Lecture 8:

STRINGS

CSC111: Introduction to CS through Programming

R. Jordan Crouser

Assistant Professor of Computer Science

Smith College

Discussion

*s-u-converter.py - /Users/jcrouser/Google Drive/Teaching/Course Material/CSC111...

```
score = eval(input("Please enter score: "))
isTakingSU = input("S/U? ")

if (isTakingSU):
    # calculate S or U
    if (score >= 70):
        print("S")
    else:
        print("U")
else:
    print(score)
```

Python 3.7.0 Shell

```
Please enter score: 30
S/U? False
U
```

Ln: 14 Col: 4

what's going on?

Overview

- ✓Recap of data types
- Strings
  - operations on strings
  - accessing individual letters
  - handy methods
- The `main()` function
- Lab: Pretty Printing
- Life skill #2: debugging

(RECAP) Core concept 3: strings

- In CS, a sequence of characters that isn't a number is called a **`string`**
- In Python, a string is declared using **quotation marks**
- Strings can contain letters, numbers, spaces, and special characters
- Example:

```
x = "Jordan"
x = "Stoddard G2"
```

Operations on strings

- **Concatenation**: join two strings together with +, e.g.

```
"SCS" + " " + "Noonan"
```

- **Repetition** (i.e. self-concatenation): use *, e.g.

```
3 * "hi"
```

Multi-line strings

- **Problem**: a string that looks ugly when you try to type it all on one line, e.g.

```
desc = "This course is an introduction to co
mputer science and computer programming. The
programming language Python (Version 3) is u
sed to introduce basic programming skills an
d techniques."
```

- We can use **triple quotes** to make a multi-line string, e.g.

```
desc = """This course is an introduction to
computer science and computer programming.
The programming language Python (Version 3)
is used to introduce basic programming
skills and techniques."""
```

Escaping quotes

- **Problem:** you have a statement that contains both an apostrophe and double quotes, e.g.

```
"I can't!" he said
```

- What's the **issue** here?
  - If we try to wrap it in single quotes, Python thinks the apostophe in should end the string:

```
s = '"I can't!" he said'
```

  - If we try to wrap it in double quotes, Python thinks the double quote at the beginning of the sentence should end the string

```
s = ""I can't!" he said"
```

Escaping quotes

- **Problem:** you have a statement that contains both an apostrophe and double quotes, e.g.

```
“I can’t!” he said
```

- **Solution**: protect (“escape”) special characters using a backslash, e.g.

```
s = ‘“I can\’t!” he said’
```

or

```
s = “\“I can’t!\” he said”
```

Accessing individual letters

- One way to think about a **`string`** is as a **`list`** of letters:

```
name = "Jordan"
     ≈ ['J', 'o', 'r', 'd', 'a', 'n']
         0    1    2    3    4    5
```

- **Question**: how would I print out the 3rd letter (position 2)?

```
print(name[2])
```

“Slicing” (getting a substring)

- What about the 2nd - 5th letters (positions 1-4)?

```
print(name[1:5])
```

up to, but **not including**

- What happens if we do this?

```
print(name[2:])
```

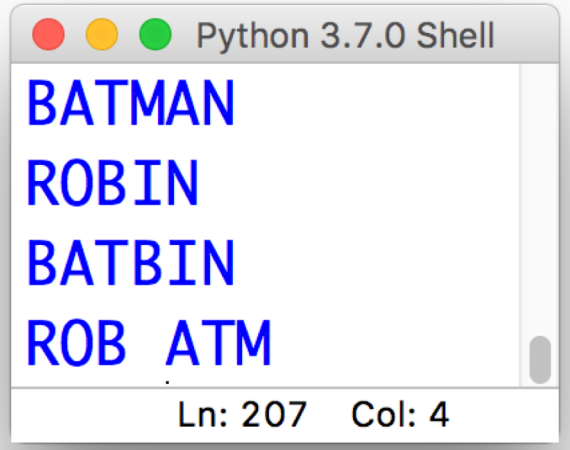
- What about this?

```
print(name[-2:])
```

15-minute exercise

- Given this string:

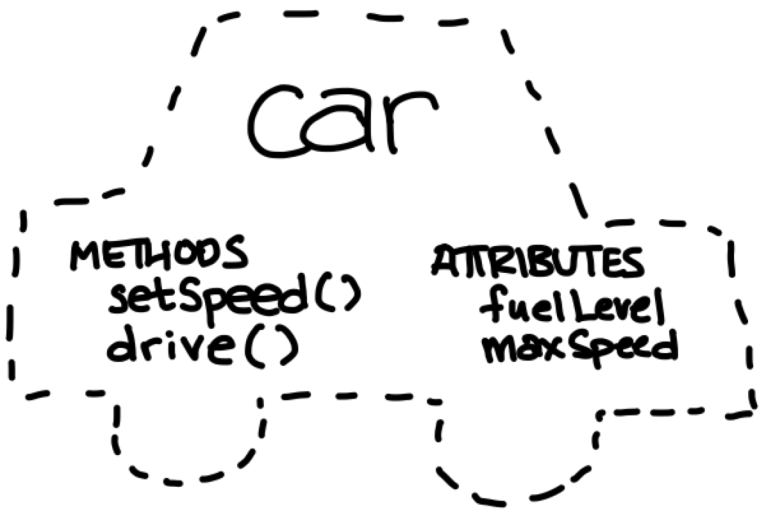
- Write a short program that uses **slicing** to produce**:**



Discussion

What did you come up with?

Strings as objects



“object-oriented”

Useful methods for working with strings

- `s.lower()`: convert the string `s` to lowercase
- `s.upper()`: convert the string `s` to UPPERCASE
- `s.strip()`: remove whitespace from the start / end of `s`
- `s.replace('old', 'new')`: replace all occurrences of `'old'` in `s` by `'new'`
- `s.split(c)`: slice `s` into pieces using `c` as a delimiter
- `s.join(list)`: opposite of `split()`, join the elements in the list together using `s` as the delimiter, e.g.

```
'-'.join(['a', 'b', 'c']) # a-b-c
```


DEMO
TIME

Fun fact

- **`strings`** in python are **immutable** (along with **`ints`**, **`floats`**, **`bools`**, and a few other built-in types)
- This means that when we call a method on them, the original isn't modified

Assignment #3: copycat

- In this assignment, you will write a python program that **manipulates a user-entered string**. For example:

```
Enter a sentence: I love computer science!
```

- You program will then output the following:

```
0. I love computer science!
1a. I LOVE COMPUTER SCIENCE!
1b. i love computer science!
2. I loovee coompuuteer sciieencee!
3. I lov...ence!
4. I Love Computer Science!
5. !ecneics retupmoc evol I
```

- Submit your **`copycat.py`** on Moodle by **11:55pm on Sunday**

Overview

- ✓Strings
  - ✓operations on strings
  - ✓accessing individual letters
  - ✓handy methods
- The `main()` function
- Lab: Pretty Printing
- Life skill #2: debugging