COMP 317: Semantics of Programming Languages

Problem Sheet 2 Solutions

1.
   1. [[`'x := 25`]] is a function that takes a state as input and gives a state as output; given input *S*, [[`'x := 25`]](*S*) is a state; i.e., a function that takes a variable as input and gives an integer as output; `'x` is the variable given as input, so the result should be the value of `'x` in the updated state.
   2. After a few steps, the result should be 25.
      ([[`'x := 25`]](*S*))(`'x`) = *S*[`'x` \ [[`25`]](S)](`'x`) = [[`25`]](S) = [[`25`]]
      = 10 * [[`2`]] + [[`5`]] = 10*2 + 5 = 25.

2. [[`'t := 'x ; 'x := 'y ; 'y := 't`]](*S*) = [[`'x := 'y ; 'y := 't`]]([[`'t := 'x`]](*S*))
   = [[`'y := 't`]]([[`'x := 'y`]]([[`'t := 'x`]](*S*)))
   = [[`'y := 't`]]([[`'x := 'y`]](*S*[`'t` \ [[`'x`]](*S*)]))
   = [[`'y := 't`]]([[`'x := 'y`]](*S*[`'t` \ *S*(`'x`)]))
   = [[`'y := 't`]](*S*[`'t` \ *S*(`'x`)][`'x` \ [[`'y`]](*S*[`'t` \ *S*(`'x`)]))
   = [[`'y := 't`]](*S*[`'t` \ *S*(`'x`)][`'x` \ *S*[`'t` \ *S*(`'x`)](`'y`)])
   = [[`'y := 't`]](*S*[`'t` \ *S*(`'x`)][`'x` \ *S*(`'y`)])
   = *S*[`'t` \ *S*(`'x`)][`'x` \ *S*(`'y`)][`'y` \ [[`'t`]](*S*[`'t` \ *S*(`'x`)][`'x` \ *S*(`'y`)])]
   = *S*[`'t` \ *S*(`'x`)][`'x` \ *S*(`'y`)][`'y` \ *S*[`'t` \ *S*(`'x`)][`'x` \ *S*(`'y`)](`'t`)]
   = *S*[`'t` \ *S*(`'x`)][`'x` \ *S*(`'y`)][`'y` \ *S*[`'t` \ *S*(`'x`)](`'t`)]
   = *S*[`'t` \ *S*(`'x`)][`'x` \ *S*(`'y`)][`'y` \ *S*(`'x`)]

   Now prove that `'x := 'y ; 'y := 'x` *doesn't* swap the values of `'x` and `'y`.
   [[`'x := 'y ; 'y := 'x`]](*S*) = [[`'y := 'x`]]([[`'x := 'y`]](*S*))
   = [[`'y := 'x`]](*S*[`'x` \ [[`'y`]](*S*)])
   = [[`'y := 'x`]](*S*[`'x` \ *S*(`'y`)])
   = *S*[`'x` \ *S*(`'y`)][`'y` \ [[`'x`]](*S*[`'x` \ *S*(`'y`)])]
   = *S*[`'x` \ *S*(`'y`)][`'y` \ *S*[`'x` \ *S*(`'y`)](`'x`)]
   = *S*[`'x` \ *S*(`'y`)][`'y` \ *S*(`'y`)]
   So this program won't swap the values in any state where the value of `'x` is different to the value of `'y`.

3. Write a program that sets `'z` to either the value of `'a` or the value of `'b`, whichever is the greater.

   ```
   if 'a < 'b then 'z := 'b else 'z := 'a fi
   ```

   Test your program by simplifying

   - [[`'a := -2; 'b := 4 ; if 'a < 'b then 'z := 'b else 'z := 'a fi`]](*S*)
     = [[`if 'a < 'b then 'z := 'b else 'z := 'a fi`]]([[`'a := -2; 'b := 4`]]](*S*))
     = [[`if 'a < 'b then 'z := 'b else 'z := 'a fi`]](*S*[`'a` \ -2][`'b` \ 4])
     Now [[`'a < 'b`]](*S*[`'a` \ -2][`'b` \ 4]) = -2 < 4 = true, so, carrying on from above:

= [[`'z := 'b`]](S[`'a` \ -2][`'b` \ 4])
= S[`'a` \ -2][`'b` \ 4][`'z` \ S[`'a` \ -2][`'b` \ 4](`'b`)]
= S[`'a` \ -2][`'b` \ 4][`'z` \ 4]

- [[`'a := 9; 'b := 4 ; if 'a < 'b then 'z := 'b else 'z := 'a fi`]](S)
= S[`'a` \ 9][`'b` \ 4][`'z` \ 9].

4. Write a program (using a while-loop) that sets `'f` to 100!.

```
'x := 1;
'f := 1;
while 'x < 100
do
  'x := 'x + 1;
  'f := 'f * 'x
od
```

Now simplify (highlights only):
[[`'x := 1 ; 'f := 1 ; while 'x < 3 do 'x := 'x + 1 ; 'f := 'f * 'x od`]](S)
= [[`while 'x < 3 do 'x := 'x + 1 ; 'f := 'f * 'x od`]](S[`'x` \ 1][`'f` \ 1])
= [[`while 'x < 3 do 'x := 'x + 1 ; 'f := 'f * 'x od`]](S[`'x` \ 1][`'f` \ 1][`'x` \ 2][`'f` \ 2])
= [[`while 'x < 3 do 'x := 'x + 1 ; 'f := 'f * 'x od`]](S[`'x` \ 1][`'f` \ 1][`'x` \ 2][`'f` \ 2][`'x` \ 3][`'f` \ 6])
= S[`'x` \ 1][`'f` \ 1][`'x` \ 2][`'f` \ 2][`'x` \ 3][`'f` \ 6].

Challenge: we'll see later on in this module how to prove your program is correct, but right now can you explain why it's correct?
`'f` is always equal to the factorial of the value of `'x`.

5. [[`'x := 'x + 'y ; 'y := 'x - 'y ; 'x := 'x - 'y`]](S)
= [[`'x := 'x - 'y`]]([[`'y := 'x - 'y`]]([[`'x := 'x + 'y`]](S)))
= [[`'x := 'x - 'y`]]([[`'y := 'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)]))
= [[`'x := 'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ [[`'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)])])
= [[`'x := 'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ [[`'x`]](S[`'x` \ S(`'x`) + S(`'y`)]) - [[`'y`]](S[`'x` \ S(`'x`) + S(`'y`)])])
= [[`'x := 'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S[`'x` \ S(`'x`) + S(`'y`)](`'x`) - S[`'x` \ S(`'x`) + S(`'y`)](`'y`)])
= [[`'x := 'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`) + S(`'y`) - S(`'y`)])
= [[`'x := 'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)])
= S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)][`'x` \ [[`'x - 'y`]](S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)])]
= S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)][`'x` \ S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)](`'x`) - S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)](`'y`)]
= S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)][`'x` \ S(`'x`) + S(`'y`) - S(`'x`)]
= S[`'x` \ S(`'x`) + S(`'y`)][`'y` \ S(`'x`)][`'x` \ S(`'y`)]
= S[`'y` \ S(`'x`)][`'x` \ S(`'y`)].
Phew!
Bonus challenge: why is the last step valid?

Grant Malcolm

Last modified: Tue Feb 1 15:25:40 GMT 2011