Appendix D: List Operators and Methods

11

This documentation was generated from the Python documentation available by typing *help*(*list*) in the Python shell. In the documentation found here the variables *x* and *y* are references to lists (Table 11.1).

Table 11.1 List operators and methods

Method	Returns	Comments
list()	list	Returns a new empty list. You can also use `[]` to initialize a new empty list
list(sequence)	list	Returns new list initialized from sequence's items
[item [,item]+ **]**	list	Writing a number of comma-separated items in square brackets constructs a new list of those items
x+y	list	Returns a new list containing the concatenation of the items in x and y
e in x	bool	Returns True if the item e is in x and False otherwise
del x[i]		Deletes the item at index i in x. This is not an expression and does not return a value
x==y	bool	Returns True if x and y contain the same number of items and each of those corresponding items are pairwise equal
x>=y	bool	Returns True if x is greater than or equal to y according to a lexicographical ordering of the elements in x and y. If x and y have different lengths their items are == up to the shortest length, then this returns True if x is longer than y
x<=y	bool	Returns True if x is lexicographically before y or equal to y and False otherwise
x>y	bool	Returns True if x is lexicographically after y and False otherwise
x<y	bool	Returns True if x is lexicographically before y and False otherwise
x!=y	bool	Returns True if x and y are of different length or if some item of x is not == to some item of y. Otherwise it returns False

(continued)

K.D. Lee, *Python Programming Fundamentals*,
Undergraduate Topics in Computer Science, DOI 10.1007/978-1-4471-6642-9_11

Table 11.1 (continued)

Method	Returns	Comments
x[i]	item	Returns the item at index i of x
x[[i]:[j]]	list	Returns the slice of items starting at index i and extending to index j−1 in the string. If i is omitted then the slice begins at index 0. If j is omitted then the slice extends to the end of the list. If i is negative then it returns the slice starting at index len(x)+i (and likewise for the slice ending at j)
x[i]=e		Assigns the position at index i the value of e in x. The list x must already have an item at index i before this assignment occurs. In other words, assigning an item to a list in this way will not extend the length of the list to accommodate it
x+=y		This mutates the list x to append the items in y
x*=i		This mutates the list x to be i copies of the original x
iter(x)	iterator	Returns an iterator over x
len(x)	int	Returns the number of items in x
x*i	list	Returns a new list with the items of x repeated i times
i*x	list	Returns a new list with the items of x repeated i times
repr(x)	str	Returns a string representation of x
x.append(e)	None	This mutates the value of x to add e as its last element. The function returns None, but the return value is irrelevant since it mutates x
x.count(e)	int	Returns the number of occurrences of e in x by using == equality
x.extend(iter)	None	Mutates x by appending elements from the iterable, iter
x.index(e,[i,[j]])	int	Returns the first index of an element that == e between the start index, i, and the stop index, j−1. It raises ValueError if the value is not present in the specified sequence. If j is omitted then it searches to the end of the list. If i is omitted then it searches from the beginning of the list
x.insert(i, e)	None	Insert e before index i in x, mutating x
x.pop([index])	item	Remove and return the item at index. If index is omitted then the item at len(x)−1 is removed. The pop method returns the item and mutates x. It raises IndexError if list is empty or index is out of range
x.remove(e)	None	remove first occurrence of e in x, mutating x. It raises ValueError if the value is not present
x.reverse()	None	Reverses all the items in x, mutating x
x.sort()	None	Sorts all the items of x according to their natural ordering as determined by the item's __cmp__ method, mutating x. Two keyword parameters are possible: key and reverse. If reverse = True is specified, then the result of sorting will have the list in reverse of the natural ordering. If key = f is specified then f must be a function that takes an item of x and returns the value of that item that should be used as the key when sorting