Computer Science Foundation Exam

August 14, 2015

Section I A

COMPUTER SCIENCE

NO books, notes, or calculators may be used,
and you must work entirely on your own.

SOLUTION

Question #	Max Pts	Category	Passing	Score
1	10	DSN	7	
2	10	ANL	7	
3	10	ALG	7	
4	10	ALG	7	
5	10	ALG	7	
TOTAL	50			

You must do all 5 problems in this section of the exam.

Problems will be graded based on the completeness of the solution steps and not graded based on the answer alone. Credit cannot be given unless all work is shown and is readable. Be complete, yet concise, and above all be neat.

1) (10 pts) DSN (Recursive Functions)

Consider the problem of transforming a positive integer X into a positive integer Y when the only two operations you are allowed are adding 1 to the current number or multiplying the current number by 2. Write a recursive function that returns the minimum number of steps necessary to transform X to Y. If X > Y, return 1000000000, to indicate that no solution exists. (For example, if X = 13 and Y = 28, the correct response would be 2 - first you would add 1 to 13 to obtain 14, then multiply 14 by 2 to obtain 28.) Feel free to call the provided function. *Note: don't worry about the run time of your function - assume that the inputs are such that the run time is relatively small, even when written using straight-forward recursion. There is a clever, efficient solution without recursion but please write the slower recursive solution since the goal of this question is to test recursive thinking.*

```
#define NO_SOLUTION 1000000000

int min(int x, int y) {
    if (x < y) return x;
    return y;
}

// Returns the minimum number of steps to transform x into y, or
// 100000000 to indicate no solution.
int minSteps(int x, int y) {

    if (x > y) return NO_SOLUTION;      // 2 pts
    if (x == y) return 0;               // 2 pts

    int mult = 1 + minSteps(2*x, y);    // 2 pts
    int add = 1 + minSteps(x+1, y);     // 2 pts

    return min(add, mult);              // 2 pts
}
```

2) (10 pts) ANL (Summations and Algorithm Analysis)

Consider the following segment of code, assuming that n has been previously declared and initialized to some positive value:

```
int i, j, k;
for (i = 1; i <= n; i++){
    for(k =1; k <= i; k++){
        j = k;
        while(j > 0)
            j--;
    }
}
```

(a) (3 pts) Write a summation (3 nested sums) equal to the number of times the statement `j--;` executes, in terms of n.

$$\sum_{i=1}^{n}\sum_{k=1}^{i}\sum_{j=1}^{k} 1$$

Grading: 1/2 pt for outer sum, 1 pt for each inner sum, 1/2 for 1 inside, round down. Note - variable names used in sums are independent of those in code…

(b) (7 pts) Determine a closed form solution for the summation above in terms of n.

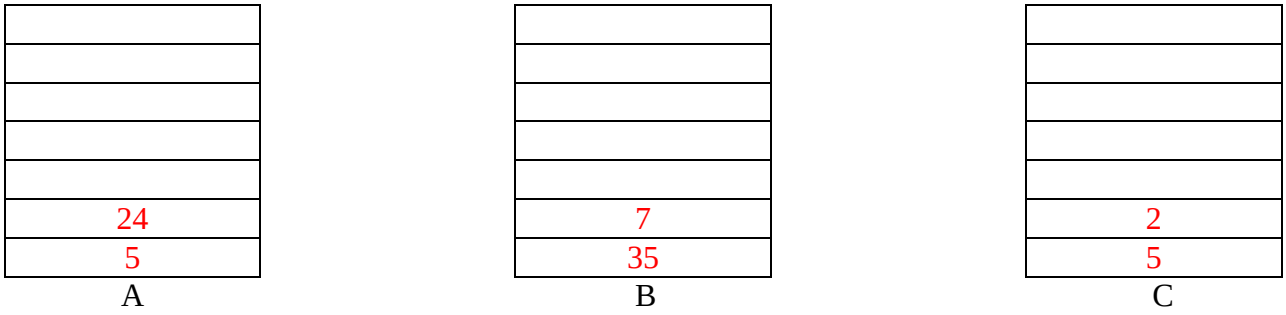
$$\sum_{i=1}^{n}\sum_{k=1}^{i}\sum_{j=1}^{k} 1 = \sum_{i=1}^{n}\sum_{k=1}^{i} k = \sum_{i=1}^{n} \frac{i(i+1)}{2}$$
$$= \frac{1}{2}\left(\sum_{i=1}^{n} i^2 + \sum_{i=1}^{n} i\right)$$
$$= \frac{1}{2}\left(\frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2}\right)$$
$$= \frac{n(n+1)}{12}((2n+1)+3)$$
$$= \frac{n(n+1)}{12}(2n+4)$$
$$= \frac{n(n+1)(n+2)}{6}$$

Grading: 1 pt solving inner sum, 1 pt solving middle sum, 2 pts for i^2 sum, 1 pt for i sum, 2 pts for algebra - accept either factored or poly form)

3) (10 pts) ALG (Stacks)

Use a stack to evaluate the postfix expression below. Please show the state of the stack at the exact point in time during the algorithm that the marked (A, B, C) locations are reached while processing the expression. Also, write down the equivalent infix expression, placing parentheses when necessary.

