while loops

Announcements
- Homework #5 due Thursday after break

Outline
- Warm-up
- Review mid-semester feedback
- while loops
- Loop patterns
- Programming practice

Warm-up
Write a function called distances that takes a list of distances in kilometers and converts each distance to meters and computes the average distance in meters. The function should return the list of distances in meters and the average distance in meters. The original list should not be modified.

```python
def distances(dists_km):
    dists_m = []
    total = 0
    for km in dists_km:
        m = km * 1000
        dists_m.append(m)
        total = total + m
    avg = total/len(dists_km)
    return (dists_m, avg)
```

while loops
- What is the purpose of a for loop? --- to do something repeatedly
- How many times is the body of a for loop executed? --- once for each item in the list over which the for loop iterates
  - E.g., for i in range(4) executes 4 times
  - E.g., for i in [1,2,3] executes 3 times
  - E.g., for c in "hello" executes 5 times
- We know how many times the loop will execute before the loop starts
- Another way to do repetition: while loop
- Use a boolean expression to determine whether or not the loop executes again
  - Similar to using a boolean expression in an if statement to determine whether to execute the if-body or the else-body
- Example
  ```python
count = 0
while count < 5:
    count = count + 1
    print(count)
print("Done")
```
The body of the loop must change the value of one or more variables used in the loop condition such that eventually the loop condition is false.

What happens if the loop condition is always true? --- the loop runs forever; called an infinite loop.

What is the output of each of the following programs?

a) total = 1
   while total < 16:
       total = total * 2
       print(total)
   Output:
   2
   4
   8
   16

b) a = 6
   b = 5
   while a >= 0 and b > 0:
       print(a, b)
       a = a - 2
       b = b - 1
   Output:
   6 5
   4 4
   2 3
   0 2
c) primes = [2, 3, 5, 7]
i = 0
while i < len(primes):
    print(primes[i])
i += 1
Output:
2
3
5
7
d) i = 1
while i < 5:
    if i % 3 == 1:
        i = i + 1
    elif i % 3 == 2:
        i = i - 1
    print(i)
Repeat forever:
2
1

- Rewrite each of these programs to use a while loop instead of a for loop.
  a) for i in range(5):
     print(i*2)
  Rewritten:
i = 0
while i < 5:
    print(i*2)
i += 1
  b) values = [95, 90, 85, 80]
  result = 0
  for v in values:
      result = result + v
  print(result/4)
  Rewritten:
values = [95, 90, 85, 80]
result = 0
i = 0
while i < len(values):
    result = result + values[i]
i = i + 1
print(result/4)
Loop patterns

- Accumulator pattern
  - Applying the pattern
    - Assign an initial value (e.g., 0 or "" to a variable) before the loop
    - Update the variable's value in the body of the loop
    - After the loop finishes (i.e., all iterations have completed), the variable should hold the final desired value
  - Use cases
    - Calculating a total
    - Building a string
    - Building a list

- Sentinel pattern
  - You may want to prompt a user for input multiple times, but you don’t know how many times to prompt the user --- use a while loop
  - To signal they are done entering input, the user enters a predetermined value (e.g., 0) called the sentinel value --- the loop should terminate when the user enters this value
  - Applying the pattern
    - Assign the value true to a variable (called the sentinel variable) before the while loop
    - Use the sentinel variable as the condition for the while loop
    - In the loop body, prompt for input
    - If the user entered the sentinel value, re-assign the value false to the sentinel variable; Otherwise, process the input as desired
  - Example
    ```python
    sentinel = True
    while sentinel:
        num = int(input("Enter a positive number (or -1 if finished):"))
        if num == -1:
            sentinel = False
        else:
            print("You entered",num)
    print("Done")
    ```

