Test 2 Practice: Compsci 06

November 15, 16, 2010

Name: _____

NetID/Login: _____

PROBLEM 1 : (*Choices*)

Use words, not code for the first three problems.

1. Given two lists, how would you find the elements in common to both lists?

2. In writing code to store the names of students on the wait-list for a course the programmer uses a list instead of a set. What's a good reason to use a list for this task (rather than a set)?

3. You want to find which of the people in your email contacts list you sent the most email to last year. You have a log file of every email you sent. How would you use a dictionary in a Python program to help solve this problem?

PROBLEM 2 : (*Code Review*)

```
colors = ["red", "green", "blue", "aqua", "brown", "orange",
          "magenta", "violet", "white", "yellow"]
```

1. What does this code print?

```
for c in [x for x in colors if len(x) < 5]:
    print c
```

2. What is printed by the code below:

```
def rainbow(c):
    return c in ["red", "orange", "yellow", "green", "blue", "indigo", "violet"]

for c in [x for x in colors if not rainbow(x)]:
    print c
```

3. What are the values stored in lists `combo` and `bunch` (these are tricky).

```
x = [1,2,3,4]
y = [5,6,7,8]

combo = [(a,y[i]) for i,a in enumerate(x)]
bunch = [(a,b) for a in x for b in y]
```

PROBLEM 3 : (*Simple Simon Met a PI-man*)

Part A

The Python list comprehension below is a list of all integers between 0 and 100, e.g., [0,1,2,...100].

```
[x for x in range(0,101)]
```

Write list comprehensions to represent each of the following:

1. all numbers between 500 and 600, inclusive

2. all multiples of three less than 100, e.g., [0,3,6,...,99]

3. all perfect squares less than 500, e.g., [0,1,4,9,16,25,...,484]

Part B

In the Jotto program there was a function named `commonCount` that returns the number of letters in common to two strings. Write a list comprehension that represents all the strings in a list `words` that have 3 letters in common with the string `"snipe"`

```
def commonCount(a,b):
    """
    you can call this function, it returns an int
    parameters are both strings
    """
```

PROBLEM 4 : (*What's the Point?*)

Write the function `maxDistance` that returns the maximal distance between two points in a list of points where points are represented as tuples.

For example the code below will print 19.209 which is the distance between `(-6,6)` and `(9,-6)` which are the two points that are farthest apart.

```
points = [(-7, 2), (-6, 6), (1, 2), (4, 5), (9, -6)]
print maxDistance(points)
```

Write your code below, you can call `distance`

```
def distance(a,b):
    """
    return distance between points represented as tuples
    e.g., between (a_0,a_1) and (b_0,b_1)
    """
    return math.sqrt((a[0]-b[0])**2 + (a[1]-b[1])**2)


def maxDistance(points):
    """
    return maximal distance between 2-tuples in points, a list of (x,y) points
    """
```

PROBLEM 5 : (*Banco Popular*)

Data for all students and their extra curricular clubs/organizations is stored in a file in the format below. Each club is named as the first item on a line of the file, and the net-ids of members of the club are given after the club's name.

```
chronicle,svp9,tlm,fro72
dsg,tlm,ola,ezp9
ski,fro72,frp9,tlm,ght
```

Complete the method below which reads this file and stores information in a global dictionary `clubs` in which the keys are net-ids and the value associated with each net-id is a list of the clubs the students with the given net-id belongs to. For example: in the file above the student with net-id `tlm` is a member of all three clubs, so that student's entry can be thought of as:

```
tlm = [chronicle,dsg,ski]
```

Complete the code below

```
clubs = {}    # global dictionary

def readFile(filename):



    file = open(filename)


    for line in file:









    file.close()
```

Part B

Write a list comprehension using global variable/dictionary `clubs` that represents a list of all netids for students who are members of more than 3 clubs.

Part C

Given only the dictionary write the function `mostMembers` that returns the name of the club that has more student members than any other club. Reading the file with the information would make this straightforward, but for this problem you have only the dictionary in which keys are netids and values are lists of clubs for the student with the given netid.

```
def mostMembers(clubs):
    """
    returns name of club that has the most members,
    using information in dictionary clubs
    """
```