MIE245 Lab 2: Sorting Algorithms I
Insertion Sort and Merge Sort

Winter 2026

Deadline: Tuesday, January 20, 11:59pm EST

Submission Instructions

- **Reminder:** You can not claim work that was completed by others or generative AI as your own.
- Submit a completed `readme` template file outlining what code you contributed and what code was completed by other students, people, tools, and/or leveraged from websites.
- Add **_utorid** (with **utorid** replaced by your UTORid in all lower-case characters) to the end of both the completed template file names (i.e., the `.py` and the `readme` files).
- Please ensure that you use your UTORid (i.e., what you use to sign in to Quercus) and not other identifiers, such as your student number.
- Compress your completed template files to a zip file and upload the zip file to Quercus.

Lab Instructions

In this lab, you will implement two fundamental sorting algorithms: **Insertion Sort** and **Merge Sort**.

Sorting is a computational process of arranging elements in a list in a specific order (e.g., ascending). Efficient sorting is critical for optimizing other algorithms (such as search) and data processing tasks.

For this lab, complete the template file `MIE245_Lab_2_Template.py`. You must implement the following methods of the Sorter class:

- `insertion_sort(self, input_list)`: Sorts the `input_list` of integers in ascending order using the Insertion Sort algorithm.
- `merge_sort(self, input_list)`: Sorts the `input_list` of integers in ascending order using the Merge Sort algorithm (a recursive divide-and-conquer approach).

Note: Both functions should return the sorted list. You must implement the logic manually; do not use Python's built-in `.sort()` or `sorted()`.

AutoGrading

Additionally, you have access to an example of the auto-grading script that will be used to test your submission (`MIE245_Lab_2_Autograder.py`). Please follow the steps below to run the auto-grader for the provided test cases:

1. Place the auto-grader script in a new folder.
2. Inside the newly created folder, create a folder called "submissions".
3. Place your script (renamed with your UTORid) in the "submissions" folder.
4. Run the auto-grader script and open the `grades.csv` file to see if you passed the test case (1 is pass, 0 is fail).

Note: The auto-grading script only contains one of the several test cases we will evaluate against, so even if you pass this test case, you may not have a completely correct solution.
Tip: You can modify the test case to test additional functionality or edge cases (e.g., empty lists, negative numbers).