C M S I 3 8 6

Homework #4

Due: 2013-11-12

Submit hardcopy solutions for each of the problems. Where code is required, make sure the code is embedded within your document, but *also* provide links to the code that you have saved on ideone or codepad.

1. Give an abstract syntax tree for the following C expression:

```
(a = 3) >= m >= ! & 4 * ~ 6 || y %= 7 ^ 6 & p
```

2. JavaScript's implicit semicolon insertion is often considered to be poorly designed because the following four cases aren't exactly intuitive:

```
function f() {
    return
       {x: 5}
}
```

```
var b = 8
var a = b + b
(4 + 5).toString(16)
```

```
var place = "mundo"
["Hola", "Ciao"].forEach(function (command) {
  alert(command + ", " + place)
})
```

```
var sayHello = function () {
    alert("Hello")
}

(function() {
    alert("Goodbye")
}())
```

   What is being illustrated in each of the above? Go, Python, Scala, and Ruby all allow line endings to end statements and you don't hear people complaining about them the way they do about JavaScript. Pick one of these four languages and show why they handle the four "problematic" cases of JavaScript.

3. Give an example of a program in C that would not work correctly if **local variables** were allocated in static storage as opposed to the stack. For the purposes of this question, local variables do not include parameters.

4. Consider the following pseudocode:

```
var x = 100;
function setX(n) {x = n;}
```

```
function printX() {console.log(x);}
function first() {setX(1); printX();}
function second() {var x; setX(2); printX();}
setX(0);
first();
printX();
second();
printX();
```

What does this program print if the language uses static scoping? What does it print with dynamic scoping? Why?

5. The expression `a-f(b)-c*d` can produce different values depending on how a compiler decides to order, or even parallelize operations. Give a small program in the language of your choice (or even one of your own design) that would produce different values for this expression for different evaluation orders. Please note that by "different evaluation orders" we do *not* mean that the compiler can violate the precedence and associativity rules of the language.

6. Explain the meaning of the following C declarations:

```
double *a[n];
double (*b)[n];
double (*c[n])();
double (*d())[n];
```

7. Rewrite the four declarations in the previous problem in Go. Yes, I did say Go.

8. Translate the following expression into (a) postfix and (b) prefix notation, in both cases *without using parentheses*:

```
(-b + sqrt(4 × a × c)) / (2 × a)
```

Do you need a special symbol for unary negation? Why or why not?

9. Write the interleave function from the previous three assignments in C++, using C-style arrays.

10. Write the interleave function from the previous three assignments in C++, using C++ **vectors** (from the Standard Library).