Julian Garcia
CSE 344
HW 1

I certify that every answer in this assignment is the result of my own work; that I have neither copied off the Internet nor from any one else's work; and I have not shared my answers or attempts at answers with anyone else.

1. Consider a sorting algorithm EVENTUALSORT, whose pseudo-code is given below.

```
EVENTUALSORT(A)
1  n ← A.length
2  for i ← 1 to n − 1 do
3      j ← i
4      while j ≤ n − 1 do
5          j ← j + 1
6          if A[i] > A[j] then
7              ▷ now swap A[i] with A[j]
8              EXCHANGE(A, i, j)
9              j ← i
```

Its only input $A[1 \cdots n]$ is an array of n elements which can be compared using $>$.

Algorithm EXCHANGE, given an array followed by two valid indices as input, swaps the contents of the two corresponding elements. Determine the best-case time complexity of EVENTUALSORT and the worst-case time complexity of EVENTUALSORT. For both cases,

1. create a table with three columns: *Line number* (from the pseudo-code), *Cost, Best-case Number of Times / Worst-case Number of Times*;
2. fill in appropriate entries: for a sequence, write it out using ellipsis, then in sigma-notation, and finally as a polynomial (state and prove any formula you use to sum those sequences); and
3. finally, combine those entries into a closed-form polynomial for $T(n)$, the time taken by EVENTUALSORT. Simplify as much as possible.

Assume a cost of 3 for EXCHANGE and c for all executable statements. Show your steps clearly.

First, establishing what X and Y are:

Let t_j be the #times the while-stmt executes for a particular assignment to the loop variable j. Thus,

$$X = t_1 + t_2 + \ldots + t_n = \sum_{j=1}^{n} t_j$$

$$Y = (t_1 - 1) + (t_2 - 1) + \dots + (t_n - 1) = \sum_{j=1}^{n} t_j - \sum_{j=1}^{n} 1$$

Line	Cost	#Times Worst Case	#Times Best Case
1	c_1	1	1
2	c_2	n	n
3	c_3	n-1	n - 1
4	c_4	Occurs when $t_j = j$ for all j So, in this case: As a sequence: $X = 1 + 2 + \dots + (n - 1) + n$ In sigma notation: $X = \sum_{j=1}^{n} j$ As a polynomial: $X = n(n+1) / 2$ Proof: Take our sequence: $X = 1 + 2 + \dots + (n - 1) + n$ Reverse that sequence: $X = n + (n-1) + \dots + 2 + 1$ Add these two sequences, term-by-term, each term results in n+1 so: $2X = (n + 1) + (n + 1) + \dots + (n + 1) + (n + 1) = n (n+ 1)$ Divide that result by 2: $X = n(n + 1) / 2$	Occurs when $t_j = 1$ for all j. As a sequence: $X = 1 + 1 \dots + 1$ In sigma notation: $X = \sum_{j=1}^{n} 1$ As a polynomial: $X = n$
5	c_5	Also occurs when $t_j = j$ for all j So in this case: As a sequence: $Y = (1 - 1) + (2 - 1) + \dots (n - 1 - 1) + (n - 1)$ In sigma notation: $Y = \sum_{j=1}^{n} j - \sum_{j=1}^{n} 1$ As a polynomial: $Y = (n(n+1) / 2) - n$	Occurs when $t_j = 1$ for all j. $Y = 0$

		Proof: Take our Sequence: Y = (1 - 1) + (2 - 1) + … (n - 1 - 1) + (n - 1) Extract the -1 from each term: Y = (1 + 2 + … + (n - 1) + n) - n Looking at the proof from line 4, we have already proved that: (1 + 2 + … + (n - 1) + n) = n(n + 1) / 2 Therefore, Y = (n(n+1) / 2) - n	
6	c_6	Y = (n(n+1) / 2) - n	Y = 0
7	0	Y = (n(n+1) / 2) - n	Y = 0
8	3	Y = (n(n+1) / 2) - n	Y = 0
9	c_7	Y = (n(n+1) / 2) - n	Y = 0

Best Case:
$T(n) = c_1(1) + c_2(n) + c_3(n - 1) + c_4(n)$
$T(n) = \Theta(n)$

Worst Case:
$T(n) = c_1(1) + c_2(n) + c_3(n-1) + c_4(n(n+1)/2) + c_5((n(n+1)/2) - n) + c_6((n(n+1)/2) - n) + 3((n(n+1)/2) - n) + c_7((n(n+1)/2) - n)$
$T(n) = \Theta(n^2)$

2. Assuming A, B, and C are non-zero constants and A is positive, prove that

$$An^2 + Bn + C = \Theta(n^2).$$

Let $An^2 + Bn + C$ be represented by h(n)
To prove this, we have to find that there are constants c_1 , c_2 and n_0 such that for all $n \geq n_0$:
$0 \leq c_1 * n^2 \leq h(n) \leq c_2 * n^2$

Let $n_0 = 1$
Since $n \geq 1$, we can multiply both sides by n and show that $n^2 \geq n$
Therefore, $n^2 \geq n \geq 1$
And, by the rules of inequalities, $n^2 \geq n^2$
So, we can form the inequality:
$An^2 + Bn + C \leq An^2 + |B|n^2 + |C|n^2$
Since each term on the right is equal to or greater than the term on the left.
So,

$An^2 + Bn + C \leq (A + |B| + |C|)n^2$

Now that we now $(A + |B| + |C|)n^2$ is greater than our function
we can choose our upper bound value, $c_2 = A + |B| + |C|$

To find our lower bound c_1:
Again, we know that $n^2 \geq n \geq 1$
Multiplying that by -1 we can now see that:
$-1 \geq -n \geq -n^2$

We also know that An^2 is a positive value, since A is given as positive and non-zero.
So, the simplest inequality we can form, given that we don't know the sign of A and B is:

$An^2 + Bn + C \geq An^2 - |B|n^2 - |C|n^2$
Since we know that each term on the right is equal to or less than each term on the left
So,
$An^2 + Bn + C \leq (A - |B| - |C|)n^2$

So we can choose $c_1 = A - |B| - |C|$ for our lowerbound.
Therefore, we've found values for n_0 , c_1 and c_2 such that for all $n \geq n_0$:
$0 \leq c_1 * n^2 \leq h(n) \leq c_2 * n^2$
And we have proven that $An^2 + Bn + C = \Theta(n^2)$