				A				**B**					**C**	
5	**8**	**3**	*****	**+**	**6**	**+**	**7**	**/**	**4**	**2**	**–**	*****		



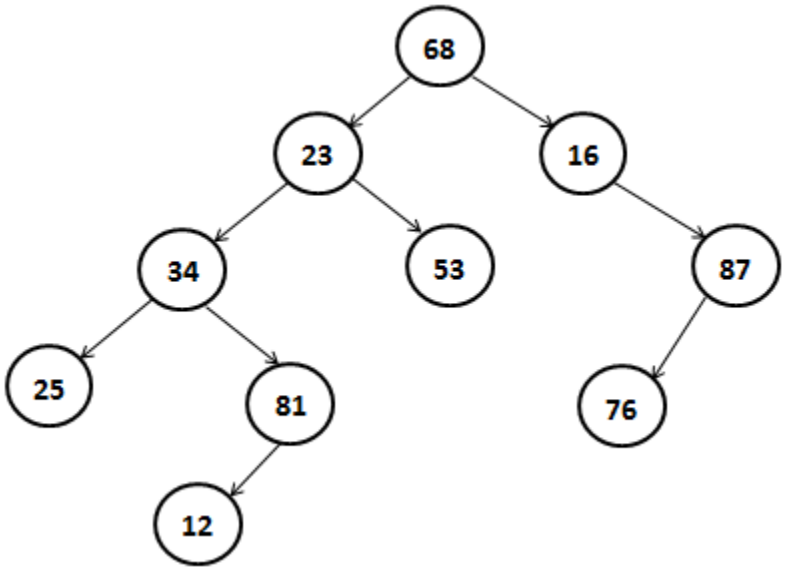
Value of the Expression: 10

Equivalent Infix Expression: (5 + (8 * 3) + 6) / 7 * (4 – 2)

Grading: 2 pts per stack, 1 pt expression, 3 pts expression - extra parens allowed, assign partial as needed.

4) (10 pts) ALG (Binary Trees)

Please give the preorder, inorder and postorder traversal of the binary tree shown below. In addition, determine whether or not the tree is a valid binary search tree.



Preorder:

Preorder: **68, 23, 34, 25, 81, 12, 53, 16, 87, 76**

Inorder: **25, 34, 12 81, 23, 53, 68, 16, 76, 87**

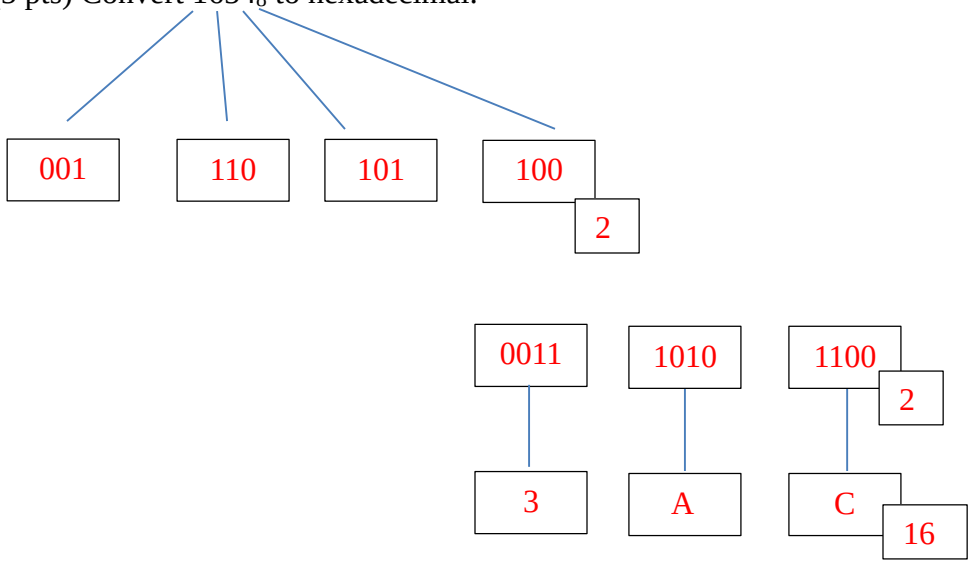
Postorder: **25, 12, 81, 34, 53, 23, 76, 87, 16, 68**

Is this valid binary search tree? (Circle your answer.) YES **NO**

Grading: 3 pts each traversal - give a total of 5 out of 9 if the traversal names are switched but the answers are the three distinct traversals. Otherwise, give partial as follows - 2 pts if 6-9 items correct, 1 pt if 3-5 items correct, 0 otherwise.

5) (10 pts) ALG (Base Conversion)

(a) (5 pts) Convert 1654_8 to hexadecimal.



$3AA_{16}$

Grading: 2 pts for converting from base 8 to base 2, 1 pt for regrouping from sets of three values to sets of four values, 2 pts for converting regrouped base 2 to base 16.

(b) (5 pts) Convert 1925_{10} to octal.

1925/8	=	240	with remainder	5
240/8	=	30	with remainder	0
30/8	=	3	with remainder	6
3/8	=	0	with remainder	3

Answer: $1928_{10} = 3605_8$

Grading: 1 pt for each division by 8 with remainder, 1 pt for final solution in correct order