A201/A597 Introduction to Programming I

First Summer 2007

Lecture Seven: Wednesday May 16, 2007 (ED1204)

Today we want to make sure we are entirely caught up with the problems worked thus far in lab. The new material will be presented as succinctly as possible, and one example will be worked out completely.

Strings are sequences. Their elements are characters. Strings (including the empty string) have a length. Each character in a string has a location, given by a zero-based index. One can extract one or more characters in a string by indexing (and slicing).

```
>>> a = "something"
>>> len(a)
9
>>> a[3]
'e'
>>> a[0]
's'
>>> a[len(a)-1]
'g'
>>> a[3:5]
'et'
>>> a[:5]
'somet'
>>> a[3:]
'ething'
>>>
```

Notice two aspects: a) strings are immutable, and b) for every character in the string there are two indices that locate it, one is zero-based and positive and expresses the number of characters it has to its left and the other is negative, starts at -1 and shows (in absolute value) how far away the character is from the end of the string. Slicing works with both types of indices.

```
>>> a = "nectarine"
>>> a[len(a)-1]
'e'
>>> a[-1]
'e'
>>> a[-2]
'n'
>>> a[-5:-2]
'ari'
>>> a[0]
'n'
>>> a[-len(a)]
'n'
>>> a[100]
```

```
Traceback (most recent call last):
  File "<pyshell#16>", line 1, in ?
    a[100]
IndexError: string index out of range
>>> a[2] = ' '
Traceback (most recent call last):
  File "<pyshell#14>", line 1, in ?
    a[2] = ' '
TypeError: object doesn't support item assignment
>>>
```

Is this the only type of sequence? No. Tuples and lists are also sequences.

The function range(…) returns a list. Lists are mutable sequences:

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> range(6, 18, 3)
[6, 9, 12, 15]
>>> a = range(10)
>>> a
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> a[4]
4
>>> a[4] = 100
>>> a
[0, 1, 2, 3, 100, 5, 6, 7, 8, 9]
>>> a[2:4]
[2, 3]
>>> a[2:4] = [9, 8, 7, 6]
>>> a
[0, 1, 9, 8, 7, 6, 100, 5, 6, 7, 8, 9]
>>>
```

This last operation is called a splice. We see lists are almost like strings but use square brackets for delimiters and commas as separators between elements. And are mutable.

We learned about while loops: they keep looping while a condition is true. Another (very popular) kind of looping construct works on sequences.

```
>>> for c in "nectarine":
      print c


n
e
c
t
a
r
i
n
e
>>>
```

It works by mapping a variable (in this case c) over the elements of a sequence, in order, one mapping at a time. For each such mapping the body of the loop is executed.

That's all we need to know for now.

Minute paper: calculate the sum of the first 100 positive integers.

Hint: first generate a sequence that contains these numbers, then use a for loop.

Let's work out an extensive example now. The chapter is developing a word jumble program: given a word (string of characters) from the user print back a permutation of the word. So no character is lost, the order is just random this time.

Here's a solution: for 100 times cut the string in two at a random position in the string. Take the two pieces and make a new string in which the pieces are reversed. It's like a crude type of shuffle. The program is presented below:

```
import random

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

Well, apparently this doesn't do a good job of shuffling. Can you see why?

What can we do?

Here's a fix:

```
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
```

Can you describe how it works and why it's working better than the previous one?

Lab Five: Wednesday May 16, 2007 (ED2025)

We will keep the lab notes extremely simple this time around.

Recall the first two problems in Homework One? You solved them with .replace(…).

It would be a good exercise to redo them by using for loops instead of string methods.

Here's the lab assignment for today: write a program that places the string "egg" in front of every vowel in a string given by the user. This is how your program might behave:

```
>>>
Gimme: Hi.
Heggi.

>>>
Gimme: Convoluted
Ceggonveggolegggutegged

>>>
Gimme: I am here.
I eggam heggeregge.

>>>
Gimme: This is a fun program.
Theggis eggis egga feggun preggogreggam.

>>>
Gimme: Thank you.
Theggank yeggoeggu.

>>>
```

Notice that the in operator in the for loop has two uses when it comes to sequences: first, the use that we have seen already, it helps for with the mapping; second, by itself, it helps determine whether an element belongs to a sequence or not. Here's an example:

```
>>> a = "nothing"
>>> "h" in a
1
>>> if "h" in a:
      print "Yes"
else:
      print "No"


Yes
>>>
```

So lab assignment for today is just one program (eggy-peggy, above) but feel free to experiment with strings and lists, indexing/slicing them and maybe even mutating them.

Homework Four

First, write an algorithm to settle the following question: A bank account starts out with `$10,000`. Interest is compounded at the end of every month at `6` percent per year (`0.5` percent per month). At the beginning of every month, `$500` is withdrawn to meet college expenses after the interest has been credited. After how many years is the account depleted?

Now suppose the numbers (`$10,000`, `6` percent, `$500`) were user-selectable. Are there values for which the algorithm you developed would not terminate? If so, make sure it always terminates.

Here's how my program works (I wrote my program a few years ago, in Java):

```
frilled.cs.indiana.edu%java Interests
Welcome to the financial calculator.
What's your initial balance? 10000
What's the yearly interest? 6
How much do you plan to withdraw monthly? 500
The account will last 1 year(s) and 9 month(s).
Ending balance will be: 62.2 dollars.
frilled.cs.indiana.edu%java Interests
Welcome to the financial calculator.
What's your initial balance? 10000
What's the yearly interest? 6
How much do you plan to withdraw monthly? 100
The account will last 11 year(s) and 6 month(s).
Ending balance will be: 97.09 dollars.
frilled.cs.indiana.edu%java Interests
Welcome to the financial calculator.
What's your initial balance? 10000
What's the yearly interest? 6
How much do you plan to withdraw monthly? 10
This will last forever.
frilled.cs.indiana.edu%java Interests
Welcome to the financial calculator.
What's your initial balance? 10000
What's the yearly interest? 6
How much do you plan to withdraw monthly? 50
The account will last 525 year(s) and 1 month(s).
Ending balance will be: 48.1 dollars.
frilled.cs.indiana.edu%
```

Note especially how my program detects when the funds can last forever.