Group 6

Humza Haider & Shawn Seymour

Sorting Competition Results

Placement:

- Placed **6th** in the competition.
- Accounting for disqualifications, we placed **5th** in the competition and **4th** in the class.

Correctness:

- Our algorithm correctly sorted in every run of the data. However, an error was pointed out in our analysis.
  - We modified a global variable (`threshold`) which was against the rules.
  - Each run of our `sort` method would square our `threshold` value.
  - Eventually, the `threshold` would change too much and we'd likely get incorrect output; though this never happened in any of our testing.

Data Storage and Algorithm

Data Storage:

- Data was stored in our own `Banana` data structure:
  - `int x, y, time; double distance;`
- Implements `Comparable<Banana>` to compare the distances of two `Banana` objects.

Algorithm:

- We stuck with Java's `Arrays.sort()` which is an implementation of Timsort.
- We figured it would be quite fast if we could optimize the calculations going on before the sorting actually happens.

Sorting Algorithm Analysis

Timsort:

- Derivation of merge sort and insertion sort.
  - Worst-case: $\theta(n \log n)$.
  - Average-case: $\theta(n \log n)$.
  - Best-case (sorted data): $\theta(n)$.

Constants & Memory:

- Two linear $\theta(n)$ loops outside of sorting for data storage & retrieval.
- We use one array of size `n` to store our `Banana` objects.