Homework Assignment #9

Note

This assignment is due 2:10PM Wednesday, December 26, 2012. Please write or type your answers on A4 (or similar size) paper. Drop your homework by the due time in Yih-Kuen Tsay's mail box on the first floor of Management College Building 2. Late submission will be penalized by 20% for each working day overdue. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

1. (10 points) Draw a control flow diagram for the following program fragment:

```
repeat
    $S_1$;
    if $E$ then
        $done$ := $true$
    else $S_2$
until $done$;
```

2. (20 points) We have examined in class the following program fragment in Pascal for removing adjacent duplicates:

```
read($x$);
while $x \neq 0$ do begin
    writeln($x$);
    repeat
        read($next$)
    until $next \neq x$;
    $x$ := $next$;
end;
```

   Please rewrite the program in C using the **for** statement as the only looping construct:

   for (⟨initialize⟩; ⟨test⟩; ⟨step⟩) ⟨statement⟩

3. (20 points) Write a program in C, C++, or Java to implement the task described by the following pseudocode:

```
loop
    copy characters up to "(*";
    throw away characters until "*)" is seen;
end;
```

   Please try to make good use of the **break** and **continue** statements. You must not use any user-defined procedures/functions. Remember to handle the end of file.

4. (30 points) It is possible to test if a C compiler computes the address of an array element according to the row-major or column-major layout (although this may be clear from the way a multi-dimensional array is declared in C). Please design a program to accomplish the test. (Hint: utilize "pointer arithmetic".)

5. (20 points) As we have discussed in class, it is easy to test if the compiler of an imperative language uses static scoping or dynamic scoping to bind a local variable (not defined in the current procedure/function) to a declaration (outside of the procedure/function). We may adapt the test for the interpreter/compiler of a functional language.

   Please devise an OCaml program to determine which (static or dynamic) scoping rule the OCaml interpreter adopts.