Today's "Menu"

(week 6)

- Announcements
- Solutions to 3 practice functions
- Look over some sample code
- **Introduction to Loops**
- Questions?

Announcements

- Do not write nested functions
- Exam II will deal with loops … just FYI
- A program a day keeps the bugs away :-)

- Discuss sample solution
- Discuss sample code

How to improve your code

- Readability counts – someone *will* read your code (and comments)
- Group related code together, separate by blank lines from others, expressions need blanks too.
- Separate methods from each other by @ 3 blank lines
- Don't "hack" .. "think/design" before you code.
- Don't forget about testing – you know can do something about problems (with if + error messages)

Repeating Code

- Loops enable us to repeat code
- Powerful construct useful for
  - Dealing with user IO (e.g., giving the user another chance to enter data)
  - processing large amounts of data (e.g., from a file)
  - Repeating things in general if the algorithm calls for it.

2 Types of Loops

- While-loop
  - Used when we "don't know ahead of time how often to repeat" action. Examples?
- For-loop
  - Used when we "know how many times to repeat something". Examples?
- Both loops are really equivalent

While Loop

```
while <Boolean exp true>:
    statement(s)
```

- Will "enter" the loop if the expr is True, else "skip" around it.
- Will stay inside the "body" of the loop until expr is False.
- Danger of ∞ loop! Or "off-by-one" error.

Examples (1)

```
print 'hello'
count = 0

while count < 5:
    print "this is line %d" % count

print 'the end.'
```

Output: ??

Examples (2)

```
print 'hello'
count = 0

while count < 5:
    print "this is line %d" % count
    count = count + 1   # count += 1

print 'the end.'
```

Output: ??

Examples (3)

```
print 'hello'
count = 5

while count < 5:
    print "this is line %d" % count

print 'the end.'
```

Output: ??

```
DAYS_IN_YEAR = 365

print 'Program will compute approximately how old you are in days'
print '(without considering partial or leap years)\n‘

stop = False

while (not stop):
   name = raw_input('What is your name? ')
   age = input('How old are you (in years)? ')          # What problems with this
                                                        # code?
   days_lived = age * DAYS_IN_YEAR

   print ('Dear %s, you are approximately %d days old.' %
          (capitalize(name), days_lived))

   answer = raw_input('\nEnter "x" to stop: ')
   if answer == 'x':
      stop = True

print '\nEnding program\n'
raw_input('<enter> to exit')
```


Problems

1. Case-sensitive! “X” won’t quit the program, only “x” will – poor interface design.
2. It is still possible to enter negative years! Poor (unacceptable) functionality.
3. We need to be concerned about both user interface and proper functionality. One without the other is no good.

Fix Problems

1. Case-sensitive! “X” won’t quit the program
2. It is still possible to enter negative years!

FIXES!

1. String function to change input to lower case
2. Use ‘If-statement’ to check.


```
while (not stop):
   name = raw_input('What is your name? ')
   age = input('How old are you (in years)? ')

   if age >= 1:
      days_lived = age * DAYS_IN_YEAR

      print ('Dear %s, you are approximately %d days old.' %
            (capitalize(name), days_lived))
   else:
      print ('\nYou entered "%d" for age.' % age)  # note blank line
      print ('Please enter a value >= 1')          # specific/helpful error msg

   answer = raw_input('\nEnter "x" to stop: ')
   if answer.lower() == 'x':   # note use of string functions
      stop = True

print '\nEnding program\n'
```

Basic For-Loop

- Use a for-loop to repeat statements a specific number of times.

```
for n in [0, 1, 2, 3, 4]:  # this is a list
        print n   # will print 0 to 4
```

- Same as

```
for n in range(5):   # generates a list
        print n   # will print 0 to 4, not(e) "5"
```

- Can change start/end/increment values

range + lists (1)

- Built-in function range() creates lists
- Lists are fundamental Python data types – extremely useful!

Examples: `range([start],end-1,[step])`

```
>>> range(4)
[0, 1, 2, 3]
>>> range(10, 15)
[10, 11, 12, 13, 14]
>>> range(10, 20, 2)
[10, 12, 14, 16, 18]
>>> range(10, 5)
[]  # note – an empty list
```

range+lists(2)

```
>>> myList = range(1, 31)
>>> print myList
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
   19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]

for i in myList:
   print i, i*i

range(-10, 0, 1) = ?

range(10, 29, 3) = ?

range(5) == range(0, 4, 1) ??
```


While + For

- Count from 1 to 10 with both loops:

```
i = ??          # should not use range() with while!
while i < ??:
  print i

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

Let's write a program: BMI

Summary/Recap

- Questions?