Name________________________

Intro to Programming Final

December 15th 2011

For this test you can assume there is a printLine() function. The printLine function takes a variable number of arguments separates them with spaces and displays them on the page followed by a newline character (Just like the one from the homework).

There are a total of 100 possible points on the test.

I recommend getting the ones you know first and coming back for the rest.

#1 [15 points]
Given this html write the code to add the values of #num1 and #num2 and place the result in #result. The solution should place 2.8 in #result.

```
<input type=”text” id=”num1” value=”1.5” />
<input type=”text” id=”num2” value=”1.3” />
<input type=”text” id=”result” value=”” />
```

#2 [3 points]
What is local storage? How do you use it?

Name________________________

#3 [3 points]
Can you describe the DRY principle? Why is it important?

#4 [3 points]
What is the syntax to create an empty object? What about an empty array?

#5 [4 points]
Create an object using the object literal syntax that holds the following information about you: first name, last name, major, and favorite color.

#6 [3 points]
What does the following code print:

```
var sprinkles = { name: “sprinkles”, type=”cat” };
var harry = sprinkles;
harry.name = “harry”;
printLine(harry.name);
printLine(sprinkles.name);
```

#7 [3 points]
What does the following code print:

```
var sprinkles = { name: “sprinkles”, type=”cat” };
var name = sprinkles.name;
name = “harry”;
printLine(name);
printLine(sprinkles.name);
```

#8 [3 points]
What is in the **in** operator used for? Give an example.

#9 [10 points]

Name________________________

Write a function that takes an object and prints its property names and associated values. Don’t worry about nested objects. For example:

calling printObject	produces....

`var baby = {`
`name: “Elliot”,`
`weight: 9.5,`
`dad: “Jon”,`
`mom: “Julie”`
`};`
`printObject(baby);`
	`name: Elliot`
	`weight: 9.5`
	`dad: Jon`
	`mom: Julie`

#10 [12 points]
Create a **Car** constructor function that has the following attributes:
- A private variable **mileage** that starts at 0.
- A private variable **lastRotation** that starts at 0 and is set to the current mileage when the car’s rotateTires method is called.
- a public method **drive** that increments mileage by the miles given
- a public method **needsRotation** that returns true if the *lastRotation* was done more than 10,000 miles ago.

- a public method **rotateTires** that sets the *lastRotation* to the current mileage.

#11 [10 points]
Use the Car constructor function you just wrote to:
- create a new car
- drive it for 3,000 miles
- check if it needs a tire rotation
- if so rotate the tires
- drive for 9,000 miles
- check if it needs a tire rotation
- if so rotate the tires

#12 [4 points]
What does the following print?

```
var s = "bar";
function a(){
    function b(){
        printLine(s);
    }
    function c(){
        s = "lol";
    }
    c();
    b();
}
a();
printLine(s);
```

#13 [10 points]
Write a reverseArray function that takes in an array and returns a reversed version of it. You **cannot** use the native array.reverse(). For example:

```
reverseArray([1,2,3,4,”foo”, 5])    →   [5,”foo”,4,3,2,1]
```

Name________________________

#14 [3 points]
The following code creates a multidimensional array called **board** that simulates a tic-tac-toe board:

```
var board = [];
var rows = 3, cols = 3;
for(var i = 0; i < rows; i++){
    var row = [];
    for(var j = 0; j < cols; j++){
       row[j] = Math.random() > .5 ? 'X':'O';
    }
    board[i] = row;
}
```

Write some code that will print the mark in the middle of the board:

#15 [10 points]
Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".

#16 [4 points]
What is recursion?

Bonus [10 points]
(Definition From wikipedia)
In mathematics, the factorial of a non-negative integer n, denoted by n!, is the product of all positive integers less than or equal to n. For example,
5! = 5 x 4 x 3 x 2 x 1 = 120
The value of 0! is 1, according to the convention for an empty product.

Write a factorial function using **recursion** that takes a Number, *num*, and returns the factorial of *num*.

factorial(5) should return 120. factorial(4) should return 24. factorial(0) should return 1.

Name________________________

Bonus 2 [5 points]
Describe what you liked about the course, what you did not like about the course, and how you would improve it.