LING 408/508: Programming for Linguists

Lecture 24
November 30th
Administrivia

• You should be working on your programming project

• Homework 11 today
  – optional
  – easy opportunity to make up points
  – due Friday night by midnight
Last Time

- Chapter 6:
  - Cool program to interactively specify a triangle and calculate its perimeter

```python
3 def sq(x):
4     return x**2
5
def dist(p1,p2):
6     return math.sqrt(sq(p2.getX()-p1.getX())+sq(p2.getY()-p1.getY()))
7
def area(l1,l2,l3):
8     s = (l1 + l2 + l3) / 2.0
9     return math.sqrt(s*(s-l1)*(s-l2)*(s-l3))
```

```python
l1 = dist(p1,p2)
l2 = dist(p2,p3)
l3 = dist(p3,p1)
m.setText("Perimeter is %0.2f, area is %0.2f" % (l1+l2+l3, area(l1,l2,l3)))
```
Last Time

Perimeter is 15.08, area is 8.16

>>> import test
>>> test.main()
Last Time

- Chapter 7: Decision Structures
  - if <condition>:
  -     <body>
  - elif <condition>:
  -     <body>
  - else:
  -     <body>
  - try:
  -     <body>
  - except <error>:
  -     <body>
  - except:
  -     <body>
Today's Topics

• Chapter 8: Loop Structures and Booleans
conditional execution

- **variable**: `__name__`

File execution/import
- python file.py  `__name__` is '__main__'
- import file.py  `__name__` is 'file'

```python
if __name__ == '__main__':
    main()
```

This guarantees that `main` will automatically run when the program is invoked directly, but it will not run if the module is imported. You will see a line of code similar to this at the bottom of virtually every Python program.
Loops

• Already seen for loops:
  ```python
def main():
    n = input("How many numbers do you have? ")
    sum = 0.0
    for i in range(n):
      x = input("Enter a number >> ")
      sum = sum + x
    print "\nThe average of the numbers is", sum / n
  ```

• while <condition>:
  ```python
  for i in range(11):
    print i
  ```

  ```python
  i = 0
  while i <= 10:
    print i
    i = i + 1
  ```
Loops

Problem: keeps prompting for yes/no

```python
def main():
    sum = 0.0
    count = 0
    moredata = "yes"
    while moredata[0] == "y":
        x = input("Enter a number >> ")
        sum = sum + x
        count = count + 1
        moredata = raw_input("Do you have more numbers (yes or no)? ")
    print "\nThe average of the numbers is", sum / count

main()
```
Loops

A better program:
- uses `raw_input` and `eval`.
- While loop tests for empty string.

```python
def main():
    sum = 0.0
    count = 0
    xStr = raw_input("Enter a number (<Enter> to quit) >> ")
    while xStr != ":
        x = eval(xStr)
        sum = sum + x
        count = count + 1
        xStr = raw_input("Enter a number (<Enter> to quit) >> ")
    print "The average of the numbers is", sum / count

main()
```
def main():
    fileName = raw_input("What file are the numbers in? ")
    infile = open(fileName, 'r')
    sum = 0.0
    count = 0
    line = infile.readline()
    while line != "":
        sum = sum + eval(line)
        count = count + 1
        line = infile.readline()
    print "\nThe average of the numbers is", sum / count

main()
Loops

import string

def main():
    fileName = raw_input("What file are the numbers in? ")
    inFile = open(fileName,'r')
    sum = 0.0
    count = 0
    line = inFile.readline()
    while line != ":
        # update sum and count for values in line
        for xStr in string.split(line, ","):  
            sum = sum + eval(xStr)
            count = count + 1
        line = inFile.readline()
    print \nThe average of the numbers is", sum / count

main()
while True:
    number = input("Enter a positive number: ")
    if x >= 0: break  # Loop exit
print "The number you entered was not positive"
Conditions

• **bool**: True or False

```python
>>> 3 < 4
True
>>> 3 * 4 < 3 + 4
False
>>> "hello" == "hello"
True
>>> "hello" < "hello"
False
>>> "Hello" < "hello"
True
```

**Javascript:**

- undefined, null, NaN, 0 and "" are false
- everything else with a value is true
comparisons

• Compound conditional:

```python
if x1 >= x2 >= x3:
    max = x1
else:
    # points are different
```
comparisons

