CS333 Project3 **#README**

Tracy Quan

Platform: macOS version 10.12.6

Directory Layout:

Project3/

|

| |\_\_/cstk.h

| |\_\_/cstk.c

| |\_\_/cstktest.c

| |\_\_/cstktest

| |\_\_/README.docx

|

|\_\_/JavaScript/

| |

| |\_\_/Task1.html

| |\_\_/Task1.js

| |\_\_/Task2.html

| |\_\_/Task2.js

| |\_\_/Task3.html

| |\_\_/Task3.js

| |\_\_/Extension4.html

|

|

|\_\_/C++/

| |

| |\_\_/Task1.cpp

| |\_\_/Task1.out

| |\_\_/Task2.cpp

| |\_\_/Task2.out

| |\_\_/Task3.cpp

| |\_\_/Task3.out

|

|\_\_/Bash/

| |

| |\_\_/Extension1.sh

| |\_\_/Extension2.sh

| |\_\_/Extension3.sh

**Part I: C Syntax**

The purpose of this part is to understand the usage of .h and .c files, the keyword extern, and the concept of prototype in C.

**Task1:**

Create a cstk.h. The .h file should contain the necessary sturctures and the declarations of the member functions.

**Task2:**

Create a cstk.c file. This file should contain the implementation of the member functions declared in the cstk.h file.

