

CS 405/505 ASSIGNMENT #4

Due Monday, March 31, 2008

1. Design and implement a symbol table “class” in Prolog. The symbol table elements will be of the form `symbolTableEntry (Name, Category, Type, Value)` and the symbol table itself will be a list of such values. The following predicates are required:
 - (a) `add (CurrentSymbolTable, Name, Category, Type, Value, NewSymbolTable)` - creates a new symbol table with the name, category, type, and value fields added; if `Name` is already in the symbol table, fails
 - (b) `entry (SymbolTable, Name, Category, Type, Value)` - returns the category, type and value associated with `Name`; if `Name` is not in the symbol table, fails
 - (c) `category (SymbolTable, Name, Category)` - returns the category associated with `Name`; if `Name` is not in the symbol table, fails
 - (d) `type (SymbolTable, Name, Type)` - returns the type associated with `Name`; if `Name` is not in the symbol table, fails
 - (e) `value (SymbolTable, Name, Value)` - returns the value associated with `Name`; if `Name` is not in the symbol table, fails

Test your class using the following query.

```
add([], h, var, int, 0, Test2SymbolTable1),
add([], x, var, int, 0, AreaSymbolTable1),
add(AreaSymbolTable1, y, var, int, 1, AreaSymbolTable2),
add(AreaSymbolTable2, z, var, int, 2, AreaSymbolTable),
add(Test2SymbolTable1, area, method, int,
    [formalParameters([x, y]), localSymbolTable(AreaSymbolTable),
     syntaxTree(:=(z, *(2, +(+(*(x, y), *(x, h)), *(y, h))))), return(z)]),
    Test2SymbolTable2),
add([], a, var, int, 0, MainSymbolTable1),
add(MainSymbolTable1, b, var, int, 1, MainSymbolTable2),
add(MainSymbolTable2, s, var, int, 2, MainSymbolTable),
add(Test2SymbolTable2, main, method, void,
    [formalParameters([], localSymbolTable(MainSymbolTable),
     syntaxTree(:(:(:(:=(a, 3), =(b, 4)), =(h, 5)),
     =(s, apply(area, @(a, b))))), print(s)]),
    Test2SymbolTable),
add([], test2, class, test2, localSymbolTable(Test2SymbolTable), SymbolTable),
entry(Test2SymbolTable, h, Categoryh, Typeh, Valueh),
entry(Test2SymbolTable, area, Categoryarea, Typearea, Valuearea),
entry(MainSymbolTable, b, Categoryb, Typeb, Valueb),
\+(entry(SymbolTable, main, Categorymain, Typemain, Valuemain)),
category(AreaSymbolTable, y, Categoryareay),
\+(category(MainSymbolTable, y, Categorymainy)),
type(SymbolTable, test2, Typetest2),
\+(type(SymbolTable, class, Typeclass)),
value(MainSymbolTable, s, Values),
\+(value(AreaSymbolTable, area, ValueareaError)).
```

2. Consider the Core program below:

```
i := 1;
while (i <= n) loop
  i := i + 1;
end loop;
```

Using the axiomatic semantics of Core, show that the assertion

$$\{n \geq 0\} S \{i = n + 1\}$$

is correct.

Hint: Use the loop invariant $i \leq n + 1$. The consequence rules may be used to force the other rules into a form consistent with this loop invariant. (You should analyze the program to understand why this loop invariant was chosen.)