

# Midterm Exam II Key

## Problem 1

(30pts.) Define scope and lifetime of a variable. Explain differences and give examples.

Scope of a variable determines when the variable is visible and thus can be legally used. Typically, a variable is visible in the block in which it is defined and in all of the enclosed blocks unless it is hidden by a declaration of another variable with the same name.

Lifetime of a variable determines the span of time when actual storage is allocated and reserved for a variable.

The two notions are quite different. For example, memory is still allocated for a variable temporarily hidden from view by another variable declared in a sub-block.

## Problem # 2:

(70 pts.) [SIMPLESEM] Consider the following program:

- a) ( 15 pts.) Augment its code with comments describing the distance  $df(f)$  of all function calls and the distance  $dv(s)$  of all variables.
- b) ( 35 pts.) Write a complete SIMPLESEM code for the above program using the C4 language paradigm. Enter your *code* and *comments* in the table provided below. Use pencil so that you can easily make necessary corrections. Use the function  $fp()$  wherever needed (instead of using  $D[ ]$ ).
- c) ( 20 pts.) Describe the state of the data segment (and all of its *intermediate* states) for the above program right after the THIRD call to  $f()$  and following the assignment "int s=3". Enter *numbers* and *comments* in the table provided below . Use pencil so that you can easily make necessary corrections.

```

main()
{
  int s=1, q=2;
  int f()
  {
    int s=3;
                                <--- End of data segment following
                                <--- the third call to f()

    int g()
    {
      label_two:
      if ( s >= q )                In g: dv(s)=1, dv(q)=2
      {                             df(f)=2
        s = s - 1;
        q = q + 1;
        goto label_two;
      }
      else return f();
    }

    label_one:
    if ( s < q )
    {
                                In f: dv(s)=0, dv(q)=1
                                df(g)=0

        q = q - 2;
        goto label_one;
    }
    else return g();
  }
  s = f()/3 + 2*s;    In main: dv(s)=0, dv(q)=0, df(f)=0
}

```

Notes:

- a) Instead of fp(0) they can use D[0].
- b) There can optionally also be lines: set 2, OS; set 3, OS; set 4, OS.

## CODE SEGMENT

0	set 0, 2	Beginning of AR(main)
1	set 1, 8	End of AR(main)
2	set 5, 1	main: s
3	set 6, 2	main: q
4	set D[1], 10	RP
5	set D[1]+1, D[0]	DL
6	set D[1]+2, fp(0)	SL
7	set 0, D[1]	CP
8	set 1, D[1] + 5	FP  AR(f) =5
9	jump 12	jump to code for f
10	set 5, D[D[1]-1]/3 + 2 * D[2]	s = 2*s + f()/3
11	halt	STOP
12	set fp(0)+ 3, 3	Code for f(): s=3
13	jumpt 16, D[fp(0)+3] >= D[fp(1)+4]	Jump if s >=q
14	set fp(1) + 4, D[fp(1)+4]-2	q = q-2
15	jump 13	Goto
16	set D[1], 22	RP
17	set D[1]+1, D[0]	DL
18	set D[1]+2, fp(0)	SL
19	set 0, D[1]	CP
20	set 1, D[1] + 4	FP  AR(g) =4
21	jump 26	jump to code for g
22	set D[0]-1, D[D[1]-1]	return from g: RV
23	set 1, D[0]	FP
24	set 0, D[D[0]+1]	CP
25	jump D[D[1]]	Return to caller
26	jumpt 30, D[fp(1)+3] < D[fp(2)+4]	Jump if s < q
27	set fp(1) + 3, D[fp(1)+3]-1	s = s-1
28	set fp(2) + 4, D[fp(2)+4]+1	q = q+1
29	jump 26	Goto
30	set D[1], 36	RP
31	set D[1]+1, D[0]	DL
32	set D[1]+2, fp(2)	SL
33	set 0, D[1]	CP
34	set 1, D[1] + 5	FP  AR(f) =5
35	jump 12	jump to code for f
36	set D[0]-1, D[D[1]-1]	return from f: RV
37	set 1, D[0]	FP
38	set 0, D[D[0]+1]	CP
39	jump D[D[1]]	Return to caller

## DATA SEGMENT AFTER THE THIRD CALL TO f():

Nr.	Contents	Comments
0)	2/8/13/17/22/26	CP
1)	8/13/17/22/26/31	FP
2)		RP
3)		DP
4)		SP
5)	1	s
6)	2/3/4	q
7)	(no value)	RV(f)
8)	10	AR(f) begins #1
9)	2	DP
10)	2	SP
11)	3/2	s
12)	(no value)	RV(g)
13)	22	AR(g) begins
14)	8	<i>DP</i>
15)	8	SP
16)	(no value)	RV(f)
17)	36	AR(f) begins #2
18)	13	DP
19)	2	<i>SP</i>
20)	3/2	s
21)	(no value)	RV(g)
22)	22	AR(g) begins
23)	17	DP
24)	17	<i>SP</i>
25)	(no value)	RV(f)
26)	36	AR(f) begins #3
27)	22	DP
28)	2	SP
29)	3	s
30)		
31)		
32)		
33)		