The different ways of for …

- Iterating through a list can be done in different ways
- Simple is usually better
- … but you should understand "strange" constructs, too …

Sequences: Lists and Strings

- finite sequence of elements/objects: o, o, o, …, o
- Example: Strings
  - S = "Hello World! FOO BAR"

<table>
<thead>
<tr>
<th>H</th>
<th>e</th>
<th>l</th>
<th>l</th>
<th>o</th>
<th>W</th>
<th>o</th>
<th>r</th>
<th>l</th>
<th>d</th>
<th>!</th>
<th>F</th>
<th>O</th>
<th>O</th>
<th>B</th>
<th>A</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
</tr>
</tbody>
</table>

Slicing sequences …

- if S is a sequence (string, list)…
- …we can do S[start:end]
- yielding the slice from index start to end-1!
- can also start from the back of the sequence: [-1], [-2], …

- defaults:
  - for start: 0    S[3]
  - for end: len(S)    S[:]

... and more operations on strings and lists

- String concatenation:
  - S3 = S1 + S2
  - "Hi " + "there" → "Hi there!"

- String indexing:
  - S[i]
  - "foobar"[3] → "b"

- String slicing:
  - S[m:n]

- String length:
  - len(S)

- Iterating over:
  - for x in S:

- Initialization:
  - S = ""
  - L = []

- Testing membership (new!)
  - e in X → True/False

- Others (on strings):
  - .split(), .upper(), …