Name:__________________________________

Question	Points
I.	/20
II.	/30
III.	/25
IV.	/25
TOTAL:	/100

Instructions

1. This is a closed-book, closed-notes exam.
2. You have 75 minutes for the exam.
3. Calculators are not allowed.
4. Show all of your work.
5. Clearly indicate your final answer.

Definitions

The following definitions are copied verbatim from the textbook:

Definition 2.1.
$T(N) = O(f(N))$ if there are positive *constants* c and n_0 such that $T(N) \leq cf(N)$ when $N \geq n_0$.

Definition 2.2.
$T(N) = \Omega(g(N))$ if there are positive *constants* c and n_0 such that $T(N) \geq cg(N)$ when $N \geq n_0$.

Definition 2.3.
$T(N) = \Theta(h(N))$ if and only if $T(N) = O(h(N))$ and $T(N) = \Omega(h(N))$.

I. True/False (2 points each)

For each question in this section, indicate whether the statement is TRUE or FALSE. Circle **ONE** answer. Choose the **BEST** answer.

1. The function $n^2 + 15n - 4$ is $O(n)$.

   TRUE FALSE

2. The function $n^2 + 15n - 4$ is $O(n^2)$.

   TRUE FALSE

3. The function $n^2 + 15n - 4$ is $O(n^3)$.

   TRUE FALSE

4. The function $n^2 + 15n - 4$ is $\Omega(n)$.

   TRUE FALSE

5. The function $n^2 + 15n - 4$ is $\Omega(n^2)$.

   TRUE FALSE

6. The function $n^2 + 15n - 4$ is $\Omega(n^3)$.

   TRUE FALSE

7. The function $n^2 + 15n - 4$ is $\Theta(n)$.

   TRUE FALSE

8. The function $n^2 + 15n - 4$ is $\Theta(15n)$.

   TRUE FALSE

9. The function $n^2 + 15n - 4$ is $\Theta(n^2)$.

   TRUE FALSE

10. The function $n^2 + 15n - 4$ is $\Theta(n^3)$.

    TRUE FALSE

II. Short Answers (10 points each)

1. Order the following functions by growth rate. Indicate which functions grow at the same rate.

$$n^2 \log(\sqrt{n}), \quad n, \quad n^2 \log n, \quad \sqrt{n}, \quad n^2, \quad n \log(n^2), \quad n^3, \quad \log n$$

2. Argue mathematically that the function $5n^2 + 2n \log n + 3$ is $O(n^2)$. Justify your answer using the definition of $O(n^2)$.

3. Give an asymptotic upper bound on the running time of the following code fragment. Report your answer in terms of n. Briefly justify your answer. Overestimates will receive less credit.

```
int i, j ;

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

III. Short Program 1 (25 points)

Consider the following declarations for a singly-linked list data structure that uses a dummy header.

```
public class Node {
   int data ;
   Node next ;
}

public class SingleLL {
   Node header ;
   ...
}
```

Write the code for a method for the `SingleLL` class with the following signature:

```
public SingleLL reverse() ;
```

The `reverse()` method should construct a singly-linked list with the items in reverse order. The singly-linked of the host object should not be altered. For example, if the `SingleLL` object `foo` held the list 4, 5, 1, 9, 7, then calling `foo.reverse()` should return a `SingleLL` linked list that holds 7, 9, 1, 5, 4.

For full credit, your method should run in $O(n)$ time where n is the number of items in the host linked list.

```
In C++:
class Node {
   int data ;
   Node *next ;
} ;

class SingleLL {
   Node *header ;
   ...

} ;
```

IV. Short Program 2 (25 points)

Write a code fragment that works with a `LinkedList` from the Java Collections API. You should assume that just before your code fragment is executed, the variable `aList` references an object of type `LinkedList<Integer>`.

Your code fragment should look through `aList` for adjacent `Integer` values that are equal and remove one of them. At the end of your code fragment, all pairs of adjacent items in the `LinkedList` should be different. The items in the list should not be altered otherwise. For example, if the list originally held 4, 5, 5, 1, 9, 9, 9, 7, 7, then after executing your code fragment, the list should hold 4, 5, 1, 9, 7.

For full credit, your code fragment should run in $O(n)$ time where n is the number of items in the `LinkedList` referenced by `aList`. *Do include comments in your program code fragment!*