CSCI 141

Scott Wehrwein

String Comparisons and Ordering

Goals

- Understand the behavior of the following operators on strings:
  - `<`, `>`, `==`, `!=`, `in`, and `not in`
- Understand how Python orders strings using lexicographic ordering

Operators on Strings

Familiar:

`+`	concatenation	`"a" + "b" => "ab"`
`*`	repetition	`"ha" * 3 => "hahaha"`
`[ ]`	indexing, slicing	`"batman"[:3] => "bat"`
`==`	equals	`"antman" == "natman" => False`
`!=`	not equals	`"antman" != natman" => True`

String operators

Unfamiliar, but intuitive:

```
in      "a" in "abc".                 # => True
        "dab" in "abacadabra" # => True
        "A" in "abate"                # => False
        "eye" in "team"               # => False
```

`not in`: exactly what you think (opposite of `in`)

String operators

Familiar, but (a little) unintuitive:

`<, >`

much like in a dictionary

Inequality comparisons follow lexicographic ordering:

- Order based on the first character
- If tied, use the next character,
- and so on

These are all `True`:

```
"a" < "b"
"ab" < "ac"
"a" < "aa"
"" < "a"
```

String operators

Familiar, but (a little) unintuitive:

`<, >`

Caveat: lexicographic ordering is case-sensitive, and ALL upper-case characters come before ALL lower-case letters:

These are all `True`:

```
"A" < "a"
"Z" < "a"
"Jello" < "hello"
```

Lexicographic Ordering

Example: `"Bellingham" > "Bellevue"`

```
"Bellingham"
"Bellevue
```

Lexicographic Ordering

Example: `"Bellingham" > "Bellevue"`

```
"Bellingham"
"Bellevue
```

Tie - next character

Lexicographic Ordering

Example: `"Bellingham" > "Bellevue"`

```
"Bellingham"
"Bellevue
```

Tie - next character

Lexicographic Ordering

Example: `"Bellingham" > "Bellevue"`

```
"Bellingham"
"Bellevue
```

Tie - next character

Lexicographic Ordering

Example: `"Bellingham" > "Bellevue"`

```
"Bellingham"
"Bellevue
```

Tie - next character

Lexicographic Ordering

Example: `"Bellingham" > "Bellevue"`

```
"Bellingham"
"Bellevue
```

`i > e`, so `"Bellingham" > "Bellevue"`

Aside:

```
"Bell" < "Bellingham" => True
```

When all letters are tied, the shorter word comes first.

Lexicographic Ordering: Aside

"?" < "!" # => ???

The ord function takes a character and returns its numerical (ASCII) code, which determines its ordering.

The chr function takes a numerical (ASCII) code and returns the corresponding character.

```
ord("?") # => 63
ord("!") # => 33
```

```
"?" < "!" # => False
```