CMSC 341 (Practice) Midterm Exam 1 (Fall 2018)

Instructions: Clearly write your name on this sheet. Answer each problem in the space provided. If you need extra space, clearly write your name and the problem number on an extra sheet of paper, write “on extra sheet” in the answer space on the exam paper, and turn in the extra sheet with your exam.

Write legibly. If the person grading the test cannot read something, s/he will simply assume that you meant the illegible portion as a note to yourself and they will ignore it. If you lose points because part of your answer could not be read, you will not be given the opportunity to explain what it says.!

Be clear and concise. The answers to most questions should be short. If you find yourself writing an excessively long response, you may want to think more carefully about the question. Long rambling answers generally get fewer points than short ones do because there are more opportunities to mark something wrong. **The purpose of short-answer questions is to determine if you know what you are talking about, a few key words can usually communicate this.**

You may not ask questions of other students, look at another student’s exam, use a textbook, use a phone or calculator, or seek any other form of assistance. In summary: do not cheat. Persons caught cheating will be subject to disciplinary action.

Each question is marked with a number of points. There are 100 points total.

If something isn’t clear, ask!

Good luck

1. (2 pts) **Name:** ____

2. (3 pts) **GL Login:** ____

DO NOT OPEN THE TEST UNTIL TOLD TO BEGIN

3. (5 pts) **Pointers/References:** What is the output of the following code segment:

```
int i[2];
int *k;
int &l = i[0];
int &j = i[1];
k =  &i;
i[0] = 0;
i[1] = 1;
*(k+1) = 5;
l = 6;
std::cout<<"i[0] = "<<i[0]<<std::endl;
std::cout<<"i[1] = "<<i[1]<<std::endl;
```

4. (10 pts) **Proof by Induction:** Prove by induction that

$$\sum_{i=0}^{n}(i)\cdot(i+1) = \frac{n\cdot(n+1)\cdot(n+2)}{3} \tag{1}$$

5. (6 pts) $\mathcal{O}(\cdot)$**:** Give the $\Theta(\cdot)$ running times of the following loops:

```
// Loop 1
int sum = 0;
for (int i=1; i<=n; i++) {
  sum++;
}

//loop 2
int sum = 0;
for (int i=1; i<=n; i++) {
  for (j=1; j<=i; j*=2) {
    sum++;
  }
}

//loop 3
int sum = 0;
for (int i=1; i<=2n; i++) {
  for (j=1; j<=n; j*=2) {
    sum++;
  }
}
```

6. (4 pts) $\mathcal{O}(\cdot)$**:** For the function $f(x) = 5x^2 + 10x + 25$, find the minimum (integer) n_0 such that for all $n > n_0$ $f(x) < 10x^2$.

7. (10 pts) **Recurrences:** Solve the recurrence for binary search: $T(n) = T(n/2) + 1$.

8. (10 pts) **Stacks and Queues:** What does a queue implemented with a dynamically resized array that starts with size 4 look like after the following operations: `enqueue(0), enqueue(1), enqueue(2), dequeue(), enqueue(3), dequeue(), enqueue(4), enqueue(5), enqueue(6), dequeue().`

9. (5 pts) **Stacks and Queues:** Assume a stack implemented with a linked list has the following code for `pop()`:

```
void pop() {
  Node n = top->next->next;
  top->next->next = top->next->next->next;
  delete n;
}
```

   What would the `push()` method look like? (Nevermind that no one would actually write this code for `pop()`.)

10. (5 pts) **Stacks and Queues:** Assume you have a class `Stack` that provides `T top()`, `void pop()`, `void push(T data)`, and `bool isEmpty()`. It provide **NO OTHER METHODS**. Write a function `bool contains(T data)` that determines if a value is in the stack. Upon `return` the stack should be unaltered.

11. (15 pts) **Trees:** Perform the following traversals of the tree in Figure 1.

    `pre-order:`

    `post-order:`

    `in-order:`

    `level-order:`

    List the ancestors of 55:

    List the children of 80:

    List the siblings of 80:

    List the parents of 40?

    What is the height of 75?

    What is the depth of 90?

    What is the size of the subtree rooted at 60?

12. (10 pts) **Binary Search Trees:** Perform the following operations on an initially empty Binary Search Tree: `insert(50), insert(25), insert(10), insert(5), remove(25), insert(75), insert(60), insert(80), remove(50), remove(5).`

13. (15 pts) **AVL Trees:** Remove 15 from the tree in Figure 1.

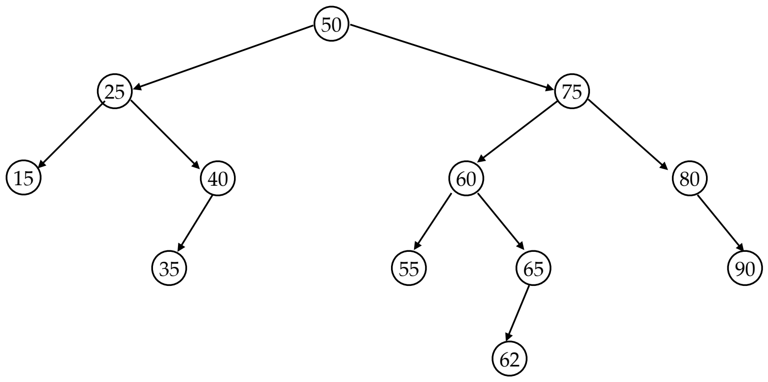

Figure 1