Due 16:00 Jan 25 (Monday). There are 100 points in this assignment.
Submit your answers (**must be typed**) in pdf file to CourSys
`https://coursys.sfu.ca/2021sp-cmpt-307-d1/`.
The work you submitted must be your own. Any collaboration and consulting outside resources must be explicitly mentioned on your submission.

1. 10 points (Ex. 1.2-2, 1.2-3 of text book)

   (a) For inputs of size n, assume insertion sort runs in $8n^2$ steps and merge sort runs in $64n \log n$ steps. For which value n does insertion sort run faster than merge sort on a same machine?

   (b) What is the smallest value of n such that an algorithm of running time $100n^2$ runs faster than an algorithm of running time 2^n on a same machine?

2. 15 points (P 1-1 of text book)

   Comparison of running times: For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t, assuming that the algorithm to solve the problem takes $f(n)$ microseconds (10^{-6} second).

	1 second	1 minute	1 hour	1 day	1 month	1 year
$\log n$						
$\sqrt{n}$						
n						
$n \log n$						
n^2						
n^3						
2^n						
$n!$						

3. 10 points (Ex 2.1-3 of text book)

   Consider the searching problem:

   Input: A sequence of n numbers $A = (a_1, a_2, .., a_n)$ and a value v.
   Output: An index i such that $v = A[i]$ or the special value nil if v is not in A.

   Write pseudocode for linear search which scans through the sequence, looking for v. Using a loop invariant, prove your algorithm is correct. Make sure your loop invariant fulfills the three necessary properties.

4. 15 points (Ex. 2.1-4)

   Consider the problem of adding two n-bit binary integers, stored in two n-element arrays A and B. The sum of the two integers is stored in binary form in an $(n + 1)$-element array C. State the problem formally and write pseudocode for adding the two integers.

5. 10 points (Ex 2.3-5 of text book)

   Referring back to the searching problem in (Ex 2.1-3), observe that if the sequence A is sorted, we can check the midpoint of A and eliminate half of the elements of A from further consideration. The binary search algorithm repeats this procedure, halving the size of the remaining portion of A each time. Write pseudocode, either iterative or recursive, for binary search. Argue that the worst-case running time of binary search is $\Theta(\log n)$.

6. 15 points (P 3-2 of text book)

   Relative asymptotic growths: Indicate, for each pair of expressions (A, B) in the table below, whether A is O, o, Ω, ω, or Θ of B. Assume that $k \geq 1$, $\epsilon > 0$, and $c > 1$ are constants. Your answer should be in the form of the table with yes or no written in each box.

A	B	O	o	Ω	ω	Θ
$\log^k n$	n^ϵ					
n^k	c^n					
2^n	$2^{n/2}$					
$n^{\log c}$	c^{logn}					
$\log(n!)$	$\log(n^n)$					

7. 15 points

   Implement the brute-force and divide-and-conquer algorithms for the maximum subarray problem, and compare the running times of the two algorithms on a same machine. Submit the source codes of your implementations, report the running times of your implementations for $n = 10, 20, 50, 100, 200, 500, 1000$ and the minimum n_0 that for every $n \geq n_0$, the divide-and-conquer algorithm runs faster than the brut-force algorithm.

8. 10 points (Ex 4.5-1 of text book)

   Use the master method to give tight asymptotic bounds for the following recurrences. (a) $T(n) = 2T(n/4) + 1$. (b) $T(n) = 2T(n/4) + \sqrt{n}$. (c) $T(n) = 2T(n/4) + n$. (d) $T(n) = 2T(n/4) + n^2$.