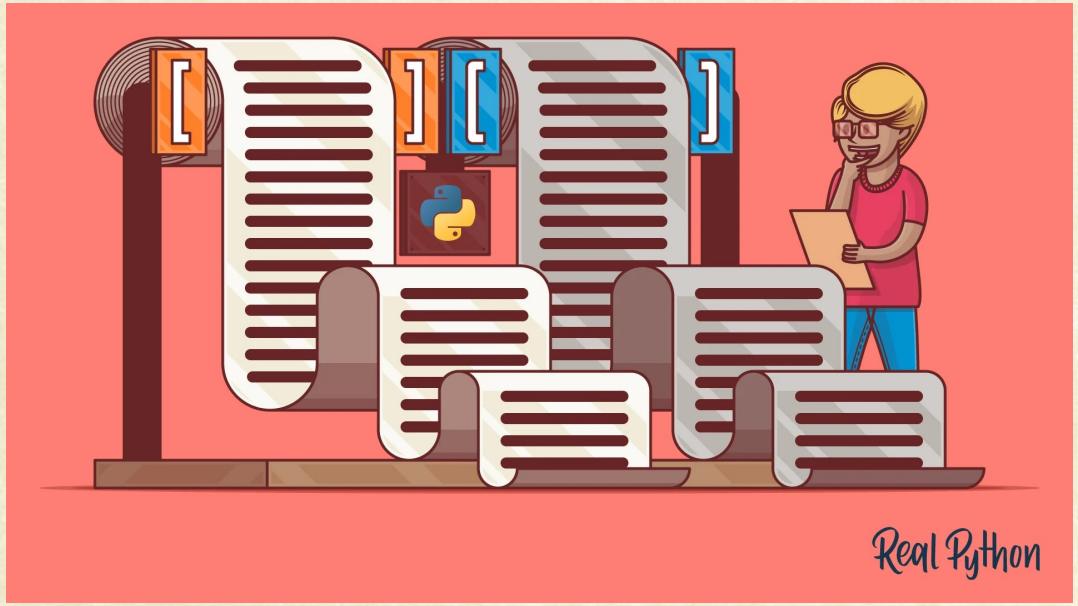
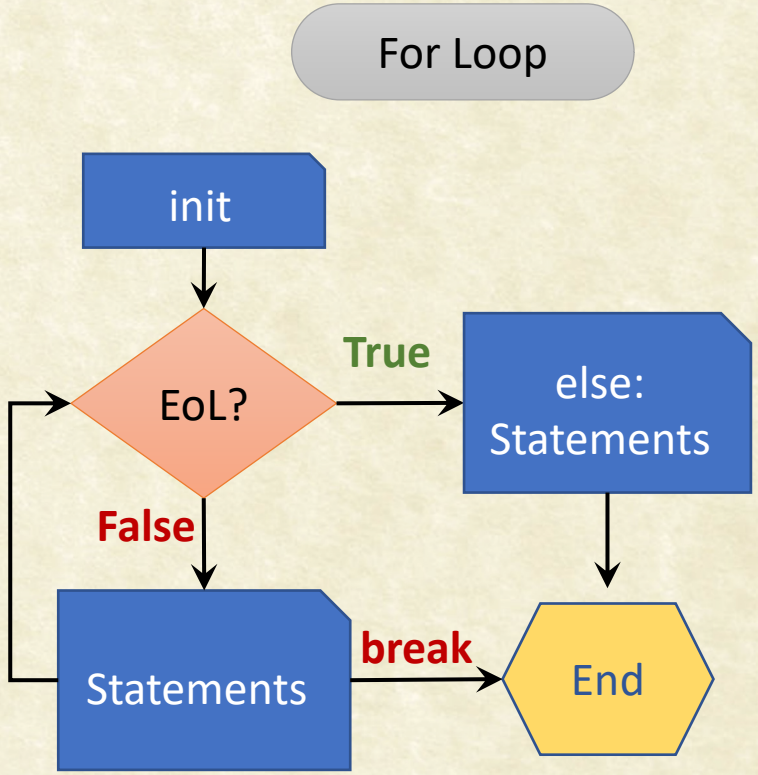
Lecture 5: Lists, For-Loop





