UNIVERSITY of WISCONSIN
LA CROSSE
COMPUTER SCIENCE

CS 224 Introduction to Python

Lists

1

Python Lists

Python's version of an array but some important differences:

- lists do not have a fixed size
- lists are heterogeneous
- can insert delete at arbitrary positions within list

2

Creating a List

```
my_list = []            # creates an empty list
my_list = list()        # creates an empty list
my_list = [1, 2, 3]     # creates a list containing
                        # elements 1, 2 and 3
list2 = my_list         # creates a reference to my_list

my_list = [1, 3.14, 'test', ['a', 'b']]
                        # note different types

s = 'abc'
my_list = list(s)       # creates list ['a', 'b', 'c']
```

3

Accessor

```
nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

nums[5]                 # value is 6
nums[len(nums)-1]       # value is 10
nums[-1]                # value is 10
nums[-2]                # value is 9
```

Accessor results in an element

4

Slices

```
nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

nums[3:6]                # value is [4, 5, 6]
t = nums[4:]             # t is [5, 6, 7, 8, 9, 10]
u = nums[:7]             # u is [1, 2, 3, 4, 5, 6, 7]
v = nums[-4:]            # v is [7, 8, 9, 10]
w = nums[:-5]            # w is [1, 2, 3, 4, 5]

Slices are lists
```

5

List Operations

```
• len(my_list)      # number of elements
• min(my_list)      # minimum value
• max(my_list)      # maximum value
• sum(my_list)      # sum of the values
• all(my_list)      # True if all elements True
• any(my_list)      # True if any element True
```

6

More List Operations

```
• my_list[i] = x          # element assignment
• my_list[i:j] = u_list   # replaces slice with
                          # another list
• del my_list[i]          # delete ith element
• del my_list[i:j]        # delete slice

• u_list + v_list         # concatenate
• e in my_list            # True if e is element in my_list
```

7

Methods to add to a list

```
my_list.append(e)
     adds element e to end of my_list

my_list.extend(u_list)
     concatenates u_list onto my_list

my_list.insert(i, e)
     inserts element e at position i
```

8

Methods to remove from a list

```
my_list.pop([i])
    returns element at position i and removes it
    from my_list. if no parameter given, returns
    and removes last element.

my_list.remove(e)
    searches for e in my_list and removes if found.
    if multiple occurrences, removes only one.
```

9

Methods to rearrange a list

```
my_list.reverse()
    reverses elements of my_list in place.

my_list.sort()
    sorts elements of my_list in place.

my_list.sort([key, [, reverse]])
    sorts elements of my_list in place. key
    is a sort-key function. reverse is an optional
    Boolean flag.
```

How is reverse different from sort with reverse flag?

10

Other list methods

```
my_list.count(e)
     returns number of occurrences of e in my_list.

my_list.index(e [, start [, stop]])
     returns the smallest index i for which
     my_list[i] == e.  optional parameters start and
     stop constrain the search.

sorted(my_list [, key [, reverse]])
     returns a sorted version of my_list but does
     not change my_list.
```

11