Homework 4
Due March 21, 2016

Submissions are due by the beginning of class on the specified due date. Handwritten or typed solutions are acceptable. If you do write your solutions by hand, be sure to write clearly. If the grader cannot read your answer, they cannot give you the points. Late submissions will be accepted with a 10% penalty for each day (24 hours) they are late.

1. (10 points) Indicate the binding time for each of the following decisions in C/C++.
   - The number of built-in types.
   - The variable declaration that corresponds to a particular variable reference.
   - The maximum length allowed for a constant character string.
   - The address of a particular library routine.
   - The total amount of space occupied by program code and data.

2. (10 points) In Fortran 77, local variables are typically allocated statically. In C++, they are typically allocated in the stack. In some other languages such as lisp, they are at least partially in the heap. What accounts for these differences? Give an example C++ routine that would not work correctly if local variables are allocated statically.

3. (10 points) For the following C++ program, indicate where the variables a, b, c, d, f, and g are allocated (stack, heap or static data?).

```cpp
int a, b;
void foo() {
    int c;
    int *d = new int;
    ...
}

main() {
    int f;
    int *g = new int;
    ...
}
```
4. (35 points) Consider the following pseudocode. What does the program print if the language uses static scoping? What does it print with dynamic scoping?

```plaintext
x:integer // global
procedure halve_x
    x := x/2    // integer division
procedure print_x
    write_integer(x)
procedure first
    x := 10
    print_x()
procedure second
    x:integer
    x := 9
    halve_x()
    third()
    print_x()
procedure third
    x:integer
    x := 7
    halve_x()
    print_x()

first()
print_x()
second()
print_x()
```

5. (35 points) Consider the following pseudocode which uses dynamic scoping. What does the program print if the language uses shallow binding? What does it print with deep binding?

```plaintext
x:integer // global
procedure print_x
    write_integer(x)
procedure first
    x := x * 2
procedure second(F: procedure)
    x:integer
    x := 5
    F()
    print_x()

x := 7
second(first)
print_x()
```