Anoop M. Namboodiri

IIIT Hyderabad

Python Lists

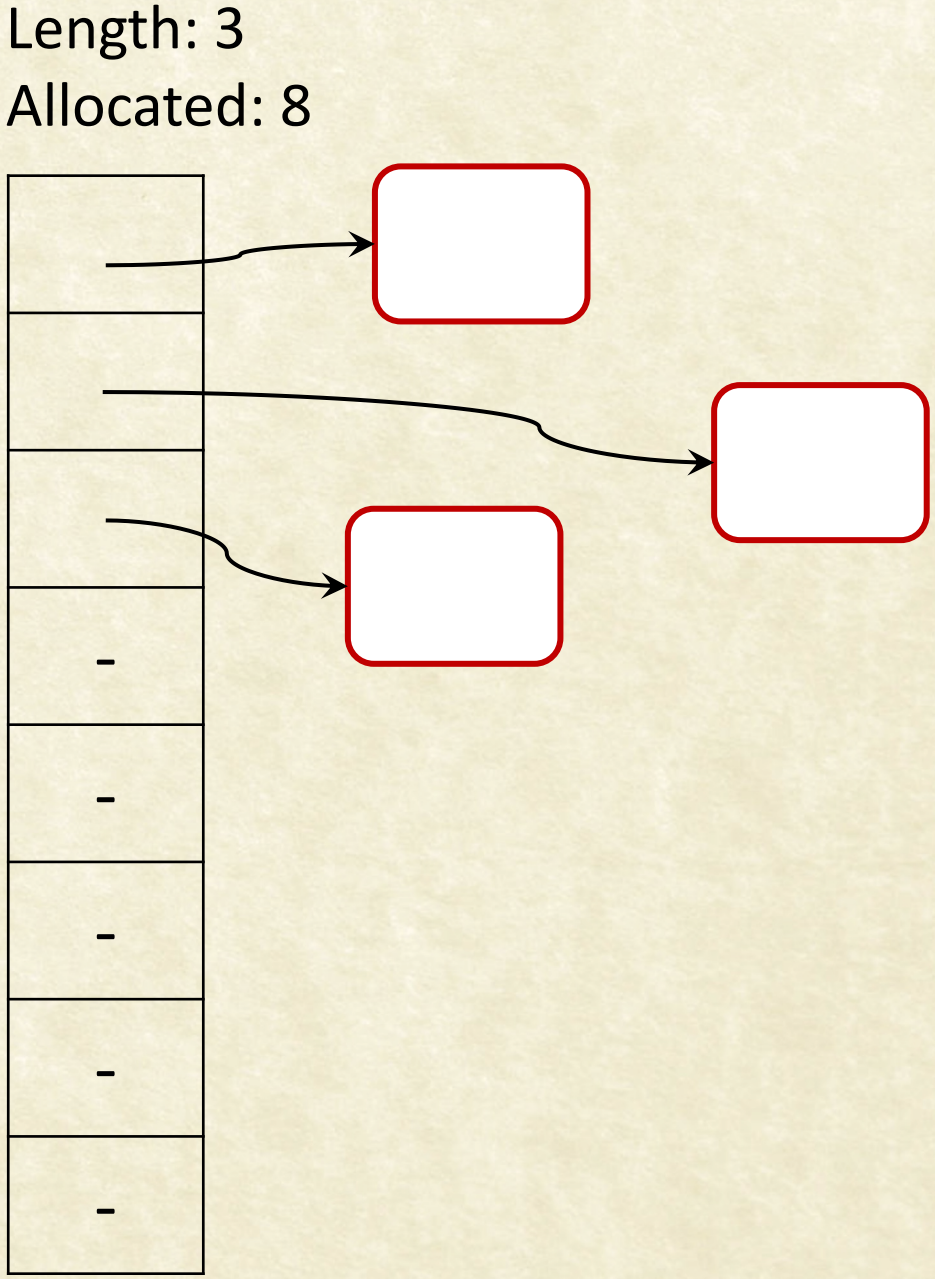
The most versatile data structure

List: Most Versatile Sequence Type

- Creating list1 = [elt1,elt2, .., eltn]
- Element access using list1[ ], just as in strings
- Elements need not be of the same type. List is a general form of arrays in c.
- Elements could in-turn be lists (multidimensional arrays)
- Like strings: Indexing, Slicing, Concatenation
- Elements can be added at the end using append or anywhere using assignment/insert (mutable type)
  - list1[3] = 'abc' or list1[2:2] = 'abc' (insertion)
- Deletion of an element: del list[3]

List Implementation

- As a dynamic array of pointers
- Array is resized when full; maintains *length* and *allocated*.
- Indexing is fast: Two accesses
- Search may not be (not in cache), unless sorted



Methods Related to List

- String to List and Vice versa
  - str.split('c') : splits str at character c to a list
  - c.join(lst): joins lst to string with c as delimiter
- lst.index(obj):
  - Returns lowest index of obj in lst (raises exception if not found)
- lst.count(obj):
  - Counts number of times obj occurs in str
- lst.insert(index,obj): inserts obj at position index
- lst.pop(), lst.remove(), lst.sort(), lst.reverse, lst.append(), lst.extend()
- Operators: +, +=, comparisons(==, !=, <, >, <=)
- max(lst), min(lst), len(lst)

Let us Code: List

- Write the code for inserting a number, num, into a sorted list, List1, at the correct position. Assume sorting to be in ascending order.

```
pos = len(List1)-1
while pos >= 0 and List1[pos] > num:
    pos -= 1
List1.insert(pos+1,num)
```

Coding Practice: List

1. Write the code for inserting a number, num, into a sorted list, List1, at the correct position. Assume sorting to be in descending order.
2. What if the sorting order is either ascending or descending, and you are supposed to insert at the correct position for both cases.
3. Carry out insertion sort using the above code.
4. Read the postal address from a user and find the pincode: a 6-digit number in the last line.

Python Loops

for: The versatile loop

for: Looping through Collections

```
for target-list  in expression-list:
     statements
else:
     statements
```

- The expression-list is evaluated once and should yeild an iterable object (say list). The statements in the for block is executed once for each item given by the iterator (each item in the list) in the ascending order of indices. When the items are exhausted (which might even be when the expression-list is evaluated), the statements in the else block (if present) are executed and the loop terminates.

Let us Code: for

- Write a code that prints all primes up to n (n>=2)

```
n=123
primes = [2]
for p in range(3,n+1,2):
    prm = True
    for p1 in primes:
        if p % p1 == 0:
            prm = False
            break;
    if prm:
        primes.append(p)
print primes
```

Let us Code: for

- Write a code that prints all primes up to n (n>=2): Use ‘for else’

```
n=123
primes = [2]
for p in range(3,n+1,2):
    for p1 in primes:
        if p % p1 == 0:
            break
    else:
        primes.append(p)
print(primes)
```

Coding Practice: for

1. Write a program that, given an integer n, finds three integers a, b and c, such that a+b+c = n and a*b*c is maximum.
2. What is the complexity of the most efficient version?