

## ASSIGNMENT #6

Due Tuesday, March 9, 2004

1. Write a Prolog predicate `replace (Atom1, Atom2, List1, List2)` that takes two atoms, `Atom1` and `Atom2`, and a list `List1` as input parameters and replaces all occurrences of `Atom1` in `List1` with `Atom2`, no matter how deeply the first atom is nested, returning the result in `List2`. Test this predicate using the following calls:

```
replace(nothing, something, [], List).
replace(a, 1, [a, b, r, a, c, a, d, a, b, r, a], List).
replace(b, 0, [no, bs, here], List).
replace(nest, here, [nest, [second, nest, level], [third, [nest], level],
  [[[big, nest]]]], List).
```

2. Write a Prolog predicate `structtypeeq (Type1, Type2)` that returns true if `Type1` and `Type2` are structurally equal according to the following rules:
  - Basic types (e.g. `int`, `float`, etc.) are structurally equal if they are identical.
  - Arrays are denoted by `array (Lowerbound, Upperbound, Elementtype)` and two arrays are structurally equal if they have the same element type and number of elements.
  - Record structures are denoted by `record ([field (Var1, Type1), field (Var2, Type2), ..., field (Varn, Typen)])` and two record structures are structurally equal if their elements have the same types.
  - Pointers are denoted by `pointer (Type)` and two pointer types are structurally equal if they point to structurally equal types.

Test this predicate using the following calls:

```
structtypeeq(array(1, 10, int), array(0, 9, int)).
structtypeeq(record([field(a, int)], field(b, float)]),
  record([field(x, int), field(y, float)])).
structtypeeq(record([field(a, int), field(b, float)]),
  record([field(a, int), field(b, float), field(c, int)])).
structtypeeq(record([field(n, int), field(trees, array(1, 100, pointer(tree)))]),
  record([field(n, int), field(trees, array(1, 100, pointer(tree)))]).
```

3. Write a Prolog predicate `delete (Atom, List1, List2)` that deletes all occurrences, no matter how deep, of `Atom` in `List1` and returns the result in `List2`. The returned list cannot contain anything in place of the deleted atoms. Test this predicate using the following calls:

```
delete(nothing, [], List).
delete(a, [a, b, r, a, c, a, d, a, b, r, a], List).
delete(b, [no, bs, here], List).
delete(nest, [nest, [second, nest, level], [third, [nest], level],
  [[[big, nest]]]], List).
```

4. Consider the Core program segment below:

```
i := n;  
while (i > 0) loop  
  i := i - 1;  
end loop;
```

Using the axiomatic semantics of Core, show that the assertion

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

is correct.

Hint: Use the loop invariant  $i \leq n$ . 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.)