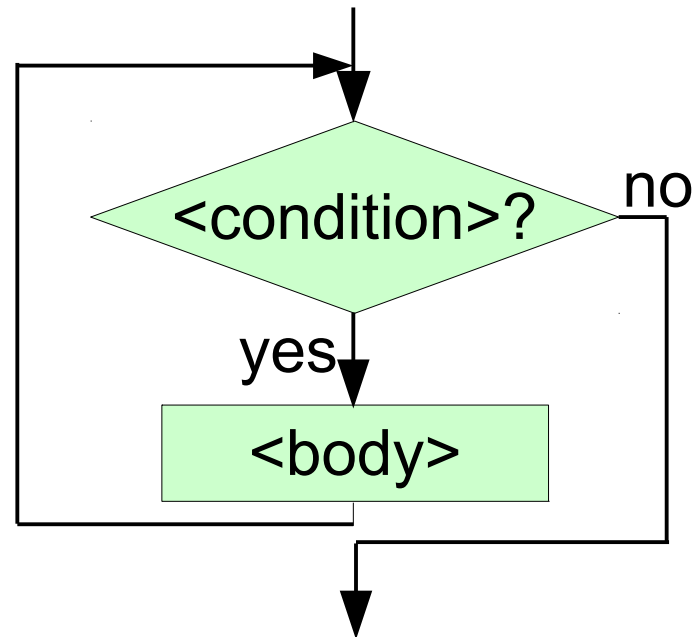
8.2 Indefinite loops

Indefinite loops

Let's take a look at **indefinite (conditional) loop**:

```
while <condition>:  
  <body>
```

Flowchart of the while loop



8.2 Indefinite loops

Indefinite loops

example: a program that counts from 0 to 9

```
i=0
while i<=9:
    print(i)
    i=i+1
```

the *for* loop would look:

```
for i in range(10):
    print(i)
```

8.2 Indefinite loops

Indefinite loops

example: a program that counts from 0 to 9

```
i=0
while i<=9:
    print(i)
    i=i+1
```

the *for* loop would look:

```
for i in range(10):
    print(i)
```

Now, if we slightly change our original program:

```
i=0
while i<=9:
    print i
```

- we get an infinite loop

Infinite loops are a bad thing.

8.2 Indefinite loops

Indefinite loops

example: a program that takes all decimal values until a user enters a negative number and finds their average.

This example is not do easy to do with a for loop

8.2 Indefinite loops

Indefinite loops

example: a program that takes all decimal values until a user enters a negative number and finds their average.

```
x, s, counter = 0, 0, 0
```

```
while x >= 0:
```

```
    x = float(input("Enter a value:"))
```

```
    s += x
```

```
    counter += 1
```

```
print("You entered {0} non-negative values, their  
average is {1}.".format(counter, s/counter))
```

8.3 Common loop patterns: interactive, sentinel

Indefinite loops

Two good uses of indefinite loops:

- **interactive loops**

at each iteration ask the user if he/she wants to input more data values or it is enough

- **sentinel loop**

loop continues to process data until reaching a special value that signals the end (that value is called **sentinel**)

8.3 Common loop patterns: interactive, sentinel

Let's take a look at two algorithms that employ those uses:

```
sum = 0
counter = 0
answer = 'yes'
```

```
while answer is 'yes':
    get the next_value
    counter = counter + 1
    sum = sum+next_value
    ask user if he/she
        wants to continue
```

```
average=sum/counter
```

```
sum = 0
counter = 0
next_value = 0
```

```
while next_value is not
-1000
```

```
    counter = counter + 1
    sum=sum+next_value
    get the next_value
```

```
average=sum/(counter-1)
```


8.3 Common loop patterns: interactive, sentinel

Let's take a look at two algorithms that employ those uses:

```
sum = 0
counter = 0
answer = 'yes'
```

```
while answer is 'yes':
    get the next_value
    counter = counter + 1
    sum = sum+next_value
    ask user if he/she
        wants to continue
```

```
average=sum/counter
```

```
sum = 0
counter = 0
next_value = 0
```

```
while next_value is not
-1000
```

```
    counter = counter + 1
    sum=sum+next_value
    get the next_value
```

```
average=sum/(counter-1)
```

8.3 Common loop patterns: interactive, sentinel

Indefinite loops

In both approaches it is necessary to pay attention to the following details:

- *make sure that the loop runs exactly as many times as it is correct (not more and not less)*
- *check what happens if there will be 0 loop iterations*
- *check what happens if there will be 1 loop iterations*
- *check what happens if there will be maximum possible for this problem number of loop iterations*
- *make sure that the loop terminates*
- *test on different inputs*

8.3 Common loop patterns: interactive, sentinel

Indefinite loops

See the previous example implemented with exceptions:

[average_i_mod_exceptions.py](#)

and

[average_s_mod_exceptions.py](#)

8.3 Common loop patterns: interactive, sentinel

Work with files

Let's write a program that finds the average of all numbers in a file. We assume that the numbers are typed into a file one per line.

Here is an example of the content of the file `input.txt`

```
1  
5  
4  
9  
2  
12  
39  
0
```

8.3 Common loop patterns: interactive, sentinel

Work with files

Let's write a program that finds the average of all numbers in a file. We assume that the numbers are typed into a file one per line.

Algorithm / design:

```
print an explanatory notice
take a file name from the user
open file
counter = 0
for loop over lines in the file
    covert the line into a float/integer
    add to the sum
    increment the counter
find the average
display the average
```

8.3 Common loop patterns: interactive, sentinel

Work with files

Let's write a program that finds the average of all numbers in a file. We assume that the numbers are typed into a file one per line.

Modification:

if a file contains character other than numbers, then they will be ignored.

see [average_file_mod.py](#)

8.3 Common loop patterns: interactive, sentinel

Work with files

Let's write a program that finds the average of all numbers in a file. We assume that the numbers are typed into a file one per line.

Modification:

if a file contains character other than numbers, then they will be ignored.

Can we use while loop?

8.3 Common loop patterns: interactive, sentinel

Work with files

Let's write a program that finds the average of all numbers in a file. We assume that the numbers are typed into a file one per line.

Modification:

if a file contains character other than numbers, then they will be ignored.

Can we use while loop? Yes

see [average_file_mod_while.py](#)