





Lecture 4: Lists

Announcements



Working on the waitlist still.

Enrollment & HR systems are complex. 😕

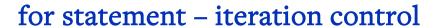


for Loops

Learning Objectives: Using Lists in Practice



- for Loops are a "generic" way to iterate over data.
- Use range in a for loop





Repeat a block of statements for a structured sequence of variable bindings





Repeat a block of statements for a structured sequence of variable bindings





Repeat a block of statements until a predicate expression is satisfied

UC Berkeley | Computer Science 88 | Michael Ball | http://cs88.org

Demo Comparing a for loop and a while loop



Learning Objectives



- Compare a for loop and a while loop.
- Learn to use range()
- Use a string as a sequence of letters

<sequence expression> — What's that?



- Sequences are a type of data that can broken down into smaller parts.
- Common sequences:
 - range() gimme all the numbers
 - strings
 - lists (next!)
- We'll start with two basic facts:
 - range(10) is the numbers 0 to 9, or range(0, 10)
 - [] means "indexing" an item in a sequence.
 - "Hello"[0] == "H"

Live Coding Demo





Lists

Learning Objectives



- Lists are a new data type in Python.
- Lists can store any kind of data and be any length.
- We start counting items of lists at o.
- Lists are mutable. We can change their data!

Lists



- A structure in Python that can hold many elements
 - Also referred to an an "array" in other programming languages.
- Lists are used to group similar items together.
 - A "contact list", a "list of courses", a "to do list"
- Python lists are really flexible!
 - Can contain any type of data
 - Can mix and match types!
 - Can add and delete items

Types We've Learned So Far



- Each type of data has a specific set of functions (methods) you can apply to them, and certain properties you can access.
- int/Integers
 - · 1, -1, 0,...
- float ("decimal numbers")
 - 1.0, 3.14159, 20.0
- string
 - "Hello, CS88"
- function
 - max(), min(), print(), your own functions!
- list
 - ['CS88', 'DATA8', 'POLSCI2', 'PHILR1B']

List Operations



- [] "square brackets": Used to access items in a list. We start at o!
- len(): The number of items in a list
- +: We can add lists together
- min(), max(): Functions that take in a list and return some info.
- Converting between types: Strings and Lists:
 - <string>.split(<separator>) → List of strings
 - 'I am taking CS88.'.split(' ')
 - <string>.join(<list>) → String, with the items of a list joined together.
 - ' '.join(['I', 'am', 'taking', 'C88C.'])
- Lots more interesting tools!
 - https://docs.python.org/3.7/tutorial/datastructures.html

 UC Berkeley | Computer Science 88 | Michael Ball | http://cs88.org

Selecting Elements From a List (A Reference, Don't Memorize Yet!)



- Selection refers to extracting elements by their index.
- Slicing refers to extracting subsequences.
- These work uniformly across sequence types.
- L = [2,0,9,10,11]
- S = "Hello, world!"
- L[2]== 9
- L[-1] == L[len(t)-1] == 11
- S[1] == "e" # Each element of a string is a one-element string.
- L[1:4] == (L[1], L[2], L[3]) == (0, 9, 10)
- S[1:2] == S[1] == "e"
- S[0:5] == "Hello", S[0:5:2] == "Hlo", S[4::-1] == "olleH"

Rules of Indexing & Slicing



- We start counting from o.
 - You will mess this up. We all do. It's ok.
 - There's lots of bad dad jokes about this.
- Python provides flexibility, but can be confusing.
 - [0] means the first item
 - [-1] means the last item, [-2] 2nd to last, and so on
- Slicing: The last value is exclusive!
 - [:stop], e.g. my_list[:5] # items 0-4
 - [start:stop], e.g. my_list[2:5] # items 2,3,4
 - [start:stop:step] e.g. my_list[0:8:2] # items
 0,2,4,6



Demo



Sequences

Learning Objects



- Lists are a type of sequence
- There are many types of sequences in Python.
 - range
 - string
 - tuples
- Sequences all share some common properties.

Sequences



- The term sequence refers generally to a data structure consisting of an indexed collection of values, which we'll generally call elements.
 - That is, there is a first, second, third value (which CS types call #0, #1, #2, etc.)
- A sequence may be finite (with a length) or infinite.
- It may be mutable (elements can change) or immutable.
- It may be indexable: its elements may be accessed via selection by their indices.
- It may be iterable: its values may be accessed sequentially from first to last.

range



- range() is a built in Python tool that generates a sequence of numbers.
 - It does not return a list unless we explicitly ask for one.
- It has many options: start, stop, and step.
- Range is lazy! It can be iterated over, but doesn't compute all its values at once.
 - We'll revisit this later.
- GOTCHA: Range is exclusive in the last value!
 - range(10) is a sequence on 10 numbers from 0 to 9.
- https://docs.python.org/3.7/library/stdtypes.html?highlight=range

Tuples



- Tuples are represented by ()
- They show up everywhere in Python, often implicitiy.
 - e.g. a,b = 1, 2 # 1,2 is really (1,2)
- Tuples are immutable.
 - t[2] = 4 is an Error.





List Comprehensions

Learning Objectives



- List comprehensions let us build lists "inline".
- List comprehensions are an expression that returns a list.
- We can easily "filter" the list using a conditional expression, i.e. if

Data-driven iteration



- describe an expression to perform on each item in a sequence
- let the data dictate the control
- In some ways, nothing more than a concise for loop.

```
[ <expr with loop var> for <loop var> in <sequence expr > ]
[ <expr with loop var> for <loop var> in <sequence expr >
if <conditional expression with loop var> ]
```