SY110
Programs – Statements and Variables

Major Brian Hawkins, USMC

U.S. Naval Academy

Fall AY 2018

Outline

Background

Executable Program vs. Programming-language program

- Executable program – a binary, machine-readable file that contains instructions and runs an application (`ssh`, notepad, Word, etc)
- Programming-language program – a text file that is written in a “high-level” programming language (C/C++, Java, Python, etc)

Programming

Why write programs?

- Because executable programs consist of millions or billions of bytes – imagine typing 0's and 1's for days...
  - Typos? How easy to check for correctness?

So when we program (verb)...

- We write text files in a high level programming language
- Our text is then either:
  - Converted directly from the programming language to machine instructions (binary) by a compiler. C, C++ work this way
  - Interpreted into a lower-level language by an interpreter, and the lower-level instructions are then converted to binary instructions. Also called scripting languages. Examples: JavaScript, Python, Ruby

JavaScript

What is JavaScript?

- It's a high-level programming language!
- It's most commonly interpreted!
- It's a scripting language for web browsers!

Yes. . .

There will be some programming in this class, and you might even like it! We'll use it for math, and later on for web programming.

What we'll cover in JavaScript

Writing programs is different than using a calculator – on a calculator you evaluate one expression at a time, by pressing "=". When programming, most often an entire file full of statements and expressions is run at once. Interpreted languages can be used in an interactive way, however – like a calculator. That's where we'll start.

What we'll cover today

1. Numbers, Expressions, and Variables
2. Strings
3. Types

JS Numbers, Expressions, Variables

Open a web-based JavaScript interpreter here:
`http://rona.cs.usna.edu/~sy110/resources/interpreterBare.html`

Numbers

- The simplest JavaScript expression is just a number
- Try typing 42 at the `js>` prompt
- You'll see 42 printed underneath
- What just happened?

JS Numbers, Expressions, Variables

Basic Arithmetic

- *, /, +, - operators all available
  - Same order of operations you know and love
  - ^ not available, but 5^3 =5*5*5
- Try typing `10 + 4 * 8` at the `js>` prompt
  - Did you get what you expected?
- ( ) can be used to specify a certain order *e.g.* `(10 + 4) * 8`

Modulo

- Modulo operator (%)
  - x%y returns the remainder of x/y
  - *e.g.* 5%3 returns 2

JS Numbers, Expressions, Variables

Built-in math functions

It would get old (or be impossible) to write many math functions using only basic operators, so JavaScript provides some for us.

- Math.sin(x), Math.cos(x) – evaluates sin/cos of x (in rad)
- Math.sqrt(x) – returns the square root of x
- And many more: `http://www.w3schools.com/jsref/jsref_obj_math.asp`

Exercise

- Calculate hypotenuse of a right triangle with base lengths 68 and 23.5
  - Recall: $a^2 + b^2 = c^2 \Rightarrow c = \sqrt{a^2 + b^2}$
- If restaurant bill is $73.50, calculate an 18% tip rounded up to the nearest whole dollar

JS Numbers, Expressions, Variables

Expressions vs. Statements

Everything we have done thus far has been an *expression* – something that returns a value, *e.g.* 4 + 38 returns 42. Not everything we write in JavaScript returns a value, however; these are called *statements*

Variables

- Variables are like containers we store values in.
- The statement `var x;` declares the variable `x` – the word "var" is mandatory!
- We can then assign a value to `x` by typing `x = 42;`, another statement.
- Note that "=" means *assignment*...
- ...so we can write things like `x = x + 1;`
- What is the value of `x` now?

JavaScript Strings

So far every value in JavaScript we've seen has been a number *type*.
But there are many more types of values in JavaScript .

Strings

Strings are just a sequence of characters, enclosed in single- or double-quotes.

- `"SY110 is fun"` is a string, so is `‘Go Navy!’`
- Type `alert("Hello, World!")` at the `js>` prompt
- Indexing strings – `"GONAVY"[0]`, `"GONAVY"[3]`
- String concatenation – what does the expression `‘go’ + "navy"` return?
- We can assign strings to variables as well
  - *e.g.* `var foo = "This is a string"`

What if we need quotation marks inside of strings?

JavaScript Strings

String functions

JavaScript gives us many functions built-in to operate on strings. For example:

- To get the length of a string, try `‘gonavy’.length`
- Is there a string with length 0?
- Substrings – make a smaller string from part of another, use `substr(x,y)`
  - *e.g.* `‘very long string’.substr(2,4)`
- To find the ASCII value of a character, we use `charCodeAt()`
  - *e.g.* `"gonavy".charCodeAt(0)`
- To return the ASCII character for a given value use `fromCharCode()`
  - *e.g.* `String.fromCharCode(97)`

JavaScript String Example Problems

What if we want to find the letter 10 letters after the letter ‘g’?

What about if we wanted to do this with the letter ‘w’? Can we wrap back around to ‘a’ after ‘z’? How would we do this?

JavaScript String Example Problems

What if we want to find the letter 10 letters after the letter ‘g’?

What about:

```
String.fromCharCode(("g".charCodeAt(0) + 10));
```

Why does it work?

What about if we wanted to do this with the letter ‘w’? Can we wrap back around to ‘a’ after ‘z’? How would we do this?

What about:

```
String.fromCharCode(("w".charCodeAt(0) - 97 + 10) % 26 +
97);
```

Why does it work?

JavaScript Data Types

We've learned about two JavaScript types that values can be – *numbers* and *strings*.

- You can use + with a number and a string
  - The number is *implicitly* converted by the JavaScript interpreter to a string first, then concatenated
  - `alert("There are " + 24*60*60 + " seconds in a day!");`
- But that doesn't always give us what we want:
  - `alert("There are " + 3 + 18 + 2+ " coastal states!");`
  - How might we fix this?

JavaScript Types

How to tell what type?

JavaScript offers a built in function called `typeof()` that tells us what the type of something is. Try:

- `typeof(7+5);`
- `typeof("7" + "5");`
- `typeof(7 * "5");`
- `typeof(7 + "5");`

United States
Naval Academy

Questions

Questions?