A201/A597 Introduction to Programming I

First Summer 2007

Lecture Fifteen: Wednesday May 30, 2007 (ED1204)

Today the last of two days dedicated to the midterm exam.

The problems you need to focus on have been listed on Sat May 26 on-line:

Yesterday we discussed problems 1-5. Today we analyze problems 6-9.

6. scalable patterns (4, A, C, E, Q, M, W, F, H, K, N, Z, P, R, V, X, Y)
7. teach the computer to play the game of hangman with the user
8. jumble the word (generate a permutation of the user input)
9. sorting a list of integers (bubble sort, selection sort)

6. Scalable patterns (4, A, C, E, Q, M, W, F, H, K, N, Z, P, R, V, X, Y).

Since you know two-dimensional data structures now you no longer need to work out a strategy by which your output needs to be generated in one pass (as below). Still, the method we used originally is entirely general and thus very powerful. Here's an example:

```
height = int(raw_input("Size: "))

for row in range(height):
    for column in range(height):
        if column + row == height / 2 and row < height/2 \
           or column < 3 * height/4 and row == height / 2 \
           or column == height / 2 and row > height / 4:
            print "*",
        else:
            print " ",
    print
```

7. Teach the computer to play the game of hangman with the user.

One such program was developed in class and is included below.

Note that it doesn't use tuples, lists or dictionaries.

It would be an interesting exercise to see the extent to which this program could be simplified if we used any of those data structures. Meanwhile the standard solution is included below:

```
import random
words = ["abracadabra", "wonderful", \
         "everything", "python", "apple"]
```

```
secret = random.choice(words)

print "The secret word is:", secret
mask = '-' * len(secret)
print "                   ", mask
wrong = 0
already = ""
while mask != secret:
    letter = raw_input("Guess: ")
    if letter not in secret:
        wrong = wrong + 1
        if wrong == 6:
            break
    if letter in already:
        print "You typed this before"
        continue
    already = already + letter
    temp = ""
    for index in range(len(secret)):
        if secret[index] == letter:
            temp = temp + letter
        else:
            temp = temp + mask[index]
    mask = temp
    print "Typed so far:", already
    print mask
if wrong == 6:
    print "Too bad, the word was:", secret
else:
    print "Thank you very much for playing hangman with me."
```

8. There were two programs we discussed in class.

One of them uses a for loop:

```
import random
a = raw_input("Gimme: ")
for i in range(100):
    j = random.randrange(len(a))
    a = a[j] + a[:j] + a[j+1:]
print a
```

The other one further capitalizes on this idea by using a while loop to extract characters (randomly) from the given word, slicing the characters out until there are no more characters to extract, and placing them sequentially (in their random order) into a second string, that holds the result.

```
import random
word = raw_input("Gimme: ")
scrambled = ""
while word:
    position = random.randrange(len(word))
    scrambled += word[position]
```

```
        word = word[:position] + word[position+1:]
    print scrambled
```

Notice that `while word:` is the same as `while not word == "":` which is to say that the while loop will keep going for as long as there are letters in the source word.

9. The second method of scrambling a word (or, in general, of a string) leads into this method of organized selection for sorting:

```
import random

a = []
for i in range(10):
    a.append(random.randrange(10))
print "We start from this random list:", a

result = []
while a:
    m = min(a)
    result.append(m)
    a.remove(m)

print "Sorted, the list looks like this:", result
```

In class another sorting algorithm was developed (bubble sort, see below) starting from the definition of a sorted sequence and considering small, incremental changes in the desired direction:

```
import random

a = []
for i in range(10):
    a.append(random.randrange(100))
print a

sorted = False

while not sorted:
    sorted = True
    for i in range(len(a)-1):
        if a[i] > a[i+1]:
            sorted = False
            temp = a[i]
            a[i] = a[i+1]
            a[i+1] = temp
    print a
```

Minute paper: for each lecture program we will ask a starting question.

Labs will be a collection of extended minute papers (see yesterday's lab notes).

Note that we might ask how we can improve the programs we have (listed above).

Lab Eleven: Wednesday May 30, 2007 (ED2025)

The lab assignment for today is to attempt to develop on your own each of the problems discussed in the lecture notes. This makes today's lab a mock exam just like yesterday's lab.

Both Adrian and Metal will be present, so the lab will be staffed such as to allow you to get help while you plan your test programs. You may check your notes but only when you are sure you can't produce anything on your own and even then you shouldn't type code straight from them: you should look up one thought or idea, then close your notes, then try to implement that idea by yourself.

Note that the exam will be closed book, two problems, randomly distributed. The labs are to be an exercise, a simulated exam to give you feedback for additional review. In class we will discuss the problems, and the discussion might overflow into the lab time. Perhaps we will distribute the problems randomly to you in lab as well.

For full credit for today and tomorrow you need only attempt all the problems and turn them in. The problems don't have to be complete, they only need to be attempted with honesty. How else can you find out what you're missing to do well on the exam, if not by trying to see how far you can go on your own, under our guidance.

There are 12 lab assignments per semester (summer session). Today's and yesterday's are lab assignments 11 and 10 respectively; their purpose is to prepare you for the exam. Lab 12 (last) is the one on Thu. Its purpose is to ensure that you grade your own exam immediately after the class, giving you a chance to catch any mistakes and fix them.

Homework assignment(s).

See comments posted under yesterday’s set of notes.