Computer Science Foundation Exam

December 12, 2014

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)

(a) (5 pts) Write a recursive function `logTwo` that takes in a positive integer **x** that is a power of 2 and returns the log to the base 2 of **x.** For example `logTwo(8)` would return 3, and `logTwo(32)` would return 5.

```
int logTwo(int x) {
                                        // Grading:
    if(x == 1)                          // 1 pt
        return 0;                       // 1 pt
    return 1 + log2(x/2);               // 1 pt 1+, 1 pt log2,
                                        // 1 pt x/2
}
```

(b) (5 pts) Write a recursive function `powTwo` that takes in a positive integer **n** and returns 2^n. For example `powTwo(3)` would return 8, and `powTwo(5)` would return 32.

```
int powTwo(int x) {
                                        // Grading:
    if(x == 0)                          // 1 pt
        return 1;                       // 1 pt
    return 2 * powTwo(x-1);             // 1 pt 2*, 1 pt powTwo,
                                        // 1 pt x-1
}
```

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

(a) (5 pts) Write a summation, **but do NOT solve it,** that represents the value of the variable sum at the end of the following code segment, in terms of the variable *n*, entered by the user. (Note: your answer should have two summation signs in it and appropriate parentheses that clearly dictate the meaning of the expression you've written.)

```
int i, j, n, sum = 0;
printf("Please enter a positive integer.\n");
scanf("%d", &n);

for (i=n; i<2*n; i++) {
    sum += i;
    for (j=1; j<=i; j++)
        sum += (j*j);
}
```

$$\sum_{1=n}^{2n-1}(i+\sum_{j=1}^{i}j^2)$$

Grading: 2 pt bounds outer sum, 1 pt i term in sum, 1 pt bounds inner sum, 1 pt j^2. Note: Give the benefit of the doubt if the parentheses are missing.

(b) (5 pts) Determine the following summation in terms of n: $\sum_{i=1}^{n}(2i+3)$.

$$\sum_{i=1}^{n}(2i+3) = 2\sum_{i=1}^{n}i + 3\sum_{i=1}^{n}1 =$$

$$2\frac{n(n+1)}{2} + 3n = n^2 + 4n = n(n+4)$$

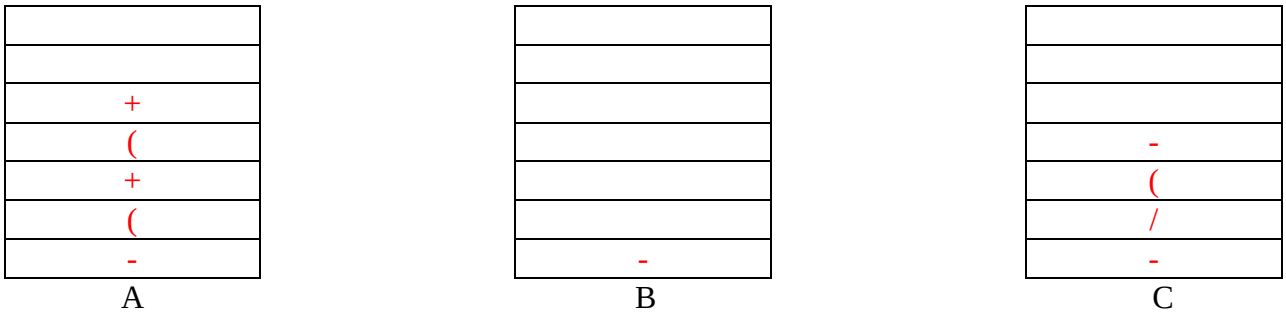
Grading: 1 pt split sum, 2 pts i formula, 1 pt const sum, 1 pt answer (doesn't need to be factorized)

3) (10 pts) ALG (Stacks)

(a) (8 pts) Convert the following infix expression to an equivalent postfix expression. Show the state of the operator stack at each of the indicated points A, B and C:

A B C

6 - (8 + (10 - 7 + 3) * 2 + 10) / ((4 - 2) * 3 - 1)



Final Postfix Expression:

6 8 10 7 - 3 + 2 * + 10 + 4 2 - 3 * 1 - / -

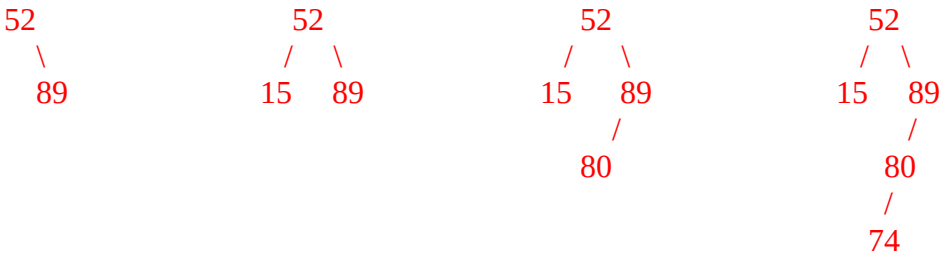
(b) (2 pts) What is the value of the postfix expression that is the result of part (a)?

0

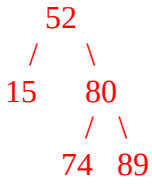
Grading: 1 pt for each stack, 5 pts for the final expression (take 1 pt off per mistake, cap at 5), 2 pts for part (b), all or nothing.

4) (10 pts) ALG (AVL Trees)

Create an AVL tree by inserting the following values into an originally empty tree. The values should be inserted in the order given: 52, 89, 15, 80, 74, and 63.



Grading: 3 pts to get to this point, BEFORE fixing the first imbalance at 89.

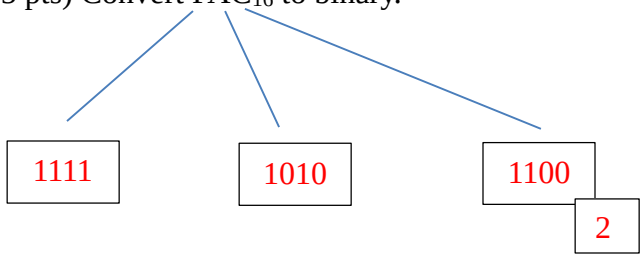


Grading: 3 pts for this fix, 1 pt for position of 80, 1 pt for 74, 1 pt for 89

Grading: 4 pts for this fix, 1 pt for 74 at root, 1 pt for 52 to left, 1 pt for 80 at right, 1 pt for rest.

5) (10 pts) ALG (Base Conversion)

(a) (5 pts) Convert FAC_{16} to binary.



Grading: If converting directly, 2 pts for the first two groups, 1 pt for the last group. If going through base 10, 2 pts for converting to base 10, 3 pts for going from base 10 to base 2.

(b) (5 pts) Convert 1235_8 to decimal.

$= 1 * 8^3 + 2 * 8^2 + 3 * 8 + 5$
$= 512 + 128 + 24 + 5$
$= 669$

Grading: 3 pts for definition breakdown in first line, 2 pts arithmetic