# CS 284 C: Quiz 7
# Spring 2022

## Exercise 1 (*10 points*)

In this problem, you need to implement a method `tuple_sort` for sorting tuples. To do that, you also need to implement the method `compareTo` in the inner class `Tuple`.

To compare two tuple $(a_1, \cdots, a_m)$ and $(b_1, \cdots, b_n)$, find the leftmost position $k$ where $a_k \neq b_k$; the tuple with larger number is larger, therefore

$$(1,2) < (1,3)$$
$$(1,2,1000) < (1,3)$$
$$(1,2) < (1,2,1)$$

The values are always non-negative integers, and the last position is always a positive integer, i.e., a tuple cannot be (1, 2, 0).

**Requirement**. Download the code template from Canvas. Use any sorting algorithm you prefer, feel free to copy/paste code from https://liususan091219.github.io/teaching/cs284_21s/java/week_11/Sort.java. You need to implement the following two functions:

- `Tuple.compareTo(Tuple other_tuple)`: return -1 if this tuple < other tuple, return 1 if this tuple > other tuple, return 0 if they are equivalent.
- `tuple_sort`: your sorting algorithm. You can choose from any sorting algorithm in Sort.java.

**Test cases**.

```
public void test_tuple_sort() {
    Tuple[] test_tuple = new Tuple[5];
    test_tuple[0] = new Tuple(new int[] {1, 2});
    test_tuple[1] = new Tuple(new int[] {2});
    test_tuple[2] = new Tuple(new int[] {1, 1, 1});
    test_tuple[3] = new Tuple(new int[] {1, 5, 0, 5});
    test_tuple[4] = new Tuple(new int[] {1, 5, -1});

    System.out.println("Before sorting: ");
    this.print_tuple_array(test_tuple);

    Tuple[] sorted_tuple = this.tuple_sort(test_tuple);

    System.out.println("After sorting: ");
    this.print_tuple_array(test_tuple);
}
```

For the above test case, your code should output:

```
Before sorting:
(1,2), (2), (1,1,1), (1,5,0,5), (1,5,-1)
After sorting:
(1,1,1), (1,2), (1,5,-1), (1,5,0,5), (2)
```