CSE 503
Introduction to Computer Science for Non-Majors

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Day 07
Python Exercises, Intro to JavaScript
Announcements

- Solutions for previous lecture's exercises posted to website
Recap

- Comments and assertions are ways to document your code and your reasoning/assumptions
  - Comments are ignored by Python. Allow you to explain to the reader what is going on and **why**
  - Assertions are a way to codify assumptions you've made about your code in a way that Python understands and can enforce
- 3 exercises bringing together all of our knowledge so far
Recap (Exercise #1 from Last Time...)

Assume we have a standard deck of playing cards.

Write a function named `color` that returns the color of a card based on the suit of the card.

Assume the function takes a single string, and that the string passed in corresponds to the suit of the card: "Clubs", "Diamonds", "Hearts" or "Spades"

For example, `color("Clubs")` should return "black".
Recap (Exercise #2 from Last Time...)

Write a function named `name` that takes the numerical value of a card and returns a string corresponding to the name of the card.

For example, `face(12)` would return "Queen".

If the card does not have a special name, the function should just return the number as a string.

For example, `face(9)` would return "9".  

Reminder: `str(x)` converts `x` to a string...

Write some tests with `assert` to verify your assumptions.
Recap (Exercise #3 from Last Time...)

Time to put it all together!

Define a function named `description`, which takes a numerical value, and a suit, and returns a description of the card.

For example, `description(12, "Clubs")` should return:

"The Queen of Clubs is black!"
Introduction to JavaScript

● Another programming language
  ○ Commonly used for web-based applications
  ○ Good for visualization/GUI
  ○ To make a REPL on repl.it choose Node.js template

● Just like Python, JavaScript has:
  ○ expressions
  ○ statements
  ○ variables
  ○ functions
  ○ etc...
Introduction to JavaScript

- Another programming language
  - Commonly used for web-based applications
  - Good for visualization/GUI
  - To make a REPL on repl.it choose Node.js template

- Just like Python, JavaScript has:
  - expressions
  - statements
  - variables
  - functions
  - etc...

Much of what we've learned in Python we can now apply to JavaScript...

...but with different syntax (and sometimes different semantics)
Expressions in JavaScript

Simple Expressions:
null, true, false
numeric literals (floating point)
string literals
variables

Compound Expressions:
<expression> <operator> <expression>
or
<operator> <expression>
or
<expression> <operator>
Expressions in JavaScript

Simple Expressions:
- null, true, false
- numeric literals (floating point)
- string literals
- variables

Compound Expressions:
- `<expression> <operator> <expression>`
- `<operator> <expression>`
- `<expression> <operator>`

binary expressions

unary expressions
Operators in JavaScript

Some examples of binary operators:

- arithmetic: +, -, *, /, %, **
- string: +
- relational: <, <=, >, >=, ==, !=
- boolean (w/short circuiting): &&, ||
Operators in JavaScript

Some examples of binary operators:

- arithmetic: +, -, *, /, %, **
- string: + (equal to and not equal to (both same as in Python)
- relational: <, <=, >, >=, ==, !=
- boolean (w/short circuiting): &&, ||
Operators in JavaScript

Some examples of binary operators:

**arithmetic**: +, -, *, /, %, **

**string**: +

**relational**: <, <=, >, >=, ==, !=

**boolean (w/short circuiting)**: &&, ||

(equal to and not equal to)

(both same as in Python)

(both same semantics as Python, but different syntax)
Operators in JavaScript

Some examples of unary operators:

arithmetic: +, -

boolean: !
Operators in JavaScript

Some examples of unary operators:

- arithmetic: +, -
- boolean: ! (not)
Variables in JavaScript

- Variables in JavaScript **must** be *declared* before use
  - Similar to how we must assign a value to a variable in Python before use
  - Variable declaration is a statement

- Statements in JavaScript **must** end with `;`

```javascript
let x;

x = 13;

let y = 12;
```
Variables in JavaScript

- Variables in JavaScript must be declared before use
  - Similar to how we must assign a value to a variable in Python before use
  - Variable declaration is a statement
- Statements in JavaScript must end with ;

```javascript
let x;
x = 13;
let y = 12;
```
Variables in JavaScript

- Variables in JavaScript **must** be *declared* before use
  - Similar to how we must assign a value to a variable in Python before use
  - Variable declaration is a statement
- **Statements in JavaScript must end with ;**

  ```javascript
  let x;
  x = 13;
  let y = 12;
  ```

  *Technically...this isn't a strict requirement, but it's safer to follow this rule*
Function Definitions in JavaScript

- Two parts: **header** and **body** (...sound familiar?)

```javascript
function area(w, h) {
    return w * h;
}
```
Function Definitions in JavaScript

- Two parts: **header** and **body** (...sound familiar?)

```javascript
function area(w, h) {
    return w * h;
}
```
Function Definitions in JavaScript

● Two parts: header and body (...sound familiar?)

```javascript
function area(w, h) {
  return w * h;
}
```
Function Calls in JavaScript

- they look the same

```javascript
let x = area(10, 12);
```

Function call
Comments in JavaScript

// This is a single line comment

/* This comment is one that spans multiple lines... */
console.log("some cool string");
<table>
<thead>
<tr>
<th>Python</th>
<th>JavaScript</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>Boolean</td>
</tr>
<tr>
<td>str</td>
<td>String</td>
</tr>
<tr>
<td>int</td>
<td>Number</td>
</tr>
<tr>
<td>float</td>
<td></td>
</tr>
</tbody>
</table>
# Type Names in JavaScript vs Python

<table>
<thead>
<tr>
<th>Python</th>
<th>JavaScript</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>Boolean</td>
</tr>
<tr>
<td>str</td>
<td>String</td>
</tr>
<tr>
<td>int</td>
<td>Number</td>
</tr>
<tr>
<td>float</td>
<td></td>
</tr>
</tbody>
</table>

...And many many more...