CSCI 330: Homework #4

(51 pts, 3 pts each – graded on completion rather than correctness)

1. What are the two general categories of a selection statement?
2. If you were designing a programming language and only wanted to include one type of looping statement would you choose *while* or *for* loops? Why?
3. What are the pros and cons of using unique closing reserved words on compound statements?
4. What are the pros and cons of allowing both single and compound statements as the body of selection statements (such as if-else statements)?
5. Describe a programming situation in which the else clause in Python’s **for** statement would be convenient.
6. Consider the following C program segment. Rewrite it using no gotos or **break**s.

j = -3;

**for** (i = 0; i < 3; i++) {

**switch** (j + 2) {

**case** 3:

**case** 2: j--; **break**; **case** 0: j += 2; **break**; **default**: j = 0;

}

**if** (j > 0) **break**;

j = 3 - i

}

1. What are some advantages and disadvantages of *keyword* parameters? Of *positional* parameters?
2. What are the semantic modes, the advantages, and the disadvantages of pass-by-value, pass-by-result, pass-by-value-result, and pass-by-reference parameter-passing methods?
3. Consider the following program written in C syntax:

void fun(int first, int second) {

first += first;

second += second;

}

void main() {

int list[2] = {1,3};

value = 1;

fun(value, list[value]);

}

For each of the following parameter-passing methods, what are the values of the list array after execution?

* 1. Passed by value
  2. Passed by reference
  3. Passed by value-result

\*\*Note that for passed by value-result, the addresses are moved in first, then the values. On the return, the values are copied to the addresses that were moved in/saved. if you are unsure of the order in which other steps should be taken, refer to section 9.5.8 (Examples of Parameter Passing) in the book.

1. Consider the following program, written in JavaScript syntax:  
   function suba(subp) {  
    var y;

y = 5;  
 subp();  
};

function sub1() {  
 var y;

y = 2;  
 function sub2() {  
 alert(y); // dialog box with value of x  
 };  
 function sub3() {  
 var y;  
 y = 3;  
 suba(sub2);  
 };  
   
 y = 2;  
 sub3();  
 };

In this code, a subprogram name is passed as a parameter. There are three choices for defining the referencing environment for this subprogram parameter: shallow binding, deep binding and ad hoc binding. Under each of these choices, what will be displayed in the dialog box?

* 1. Deep binding:
  2. Shallow binding:
  3. Ad hoc binding:

1. Show the stack with all activation record instances, including static and dynamic chains, when execution reaches position 1 in the following skeletal program. Assume bigsub is at level 1.

function bigsub() { function a() {

function b() {

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

} // end of b function c() {

...

b();

...

} // end of c

...

c();

...

} // end of a

...

a();

...

} // end of bigsub

1. Define: *exception, exception handler, raising an exception, continuation,* and *finalization*.
2. What are the advantages of having support for exception handling built in to a language? Can exception handling be implemented without having built in support?
3. What is special about the “finally” block with exception handling?
4. Differentiate black box and white box testing? Should one be chosen over the other?
5. Differentiate integration testing vs system testing.
6. When should test planning begin?