ML PATTERN MATCHING

TYPE INFERENCE EXERCISES

For each of the ML functions below . . .

i. draw a lambda-calculus style abstract-syntax tree. (In the text book these are called “abbreviated parse tree” (Mitchell, p. 137) or “Parse Graphs” (Mitchell, pp. 157ff) because it uses an edge instead of an extra constraint to equate the types of bound-variable occurrences.)

ii. label each node with that node’s literal type (if it is trivially known) or a type variable.

iii. list the set of a type constraints derivable from the variable bindings and the operations (e.g., abstraction, application, tuple construction) in the abstract tree.

iv. solve the constraints using unification to determine the type of each variable, and

v. determine the type of the function.

1. fun g(x) = 10.0 - x;
   (You can ignore the ad-hoc polymorphism and use the curried type ‘real -> real -> real’ as the type of ‘-’.)

2. fun g(f,x,y) = f(x) div y;

3. fun g(f,x,y) = (2+f(3*x),x,y);
   (You can use the curried type ‘int -> int -> int’ as the type of ‘+’ and ‘*’.)

4. fun g(x,xs) = x::xs;
   (The curried type of :: is 'a -> 'a list -> 'a list where 'a parameterizes the polymorphic 'a list’ type.)

VARIABLE SCOPE

5. A definition of global variables is given on p. 163 of Mitchell.

   (a) How does this differ from what is conventionally considered global variables in a C or C++? (cf. http://www.cplusplus.com/doc/tutorial/variables.html)

   (b) Give an example of a variable declaration in a C program that would be ‘global’ according to Mitchell’s definition but would not normally be called ‘global’ by a C/C++ programmer.

   (c) What λ-calculus concept most closely matches Mitchell’s notion of ‘global’ variables?

(over)
6. Consider the program:

```c
int x = 22;

int f(int y) {
    return x + y;
}

main() {
    int x = 33;
    {
        int x = 44;
        printf("%d", f(7));
    }
}
```

(a) To which declaration would the `x` on line 4 refer to under static scoping rules.
(b) To which declaration would the `x` on line 4 refer to under dynamic scoping rules.
(c) What is the program's output under static scoping rules?
(d) What is the program's output under dynamic scoping rules?

7. (a) List one language feature common to both C and C++ that is dynamically scoped in both languages.
(b) Would static scoping have made sense for this feature? Why or Why not?
(c) List one language feature common to both C++ and Java that is dynamically scoped in both languages.
(d) Would static scoping have made sense for this feature? Why or Why not?

8. (a) What is the difference between the scope of a variable and its lifetime?
(b) Give an example program fragment (in C, ML, Java or Scala) containing a variable and a statement for which the variable is not in scope at the statement, but the variable is live when the statement executes.

Note: If you collaborated with your classmates or used their notes, please note which classmates you collaborated with. If you use an external source, besides the text book, lectures, notes provided by the instructor, and your own intellect, please cite that source. Use quote marks if you are quoting material word-for-word from any source (including the text book).