- Polling pattern
  - You may want to prompt a user for input and keep re-prompting until they enter a valid value --- use a while loop
  - The loop condition should check whether the entered value is invalid
  - Applying the pattern
    - Initialize a variable (called the input variable) to an invalid input value (e.g., ")
    - The condition for the while loop should check if the value is invalid (e.g., is not a specific character)
    - In the loop body, prompt for input and assign the input to the input variable
  - Example
    ```python
    choice = ""
    while choice not in ['a','b','c']:
        choice = input("Do you want a, b, or c? ")
    print("You chose " + choice)
    ```
If checking whether the input is valid is complex, then create a function that checks if the input is valid and put a call to this function as the condition for the while loop.

```python
def is_valid(user_choice):
    if user_choice in ['a','b', 'c']:
        return True
    else:
        return False
```

```python
choice = ""
while not is_valid(choice):
    choice = input("Do you want a, b, or c? ")
    print("You chose " + choice)
```

**Programming practice**

a) Discount Airlines, Inc. has asked you to write a Python program to help its customers calculate how many reward miles they have earned. Your program should repeatedly ask the user for the distance of a flight until the user enters 0. Then your program will output the total number of miles flown and whether the customer achieved bronze (less than 20,000 miles flown) or silver (at least 20,000 miles flown) status.

Your output must exactly match the format of this example:

```
What was the distance of flight 1? 1000
What was the distance of flight 2? 5000
What was the distance of flight 3? 2000
What was the distance of flight 4? 0
You flew 8000 miles.
You earned bronze status.
```

```python
moreFlights = True
total_distance = 0
count = 0
while moreFlights:
    dist = int(input("What was the distance of flight " + str(count + 1) + "? "))
    if dist == 0:
        moreFlights = False
    else:
        count = count + 1
        total_distance = total_distance + dist
print("You flew", str(total_distance), "miles.")
if total_distance < 20000:
    print("You earned bronze status.")
else:
    print("You earned silver status.")
```
b) Write a program that calculates the total purchase amount charged by an online retailer. The program should ask for the merchandise subtotal and the desired shipping speed, then calculate the shipping cost and the applicable tax (both merchandise and shipping are taxed) and finally, display the total cost.

*Standard shipping costs $8, 2-day shipping $16 and next day $25. Customers get free standard shipping if their merchandise total is over $75 and they choose standard shipping. The program should re-prompt the user for their shipping choice if the input does not match one of the available shipping options. The tax rate that should be used is four percent.*

*Your output must exactly match the format of this example:*  
Order sub-total: 82.93  
Select Shipping Speed: (S)tandard, (2)-Day, or (N)Next Day E  
Invalid choice  
Select Shipping Speed: (S)tandard, (2)-Day, or (N)Next Day S  
The total cost is: $86.25  

```python  
speed = input("Select Shipping Speed: (S)tandard, (2)-Day, or (N)Next Day ")  
while speed not in ['S', 'D', 'N']:
    print("Invalid choice")
    speed = input("Select Shipping Speed: (S)tandard, (2)-Day, or (N)Next Day ")  
if speed == "S":  
    if subtotal > 75:
        ship_cost = 0  
    else:
        ship_cost = 8.00  
elif speed == "2":  
    ship_cost = 15.00  
elif speed == "N":  
    ship_cost = 25  
total = subtotal + ship_cost  
total = total + total * 0.04  
print("The total cost is: $" + str(round(total,2)) )
```

c) Extend the above program to ask the user for the cost of each item until they enter a cost of zero. Compute the merchandise subtotal based on the cost of each item.

```python  
subtotal = 0  
cost = 1  
while cost > 0:
    cost = float(input("What is the cost of item " + str(count + 1) + "? "))  
subtotal += cost
```
d) Write a function called sums that takes a list of sublists of integers and replaces each sublist with the sum of the numbers in the sublist. The function should return the maximum sum and the minimum sum.

```python
def sums(lists):
    for i in range(len(lists)):
        total = 0
        sublist = lists[i]
        for num in sublist:
            total += num
        lists[i] = total
    max_sum = max(lists)
    min_sum = min(lists)
    return (max_sum, min_sum)
```