CSCI 330: Homework 2

*Due Sunday, November 3rd @ 11:59PM*

*Submit as .pdf to autolab*

 *54 points (3 pts each)*

*\*\* .doc version of this assignment is on the website \*\**

1. What is a reserved word? How is it different from a key word?
2. How can a variable be characterized? (What are the six attributes we discussed?)
3. After language design and implementation, what are three times bindings can take place in a program?
4. What is the lifetime of a variable? How is the lifetime of a static local variable different than a non-static local variable in C++?
5. Describe a situation where a history-sensitive variable would be useful in a subprogram. Could this variable be a local variable in C++? Why or why not?
6. Assume the following JavaScript program was interpreted using static-scoping rules. What value of x is displayed in function sub1? Under dynamic-scoping rules, what value of x is displayed in function sub1?

**var** x;

**function** sub1() { document.write("x = " + x + "");

}

**function** sub2() {

**var** x; x = 10;

sub1();

}

x = 5;

sub2();

1. Consider the following C program and for each of the four marked points in this function, list each visible variable, along with the number of the definition statement that defines it.

void fun(void) {

int a, b, c; /\* definition 1 \*/

. . .

while (. . .) {

int b, c, d; /\*definition 2 \*/

. . . <------------- 1

while (. . .) {

int c, d, e; /\* definition 3 \*/

. . . <------------- 2

}

. . . <-------------- 3

}

. . . <---------------- 4

}

1. Consider the following C++ functions. What is the lifetime of the local variable sum? What is the scope of the local variable sum? Are they the same?

void fun1() {

 . . .

} /\* end of fun1 \*/

void fun2() {

 int sum;

 . . .

 fun1();

} /\* end of fun2 \*/

1. Describe and differentiate four different memory allocation schemes for arrays. Give code examples for those that exist in C++.
2. What are the arguments for and against implicit heap storage recovery (garbage collection) vs the explicit techniques required in C++? Is Java’s garbage collection lazy or eager?
3. Define what it means for a language to be strongly typed. Is C++ strongly typed? Is OCaml strongly typed? Why or why not?
4. Differentiate name type equivalence and structure type equivalence.
5. Give an example of coercion. How is this different than casting?
6. If a language includes pointers and explicit heap storage recovery, what are the two primary dangers introduced?
7. What is the difference between operator precedence and operator associativity?
8. Why do we use parenthesis in expressions?
9. What makes Boolean expressions confusing in C/C++?
10. Consider the code below. What are the values of sum1 and sum2 if the operands in the expressions are evaluated left to right? What are the values of sum1 and sum2 if the operands in the expressions are evaluated right to left?

int fun(int\* k) {

 \*k += 4;

 return 3 \* (\*k) – 1;

}

void main() {

 int i = 10, j = 10, sum1, sum2;

 sum1 = (i / 2) + fun(&i);

 sum2 = fun(&j) + (j / 2);