Working with Lists

GIS 5653 – Spatial Programming and GIS

COLLEGE OF ATMOSPHERIC AND GEOGRAPHIC SCIENCES
DEPARTMENT OF GEOGRAPHY
AND ENVIRONMENTAL SUSTAINABILITY
The UNIVERSITY of OKLAHOMA

List – Data Structure

Sequence of values

→ compare **string**: sequence of characters

- Values in a list are called **elements** or **items**
- Examples:

```
cities = ['Berlin', 'London', 'Paris']    # string
population = [3500000, 8308000, 2211000] # int
random_list = [30, 4.5, 'Hello', 7]   # different data types
city_data = [cities, population]    # lists
```

- Values in list can have any data type
- List within list → nested

Empty list: `empty_list  = []`

Indexing

- Bracket operator `[index]` → use index to access elements in a list
- Each index maps to one of the elements

```
cities = ['Berlin', 'London', 'Paris']
```

'Berlin'	'London'	'Paris'
[0]	[1]	[2]
[-3]	[-2]	[-1]

```
city = cities[1]
print(city)
city = cities[-1]
print(city)
```

Mutability

Lists are mutable

→ change order of items in a list

→ reassign an item in a list

Example:

```
cities = [‘Berlin’, ‘London’, ‘Paris’]
cities[0] = ‘Barcelona’
print(cities)
```

Similar data structure that is not mutable?

Iterate over a List

`for` loops

Version 1:

```
rasters = [“lc2011.asc”, “lc2012.asc”, “lc2013.asc”]
for raster in rasters:
    print(raster)
```

Version 2:

```
rasters = [“lc2011.asc”, “lc2012.asc”, “lc2013.asc”]
for i in range(len(rasters)):
    print(rasters[i])
```

- **Explain differences! Advantages/Disadvantages?**
- **Change list elements?**

List Comprehensions

- Use to create list from an *iterable*

Example:

```
numbers = (2, 3, 45, 67, 86, 94, 100) # tuple
squares = [num**2 for num in numbers]
print(squares)
```

→Shorthand for a `for` loop

→Often used to convert values in a list to different type (cast)

List Methods

Lists are objects

Object:

- Data/values
- Methods

Invocation:

```
object.method(argument1, argument2, argument3,...)
```

Most list methods are `void(in-place)`

Meaning?

Functions and Lists

Some **built-in functions** work with lists:

```
numbers = [2, 3, 45, 67, 86, 94, 100]
print(len(numbers))
print(max(numbers))
print(min(numbers))
print(sum(numbers))
```

Some work with a variety of data types

Example:

```
cities = ['Berlin', 'London', 'Paris']
print(min(cities))
```

Which of the four functions above do not work with lists of strings?

List Operations

`+` operator → concatenate lists

```
num1 = [1, 2]
num2 = [3, 4]
print(num1 + num2)
```

`*` operator → repeat lists

```
num2 = [3, 4]
print(num2 * 4)
```

`[:]` operator → slice lists

```
num3 = [3, 4, 5, 6, 7, 8, 9]
print(num3[4:])
```

`in` operator → search in list

```
num3 = [3, 4, 5, 6, 7, 8, 9]
print(6 in num3)
```

Delete Elements

`pop` → modifies list and returns element that was removed

```
cities = ['Berlin', 'London', 'Paris']
city = cities.pop(1)
print(cities)
print(city)
```

`del` → modifies list

```
cities = ['Berlin', 'London', 'Paris']
del cities[1] # try also del cities[:2]
print(cities)
```

`remove` → modifies list

```
cities = ['Berlin', 'London', 'Paris']
cities.remove('London')
print(cities)
```

→ use if you know the value of the element, but not the index

Differences?

Copying a List

In-place method, e.g., `append()` → original list is modified

```
pop_list = [5000, 6000]
print(pop_list)
new_pop = 5600
pop_list.append(new_pop)
print(pop_list)
```

Best way to keep a copy of the original list?

Shallow Copy

- Think of variables as tags attached to objects
- Two ways to attach tags:

  Shallow copy: attaches two tags (variables) to the same object (e.g., list)

  Deep copy: see next slide

```
a = [1, 2, 3, 4, 5, 6]
print(a)
b = a  #  shallow copy
print(b)
a.reverse()
print(a)
print(b)     # also reversed
```

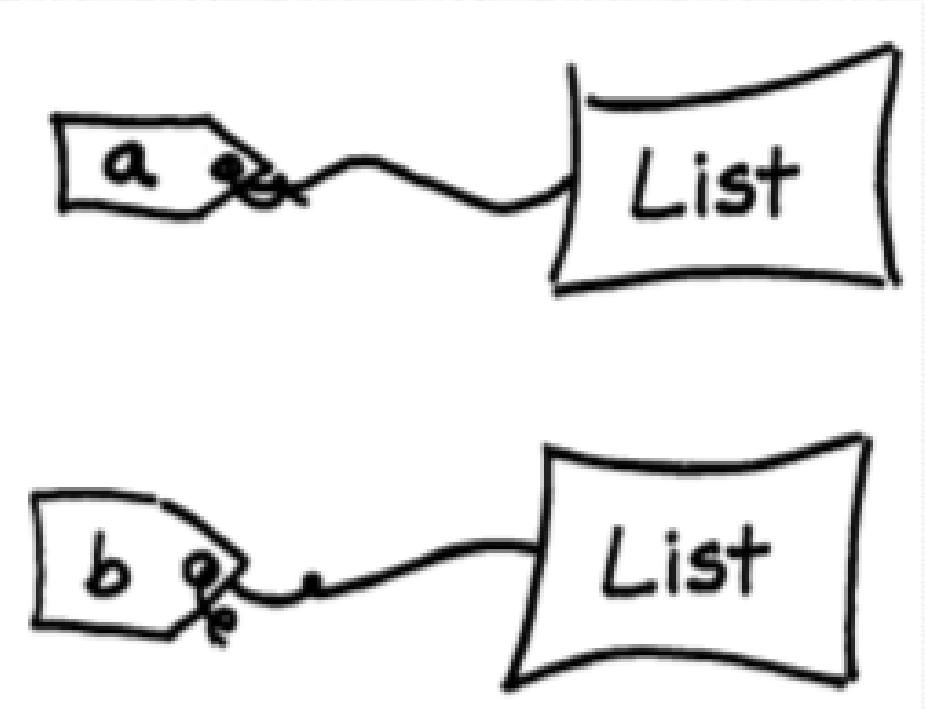
Deep Copy

- Two ways to attach tags:

  Shallow copy: see earlier slide

  Deep copy: attaches each tag (variable) to a separate object (e.g., list)

```
a = [1, 2, 3, 4, 5, 6]
print(a)
b = list(a)#  deep copy
print(b)
a.reverse()
print(a)
print(b)     # not reversed
```



COLLEGE OF ATMOSPHERIC AND GEOGRAPHIC SCIENCES
DEPARTMENT OF GEOGRAPHY
AND ENVIRONMENTAL SUSTAINABILITY
The UNIVERSITY *of* OKLAHOMA