# CSci 3501 Assignment 3

**Due Friday, September 23rd in class**

**Problem 1 (5 points)** Consider a search algorithm for a sorted array that works as follows: given an array of $n$ elements and a search value $v$, it takes an element at the index $\frac{n}{3}$ and an element at the index $\frac{2n}{3}$ (with appropriate rounding when needed). It compares $v$ to the two elements and, if no match is found, makes a recursive call on one third of the array. For instance, if $v$ is less than the smaller "pivot" value then the call is on the first third; if $v$ is larger than the smaller pivot but smaller than the larger one then the recursive call is on the middle third of the array, and so on.

Write the recurrence for the algorithm. What is the solution of the recurrence and why? You don't need a formal proof, just an explanation.

Would you consider this algorithm to be more or less efficient than binary search? Please explain your answer.

**Problem 2 (15 points).** Use the substitution method (i.e. a proof by induction) to prove the following:

1. The solution of the recurrence $T(n) = T(\frac{n}{2}) + n$ is $O(n)$.
2. The solution of the recurrence $T(n) = 2T(\frac{n}{2}) + n^2$ is $O(n^2)$.
3. The solution of the recurrence $T(n) = 3T(\frac{n}{3}) + 1$ is $O(n)$.

Assume the base case $T(1) = 1$ for all recurrences. You may ignore the precise handling of base cases.

**Problem 3 (8 points).** Use the recurrence tree method to solve the following recurrences:

1. $T(n) = T(n-2) + n$, the base cases are $T(1) = T(0) = c$ (why do we need two base cases here?)
2. $T(n) = T(\frac{n}{3}) + 1$, the base case $T(1) = c$.