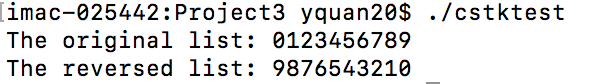
**Task3:**

**Command line:**

gcc -o cstktest cstktest.c cstk.c

./cstktest

**Output:**

****

**Part II: Selected Language**

**1. JavaScript**

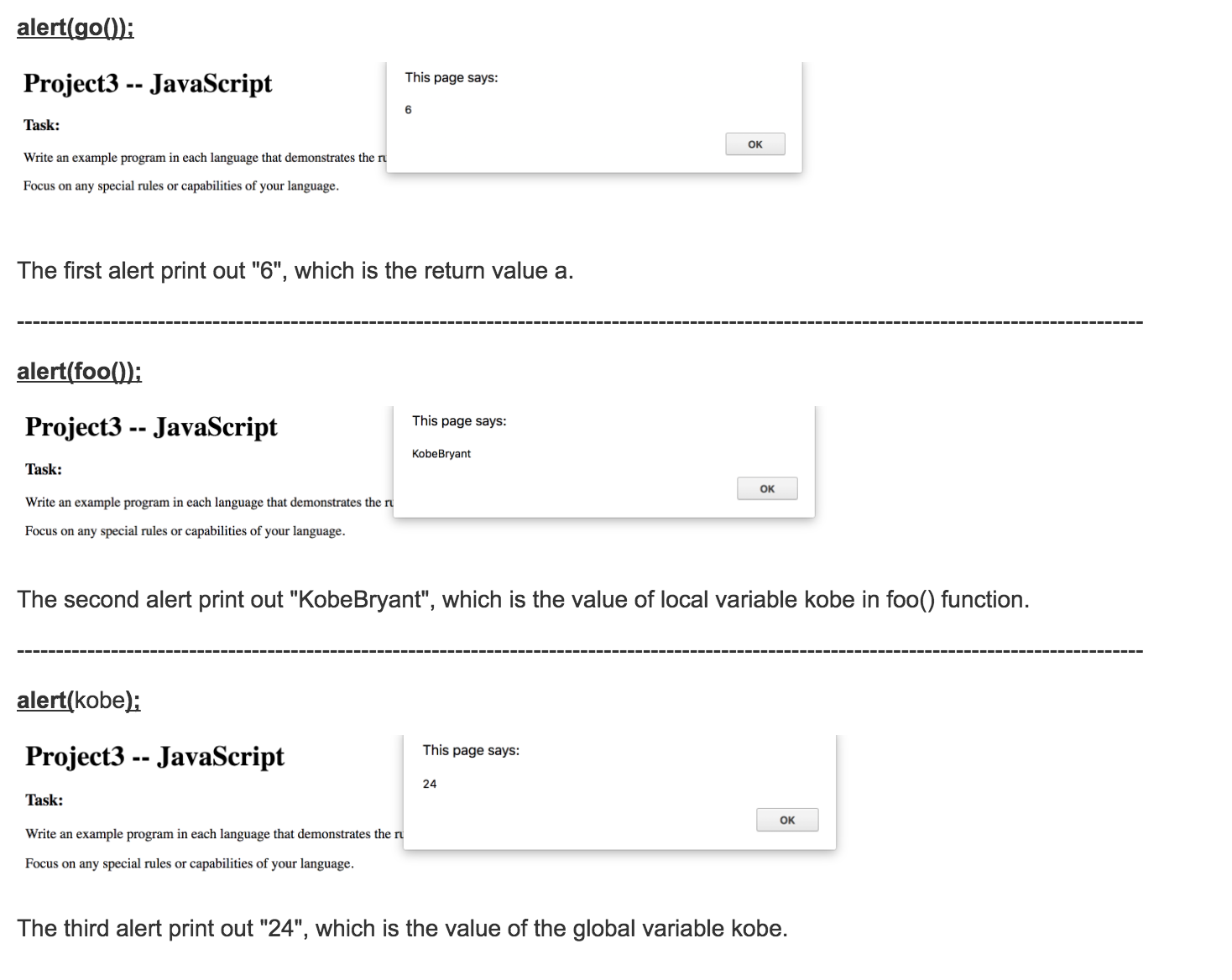
**Task1:**

Write an example program in each language that demonstrates the rules for identifier naming, variable declarations and identifier scoping.

**Instruction:**

Open the Task1.html and the outputs are in each alert window.

**Output:**



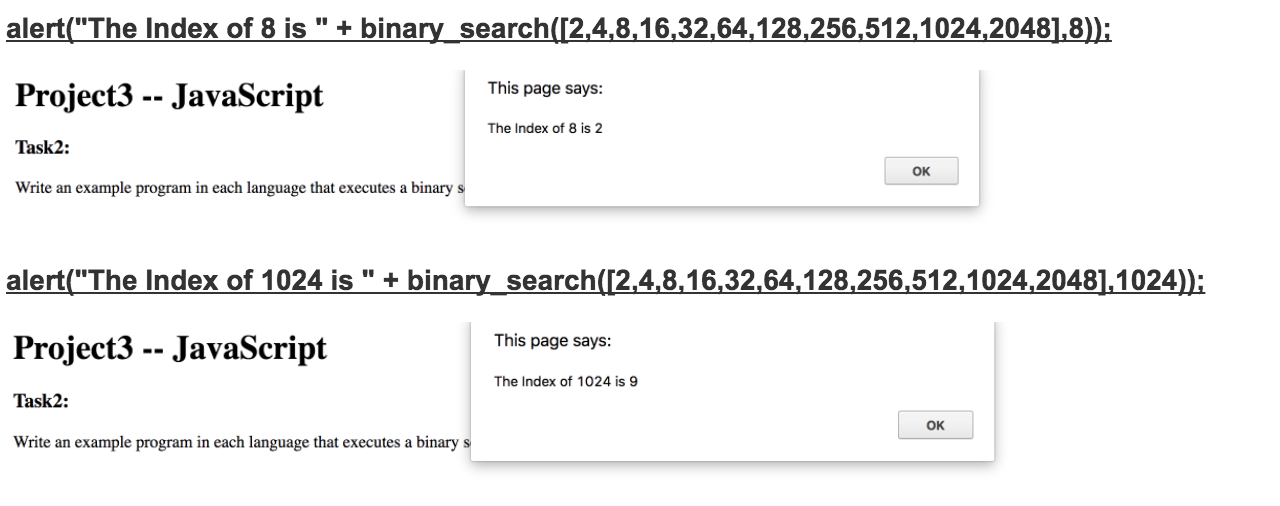
**Task2:**

Write an example program in each language that executes a binary search on a list or array of numbers. Make it a function/method, if possible.

**Instruction:**

Open the Task2.html and the outputs are in each alert window.

