CS118
List Comprehensions

List comprehensions are a means to express actions on lists using set notation. This is analogous to MATLAB's logical indexing.

There are three basic parts:

1. An action – what is being performed?
2. An iteration construct – Iterate through the available values
3. A predicate – Work only on certain values [optional]

Examples:

0. We don't have to have a list – we can use comprehensions to create a list:

```
L = [raw_input('A string? ') for x in range(0:3)]
```

1. Extract the even values:

```
L = [1, 2, 3, 4, 5]
LC = [x for x in L if x%2==0]
```

`LC` is now `[2, 4]`

Action: `x` (yield the original value)
Iteration: `for x in L`
Predicate: `if x%2==0`

2. Convert to strings

```
L = [1, 2, 3, 4, 5]
LC = [str(x) for x in L]
```

`LC` is now `['1', '2', '3', '4', '5']`

No predicate needed

3. Replace evens with strings – need to use a special operator (which doesn't look like an operator):

```
L = [1, 2, 3, 4, 5]
LC = [str(x) if x%2==0 else x for x in L]
```

`LC` is now `[1, '2', 3, '4', 5]`

The operator is the "ternary" operator:
"`<True value> if <condit> else <False value>`" – which means:
Action: `str(x) if x%2==0 else x`
Iteration: `for x in L`
Predicate: None

4. Checking data types

```
L = [1, '2', [3], (4)]
LC = ['str' if type(x)==type('') else 'other' for x in L]
```

`LC` contains `['other', 'str', 'other', 'other']`

5. Working with parts of a list – use slices

```
L = [1, 2, 3, 4, 5, 60, 75, 80, 95, 100]
LC = [x for x in L[5: ] if x%10==0]
```

6. Nesting operations

```
L = [1, '2', [3], (4,)]
LC = [
'str' if type(x)==type('') else
'int' if type(x)==type(1) else
'float' if type(x)==type(1.1) else
'list' if type(x) == type([]) else
'tuple' if type(x) == type(()) else 'other'
for x in L]
```