CS180 - More on while loops (Ch 5)

Note Title 2/8/2012

Announcements

- HW3 due Monday at 11:59 pm

  hint: index

  ```
  mystring[2:5]
  ↳ mystring[5:2:-1]
  ```

A better way: Sequential Search

```
i = 0
found = False

while  i < len(data) and not found:
    if data[i] == val:
        found = True
    else:
        i += 1

print found
```

In strings, '' == False, any other string is True

Another example: user input

```
guests = []
name = raw_input('Enter a name, or hit
                  enter to end: ')
while name: ~~!= ''~~ :
    guests.append (name)
    name = raw_input('Enter a name, or hit
                      enter to end: ')

print 'You entered', len(guests), 'guests.'
```

Sanity check

What is the output of the following:

```
val = 5
while (val < 10):
      val += 1
      print val
print 'done'
```

val = ~~5~~ ~~6~~ ~~7~~ ~~8~~ ~~9~~ 10

Output:

```
6
7
8
9
10
done
```

Caution: Infinite loops

Consider:

```
while True:
    print 'Hello'
```

Output? Hello
⋮

Infinite loops

The previous example looked silly.
What about this!?

```
data = [ - - - ]
i = 0
found = False
while i < len(data) and not found:
        if data[i] == value:
              found = True
need --> i += 1
```

Can be tricky. Think Euclid's algorithm -
what if v never = 0?

Continue & Break

- 2 commands that only work in a loop

Continue: end current iteration of loop & jump to next one

Break: end current iteration and exit the loop

Can be overused !!

while condition
  for . . . . .
    :
    if condition2
      break;

Ex:

```
while condition :
    if condition1 :
        if condition2:
            do something
        else:
            do something else
            if conditionB :
                continue
        else:
            another Something
            if condition4 :
                break
    else:
        do something
```

done w/ loop

Back to seq. search

```
data = [O - - . . . . ]
value = raw_input('Enter item to search for: ')
found = False

for item in data:
    if item == value:
        found = True
        break

print found
```

Practice:

5.3: Write a program that prompts the user to enter numbers + prints the average. Program should keep taking input until the user enters 0, & output should be a float.

Ex:

```
Enter a number: 5
Enter a number: 17
Enter a number: 14
Enter a number: 0

The average is 12.0
```