**Output:**



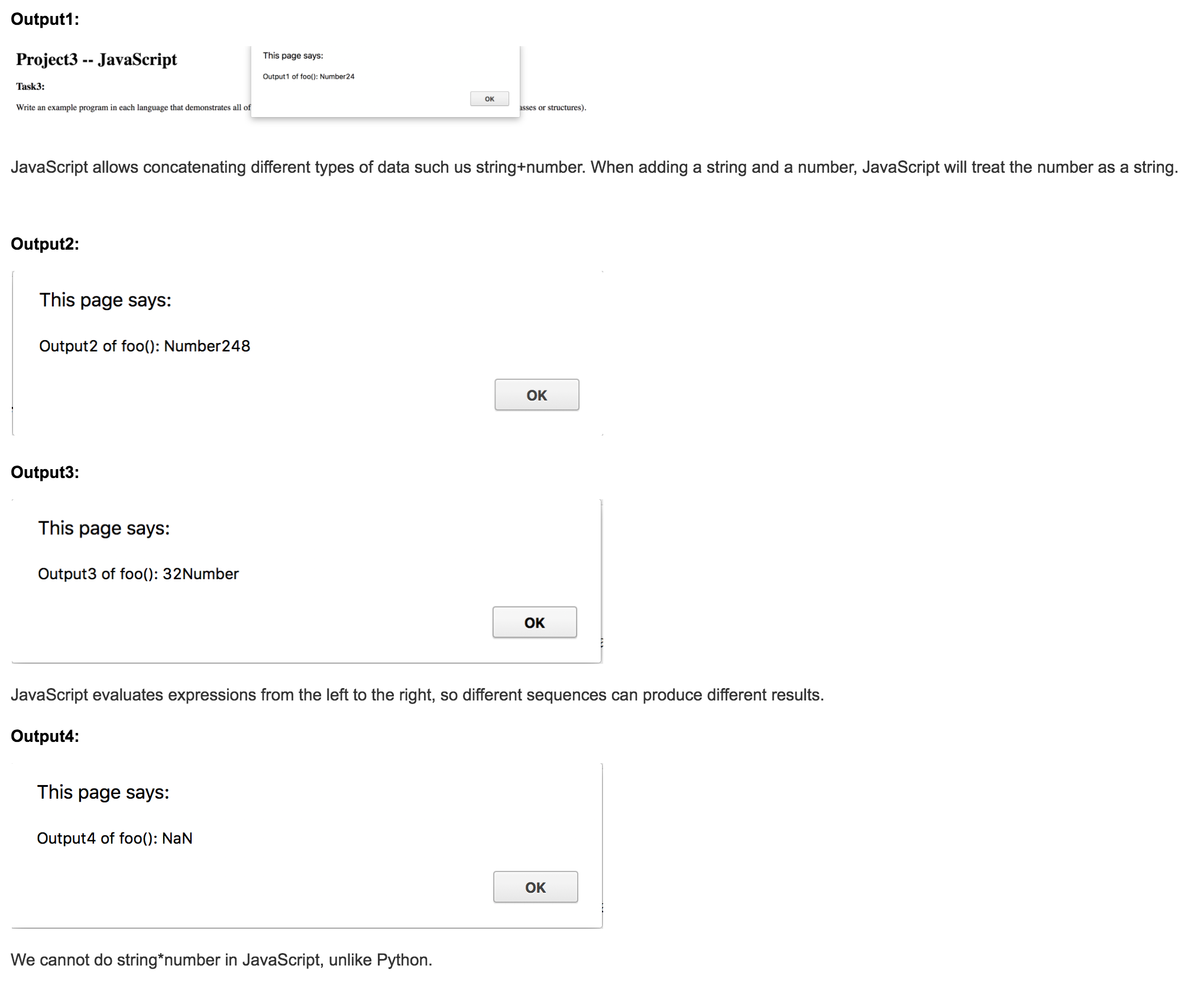
**Task3:**

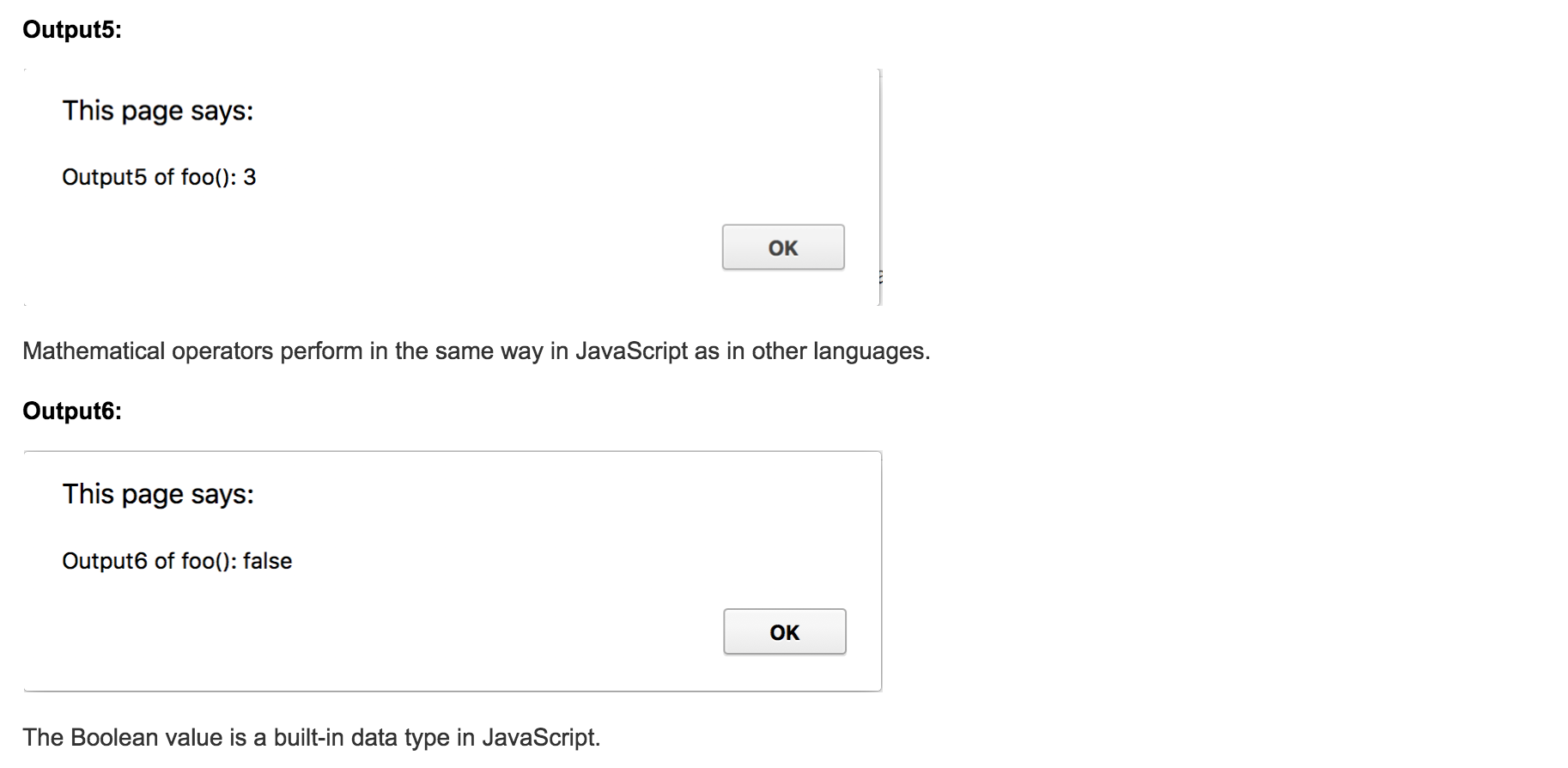
Write an example program in each language that demonstrates all of the basic built-in types and how to construct aggregate types (e.g., records, classes or structures).

**Instruction:**

Open the Task3.html and the outputs are in each alert window.

**Output:**







**2. C++**

**Task1:**

Write an example program in each language that demonstrates the rules for identifier naming, variable declarations and identifier scoping.

**Command Line:**

g++ -o Task1.out Task1.cpp

./Task1.out

**Output:**



**Task2:**

Write an example program in each language that executes a binary search on a list or array of numbers. Make it a function/method, if possible.

**Command Line:**

g++ -o Task2.out Task2.cpp

./Task2.out

**Output:**



**Task3:**

Write an example program in each language that demonstrates all of the basic built-in types and how to construct aggregate types (e.g., records, classes or structures).

**Command Line:**

g++ -o Task3.out Task3.cpp

./Task3.out

**Output:**



**Part III: Extensions**

**Extension1**

Variable Declarations, Identifiers Naming, Identifier Scope in Bash

**Command Line:**

./Extension1.sh

**Output:**



**Extension2**

Show unique features of the syntax, naming, or scoping in Bash.

**Command Line:**

./Extension2.sh

**Output:**



**The Uniqueness of Bash:**

* When we refer to the value of a variable, we have to use the dollar sign($). See line 14 and 15.
* When we assign a value to a reference, we cannot leave any whitespace. See line 5 and 6.
* The value of a variable cannot have whitespace, or we have to use quotation marks to put them in a string. See line 8 and 9.
* Other rules are shown in the following extension.

**Extension3**

Binary Search in Bash.

**Command Line:**

./Extension3.sh

**Output:**



**Extension3**

Assign a function to a variable in JavaScript.

**Instruction:**

Open Extension4.html in a web browser. Click the button in-line function.

**Output:**

