

CS 405 FINAL EXAM SAMPLE QUESTIONS (SOLUTIONS)

1. A(n) loop invariant is a condition which is true upon entry to a loop every time the loop is entered, and is still true at the termination of the loop.

2. Script started on Wed 10 Dec 2003 08:57:15 PM CST

```
redstone% sbprolog
SB-Prolog Version 3.1
| ?- consult('hanoi.pro').
yes
| ?- trace(hanoi/1).
yes
| ?- trace(move/4).
yes
| ?- hanoi(3).
    (1) Call: hanoi(3) ? y
no
| ?- hanoi(3).
    (2) Call: hanoi(3) ?
    (3) Call: move(3,left,center,right) ?
    (4) Call: move(2,left,right,center) ?
    (5) Call: move(1,left,center,right) ?
    (6) Call: move(0,left,right,center) ?
    (6) Exit: move(0,left,right,center)
[left,center] (7) Call: move(0,right,center,left) ?
    (7) Exit: move(0,right,center,left)
    (5) Exit: move(1,left,center,right)
[left,right] (8) Call: move(1,center,right,left) ?
    (9) Call: move(0,center,left,right) ?
    (9) Exit: move(0,center,left,right)
[center,right] (10) Call: move(0,left,right,center) ?
    (10) Exit: move(0,left,right,center)
    (8) Exit: move(1,center,right,left)
    (4) Exit: move(2,left,right,center)
[left,center] (11) Call: move(2,right,center,left) ?
    (12) Call: move(1,right,left,center) ?
    (13) Call: move(0,right,center,left) ?
    (13) Exit: move(0,right,center,left)
[right,left] (14) Call: move(0,center,left,right) ?
    (14) Exit: move(0,center,left,right)
    (12) Exit: move(1,right,left,center)
[right,center] (15) Call: move(1,left,center,right) ?
    (16) Call: move(0,left,right,center) ?
    (16) Exit: move(0,left,right,center)
[left,center] (17) Call: move(0,right,center,left) ?
    (17) Exit: move(0,right,center,left)
    (15) Exit: move(1,left,center,right)
    (11) Exit: move(2,right,center,left)
```

```

(3) Exit: move(3,left,center,right)
(2) Exit: hanoi(3)
yes
| ?- ^D
Halt. Program terminated normally
redstone% exit
redstone%
script done on Wed 10 Dec 2003 08:59:28 PM CST

```

3. $addr(A[1, 1, 2]) = addr(A) + (((1 - (-10)) \times 11 + (1 - (-5))) \times 10 + (2 - 1)) \times 10 = 12810$

4. Worked in class but will post a solution soon.

5. $\{n = j * (j + 1)/2 \ \& \ i \geq 0\}$
while $j \neq i$ loop $j := j + 1; n := n + j$; end loop
 $\{n = j * (j + 1)/2 \ \& \ i \geq 0 \ \& \ \neg(j = i)\}$

Let $P = n = j * (j + 1)/2 \ \& \ i \geq 0$, $S = S_1 ; S_2$, $S_1 = j := j + 1$, and $S_2 = n := n + j$.

The first step is to apply the loop rule.

$$\frac{\{P \ \& \ j \neq i\}S\{P\}}{\{P\}\text{while } j \neq i \text{ loop } S \text{ end loop}\{P \ \& \ \neg(j = i)\}} \quad (1)$$

Next we apply the composition rule.

$$\frac{\{P \ \& \ j \neq i\}S_1\{Q\}, \{Q\}S_2\{P\}}{\{P \ \& \ j \neq i\}S_1; S_2\{P\}} \quad (2)$$

To compute the weakest precondition Q , we apply the assignment rule.

$$Q \equiv P[n + j/n] \equiv n + j = j * (j + 1)/2 \ \& \ i \geq 0 \quad (3)$$

We then apply the assignment rule again to verify the correctness of S_1 .

$$R \equiv Q[j+1/j] \equiv n+(j+1) = (j + 1) * ((j + 1) + 1)/2 \ \& \ i \geq 0 \equiv n = j * (j + 1)/2 \ \& \ i \geq 0 \quad (4)$$

Since R is weaker than the precondition we needed to prove, the program segment is correct.

6. • static scoping and deep binding
- 1) ARS = MAIN; DS = MAIN
 - 2) ARS = MAIN, B; DS = MAIN, B
 - 3) ARS = MAIN, B, A; DS = MAIN, A
 - 4) ARS = MAIN, B1, B2, C; DS = MAIN, B1, C
- dynamic scoping - Y belongs to B2

7. The following table shows the results for each calling method.

Call by	A[0] i	A[0] o	A[1] i	A[1] o	A[2] i	A[2] o	I i	I o	X i	X o	Y i	Y o	Res
value	7	7	11	11	13	13	0	2	7	7	0	1	8
reference	7	7	11	11	13	13	0	1	a1	a1	a2	a2	8
value-result	7	7	11	11	13	13	0	1	7	7	0	1	8
name	7	7	11	11	13	13	0	1	t1	t1	t2	t2	12

a1 - addr (A [0])

a2 - addr (I)

t1 - thunk (A [I])

t2 - thunk (I)

Aliasing does occur in F as I may be referred to by its own name or by formal parameter Y.