List and Trees

Abhinav Ashar
CS 61A: Structure and Interpretation of Computer Programs

January 18, 2019

1 Lists

Lists are a powerful tool in Python used to store elements of different types. They are denoted by the brackets, are 0-indexed, and can contains different elements (one list can have an int, String, float etc.). The most important concept about lists is the idea of pointers, where the variable name for a list points to that list somewhere in memory.

a) The (+) sign
   (a) Example: a = a + [10]
   (b) Creates a new list
   (c) You cannot add a number to a list (Ex. a = a + 9 where a is a list)

b) .append()
   (a) Example: a.append([2,3])
   (b) Adds to old list
   (c) Add only one box, regardless of what is being added (array or number)
   (d) That one box should either contain a number if you appended a number, or a pointer to an array if you appended an array

c) .extend() or the (+=) sign
   (a) Example: a.extend([2,3]) or a += [2,3]
   (b) Adds to old list
   (c) Add the number of boxes equal to the length of the outer list. For example, if the array being added was [[[3,4,5,6,7,8]], 2], the length of the outer list is 2 so add 2 boxes
   (d) .extend() must contain an array as an argument (cannot be a number)

d) List Splicing
   (a) Example: a[1:4]

(b) Creates a new list, but...

(c) Does a shallow copy. The indices/contents of the outermost array are copied over, but if there are pointers to other arrays within the indices, those pointers point to the original lists, not a clone of them

(d) **Looking Ahead**: What is the difference between b = a and b = a[:]?

2 List Comprehension

List comprehension is just a concise and clean way of writing for loops within one line. Any list comprehension can be written as the longer version with a for loop. List comprehension also always returns an array. Here is the transition between the two:

Make an array of all the odd numbers in arr
result = []
for x in arr:
*** if x % 2 == 1: (Ignore the stars)
****** result += [x]

***** OR *****

[x for x in arr if x % 2 == 1]

3 Trees

Trees are one of the most commonly used data structures within all of computer science. A tree normally consists of a label/root and a list of branches. The bottom of trees are leaves, and trees are built of subtrees.

a) Leaves are also considered trees

b) You often use recursion and list comprehension to solve tree problems

c) Remember that you have often functions like label() and branches() to help you work with a tree