CISS450: Artificial Intelligence
Lecture 6: Lists

Yihsiang Liow

Agenda

- Study lists which is Python's version of arrays

Lists

- Lists are heterogeneous array structures that can be arbitrarily nested
- They are mutable
- Lists are under the larger class of ***sequence types***. Strings are also sequences. Therefore lists share many functions/methods of string objects.

First Examples

- Example:

```
xs = [1,2,3, "four", "five", 6.0, 7.0]
print(xs, type(xs))
xs = []
print(xs, type(xs))
```

Generating Integer lists

- Example:

```
x = range(10)
print(list(x))
x = range(10, 20)
print(list(x))
x = range(10, 20, 2)
print(list(x))
```

Type Conversion

- You can convert a string to a list:

```
hw = "Hello, World!"
hwlist = list(hw)
print(hwlist)
```

Operators

- Example

```
x = ["Hi", 3, 2.1]
print(x + x)
print(2 * x, x * 3)
print(x == x, x != x)
print(x == [], x != [])
print([1, 2]==[2, 1])
print(x[0], x[1], x[1:3], x[:])
y = x
x[0] = "Yo"
print(x, y)
print(id(x), id(y))
```

Membership and Search

- Example:

```
xs = [3, 10, 4, 5, 7, 34, 4, 10]
print(3 in xs)
print(3 not in xs)
print(xs.index(34))
print(xs.index(35))
```

Inserts

- You can append an element or a list to the end of a list, insert at an index position, overwrite slices:

```
x = [2, 3, 5]
x.append(7)
print(x)
x.extend([11, 13, 17])
print(x)
x.insert(2, "oops")
print(x)
x[2:8:2] = ["a", "b", "c"]
print(x)
```

Deletion

- You can delete the last element, or the element at an index, or a whole slice:

```
xs = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
x = xs.pop()
print(xs, x)
del xs[0]
print(xs)
del xs[2:5:2]
print(xs)
del xs[2:]
print(xs)
```

Lists in Lists

- You can have lists within lists:

```
x = [1, 2, 3]
y = ["one", "two"]
z = [x, y]
print(z)
```

- Therefore you can simulate multi-dim data with lists:

```
vector = [0, 1, 2]
matrix = [vector, vector, vector]
print(matrix, matrix[0][1])
```

Lists in Lists

- But be careful!

```
vec = [0, 1, 2]
matrix = [vec, vec, vec]
print(matrix, matrix[0][1])
matrix[0][0] = 42
print(matrix)
```

- Now try this instead:

```
vector = [0, 1, 2]
matrix = [vec[:], vec[:], vec[:]]
print(matrix, matrix[0][1])
matrix[0][0] = 42
print(matrix)
```

Useful Functions/Methods

```
xs = [3,1,2,1,6,4,1,2,8]
xs.reverse()
print(xs)
xs.sort()
print(xs)
print(len(xs))
m, M = min(xs), max(xs)
print(m,M)
print(xs.count(1))
```

Sort

- You can modify the way a list is sorted by providing a comparison function

```
def key(x):
    return -x

xs = [1, 3, 5, 2, 4, 6]
xs.sort()
print(xs)
xs.sort(key=key)
print(xs)
```

- Check Python documentation for another example

Important Gotcha

- Try:

```
matrix0 = [[],[],[]]
matrix1 = 3*[[]]
matrix0[0].append(1)
matrix1[0].append(1)
print(matrix0)
print(matrix1)
```

- Explain!

Extracting Values into Variables

- Try:

```
xs = ["john", 5.85]
name, height = xs
```

- This is called unpacking.
- Does this work?

```
xs = ["john", 5.85, "doe"]
name, height = xs
```

Zipping

- Try this:

```
names = ["john", "jane", "tom"]
heights = [5.85, 5.73, 5.56]
xs = zip(names, heights)
print(xs)
print(list(xs))
```

Python Documentation

- Make sure you check the Python documentation for sequence types and lists in particular