RULES OF THE GAME:

- Solutions need to be submitted through Gradescope.
- If you would like to learn the LaTeX typesettting system and use it for assignment submissions, then please take a look at this template for what it can do.
- If you plan to write your answers on paper, then you will need to scan your work:

App	Platform	Description
Adobe Scan	iOS, Android	Great quality, but requires an Adobe account
Microsoft Office Lens	iOS, Android, Windows 10	Works on laptops too
Evernote Scannable	iOS	Recommended by Gradescope
Genius Scan	iOS, Android	Recommended by Gradescope

- Please consult Gradescope's Scanning Work on a Mobile Device page and watch the Submitting PDF Homework video.

OUR LATENESS POLICY:

- You have one slip day for **each** assignment you can use without having to notify us. **Work submitted beyond the slip day without an academically valid excuse will not earn any points.**
- The deadline to submit documents regarding academically valid excuses is the original deadline of the assignment (i.e. not that of the slip day).
- In order to reward complying with deadlines, anyone who uses only one slip day will have a 1% added to their final score and anyone who uses zero slip days will a 2% added to their final score.

REGARDING ACADEMIC INTEGRITY:

- You may collaborate with one or two of your peers in the course to brainstorm for solving the assigned problems. However, your solution must be written up completely on your own.
- If you collaborate with others during the brainstorming part of the assignment, you must list their names at the beginning of your submission.
- You cannot use any online resources, any solutions from past semesters, etc. You are not allowed to share digital or written notes or images of your work in any form.
- Even when a question feels overwhelming, you should not compromise your academic integrity. Since the exam questions will be similar to the assigned exercises/problems, not struggling with these questions (in the context of assignments) will be counter productive with respect to grades.

QUESTIONS/PROBLEMS:

1. There is a video in our Zoom repository that shows you the process of doing an empirical comparison between Insertion Sort and Selection sort. Following the same process, do an empirical comparison between MERGE-SORT and QUICK-SORT. Use "number of numbers being sorted" and "the percentage of disorder" as your independent variables; use "completion time" as your only dependent variable. Plot your findings and decide which of these two algorithms runs faster; more importantly, quantify the speed difference between MERGE-SORT and QUICK-SORT:

   - Video: [CS 2110: Collecting Empirical Data (Selection vs Insertion sort)]
   - Spreadsheet shown in the video: [Empirical Data on Selection/Insertion Sort]
   - Java code used in the video: [Java package ps_2_ali]

2. Using either MERGE-SORT or QUICK-SORT, do a study similar to tht one in step 1 to understand the degree to which assertions make a difference in runtime.

3. Using the following array of integers, illustrate the operation of MERGE-SORT; that is, identify the particular recursive calls made, to lo/hi values, etc. Your definitive guide for MERGE-SORT should be [the implementation from our book].

7	9	5	12	4	8	2	11	6	3	10	1

4. Using the following array of integers, illustrate the operation of QUICK-SORT; that is, identify the particular recursive calls made, pivot value selections, etc. Your definitive guide for QUICK-SORT should be [the implementation from our book].

7	9	5	12	4	8	2	11	6	3	10	1

5. Using the following array of integers, illustrate the operation of QUICK-SELECT for finding the 10th ranking numbers; that is, identify the particular recursive calls made, pivot value selections, etc. Your definitive guide for QUICK-SELECT should be [the implementation from our book].

7	9	5	12	4	8	2	11	6	3	10	1

6. Suppose you have N distinct points in a three dimensional space[1]. That is, each point p_i has three coordinates as in $p_i = (x_i, y_i, z_i)$. Your task is to order these points according to the following logic: for any two chosen points $p_i = (x_i, y_i, z_i)$ and $p_j = (x_j, y_j, z_j)$

   - If $z_i > z_j$, then $p_i > p_j$.
   - If $z_i = z_j$ but $y_i > y_j$, then $p_i > p_j$.
   - If $z_i = z_j$ and $y_i = y_j$ but $x_i > x_j$, then $p_i > p_j$.

   Provide an efficient algorithm to solve this problem.

[1]That is, it is safe to assume that no two points are the same.