You need to print and fill out an assignment submission form. The form can be downloaded from [http://www.csd.uwo.ca/courses/CS2210a/submForm.pdf](http://www.csd.uwo.ca/courses/CS2210a/submForm.pdf).

Put your assignment in a 9”x12” envelope labeled with your name and CS2210 class number (get this number from OWL’s Gradebook) and drop it in the CS2210 locker (locker 300 located on the third floor of the Middlesex College Building, beside room MC300) by 11:59 pm on September 28.

For questions 1 and 3 proceed as follows:

1. Explain what needs to be proven: “We need to find constants $c > 0$ and $n_0 \geq 1$ integer such that . . .”.
2. For question 3 use the definition of “big Oh” to explain what it means for $f(n)$ to be $O(g(n))$ and for $h(n)$ to be $O(g(n))$.
3. Simplify the above inequalities.
4. Determine the values for $c$ and $n_0$.

For question 2, if you use a proof by contradiction:

- First write the claim that you will assume to be false and from which you will derive a contradiction.
- Perform steps 1 and 3 as above
- Derive a contradiction.

1. (3 marks) Use the definition of “big Oh” to prove that $n^3 + 400n$ is $O(n^3)$.
2. (3 marks) Use the definition of “big Oh” to prove that $n^3/9$ is not $O(n)$.
3. (3 marks) Let $f(n)$, $g(n)$, and $h(n)$ be non-negative functions such that $f(n)$ is $O(g(n))$ and $h(n)$ is $O(g(n))$. Use the definition of “big Oh” to prove that $f(n) + h(n)$ is $O(g(n))$.

4. Write an algorithm that given an array $A$ storing $n$ different integer values and an integer value $k$ it returns the value $true$ if there are two different integers in $A$ that sum to $k$, and it returns $false$ otherwise. For example, if $A$ is the following array

```
4 7 3 9 -1 2 11
```


- (4 marks) Write pseudocode for an algorithm as described above. The algorithm will receive as input $A$, $n$, and $k$.
- Prove that your algorithm is correct:
(a) (1 mark) Show that the algorithm terminates.
(b) (2 marks) Show that the algorithm always produces the correct answer.
- (1 mark) Explain what the worst case for the algorithm is.
- (3 marks) Compute the time complexity of the algorithm in the worst case. You must
give the order of the time complexity using “big-Oh” notation and you must explain how
you computed the time complexity.

5. (2 marks) **Optional question.** Download from the course’s website:
   http://www.csd.uwo.ca/Courses/CS2210a/
   the java class **Search.java**, which contains implementations of 3 different algorithms for solving
   the search problem:
   - **LinearSearch**, of time complexity \( O(n) \).
   - **QuadraticSearch**, of time complexity \( O(n^2) \).
   - **FactorialSearch**, of time complexity \( O(n!) \).

Modify the **main** method so that it computes the running times of the above algorithms for the
following input sizes:
- **FactorialSearch**, for input sizes \( n = 5, 8, 9, 10, 11 \). If you dare, run the algorithm for
  \( n = 12 \) or values larger than 12.
- **QuadraticSearch**, for input sizes \( n = 5, 10, 100, 1000, 2000 \).
- **LinearSearch** for, input sizes \( n = 5, 10, 100, 1000, 2000, 10000 \).

Print a table indicating the running times of the algorithms for the above input sizes. You do
not need to include your code for the Search class.