| | | |
|---|---|---|
| **CS 153** | **Foundations of Software** | **Homework 6** |
| **Fall 2006** | | **Due: Nov 13, 2006** |

1. [5] Exercise 6.3.3(b) (p. 382)

2. [15] Exercise 6.3.5(b,d) (p. 382)

3. [15] Exercise 6.3.6(d,f) (p. 382)

4. [10] Exercise 6.4.2 (p. 392)

5. [20] Exercise 6.4.3(d,h) (p. 392)

6. [35] Write (and test and debug) a program in Standard ML (SML) that implements Quine's method for testing whether a formula is a tautology. You should build on the wff.sml code provided as the solution of Problem 8, Homework 5.

You can use the following function `subst(p,f1,f2)` that substitutes the wff `f1` for each occurrence of the propositional variable `pv` throughout the wff `f2`. This will be used to substitute the boolean constants `True` and `False` for `pv` as required in Quine's method.

```
(* subst : pv * wff * wff -> wff *)
fun subst(pv,f,g as Var(pv')) = if pv = pv' then f else g
  | subst(pv,f,Or(g1,g2)) = Or(subst(pv,f,g1),subst(pv,f,g2))
  | subst(pv,f,And(g1,g2)) = And(subst(pv,f,g1),subst(pv,f,g2))
  | subst(pv,f,Imp(g1,g2)) = Imp(subst(pv,f,g1),subst(pv,f,g2))
  | subst(pv,f,g) = g
```

You can also use the following `simplify` function that simplifies away any boolean constants in its argument wff, producing the resulting simplified wff.

```
(* simplify : wff -> wff *)
fun simp(Not(True)) = False
  | simp(Not(False)) = True
  | simp(Or(True,f)) = True
  | simp(Or(False,f)) = f
  | simp(Or(f,True)) = True
  | simp(Or(f,False)) = f
  | simp(And(True,f)) = f
  | simp(And(False,f)) = False
  | simp(And(f,True)) = f
  | simp(And(f,False)) = False
  | simp(Imp(True,f)) = f
  | simp(Imp(False,f)) = True
  | simp(Imp(f,True)) = True
  | simp(Imp(f,False)) = Not(f)
  | simp f = f

fun simplify(Not f) = simp(Not(simplify f))
  | simplify(Or(f,g)) = simp(Or(simplify f, simplify g))
  | simplify(And(f,g)) = simp(And(simplify f, simplify g))
  | simplify(Imp(f,g)) = simp(Imp(simplify f, simplify g))
  | simplify f = f
```

This code is available in the simp.sml file linked from the class web page.