• Default values using booleans:

```python
ans = raw_input("What flavor do you want [vanilla]: ")
if ans != ":
    flavor = ans
else:
    flavor = "vanilla"
```

*with implicit coercion of string to bool*

```python
ans = raw_input("What flavor do you want [vanilla]: ")
if ans:
    flavor = ans
else:
    flavor = "vanilla"
```
comparisons

• Default values using booleans:

```python
ans = raw_input("What flavor do you want [vanilla]: ")
flavor = ans or "vanilla"
```

a one line wonder! (original code had 5 lines)

```python
flavor = raw_input("What flavor do you want [vanilla]: ") or "vanilla"
```
Homework 11

• Implement any two of the Programming Problems in Chapter 8 (pp. 262–264)
This program simulates a game of racquetball between two players called "A" and "B". The abilities of each player is indicated by a probability (a number between 0 and 1) that the player wins the point when serving. Player A always has the first serve.

What is the prob. player A wins a serve? 0.60
What is the prob. player B wins a serve? 0.50
How many games to simulate? 100

Games simulated: 100
Wins for A: 73 (73.0%)  
Wins for B: 27 (27.0%)

Monte Carlo methods (or Monte Carlo experiments) are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. They are often used in physical and mathematical problems and are most useful when it is difficult or impossible to use other mathematical methods. Monte Carlo methods are mainly used in three distinct problem classes:[1] optimization, numerical integration, and generating draws from a probability distribution.
Chapter 9

• Racquetball simulation program
  – uses library module random

```python
>>> from random import randrange
>>> randrange(1,6)
3
>>> randrange(1,6)
5
>>> randrange(1,6)
5
>>> randrange(1,6)
1
>>> randrange(1,6)
4
>>> randrange(1,6)
2
```

```python
>>> from random import random
>>> random()
0.545146406725
>>> random()
0.221621655814
>>> random()
0.928877335157
>>> random()
0.258660828538
>>> random()
0.859346793436
```
Chapter 9

```python
# rball.py
# Simulation of a racquetball game. Illustrates use of random
# numbers and functions to implement top-down design.

from random import random

def main():
    probA, probB, n = getInputs()
    winsA, winsB = simNGames(n, probA, probB)
    printSummary(winsA, winsB)

def printIntro():
    # Prints an introduction to the program
    print "This program simulates a game of racquetball between two"
    print 'players called "A" and "B". The abilities of each player is'
    print "indicated by a probability (a number between 0 and 1) that"
    print "the player wins the point when serving. Player A always"
    print "has the first serve."
```
Chapter 9

```python
21 def getInputs():
22     # Returns probA, probB, number of games to simulate
23     a = input("What is the prob. player A wins a serve? ")
24     b = input("What is the prob. player B wins a serve? ")
25     n = input("How many games to simulate? ")
26     return a, b, n
27
28 def simNGames(n, probA, probB):
29     # Simulates n games of racquetball between players whose
30     # abilities are represented by the probability of winning a serve.
31     # Returns number of wins for A and B
32     winsA = winsB = 0
33     for i in range(n):
34         scoreA, scoreB = simOneGame(probA, probB)
35         if scoreA > scoreB:
36             winsA = winsA + 1
37         else:
38             winsB = winsB + 1
39     return winsA, winsB
```
```python
def simOneGame(probA, probB):
    # Simulates a single game of racquetball between two players whose
    # abilities are represented by the probability of winning a serve.
    # Returns final scores for A and B
    serving = "A"
    scoreA = 0
    scoreB = 0
    while not gameOver(scoreA, scoreB):
        if serving == "A":
            if random() < probA:
                scoreA = scoreA + 1
            else:
                serving = "B"
        else:
            if random() < probB:
                scoreB = scoreB + 1
            else:
                serving = "A"
    return scoreA, scoreB
```

Chapter 9
```python
def gameOver(a, b):
    # a and b are scores for players in a racquetball game
    # Returns true if game is over, false otherwise
    return a == 15 or b == 15

def printSummary(winsA, winsB):
    # Prints a summary of wins for each player.
    n = winsA + winsB
    print('nGames simulated:', n)
    print('Wins for A: %d (%.1f%%)' % (winsA, float(winsA)/n*100))
    print('Wins for B: %d (%.1f%%)' % (winsB, float(winsB)/n*100))

if __name__ == '__main__':
    main()
```