Group 5

Ellis Weglewski & Keenan McCall

Initial Ideas

- Explore possibilities of XOR
  - All strings are fixed length binary vectors in a sense
  - Possible to XOR in the right order?
  - Possible spatial sorting - too much
- Tie-breaker operation is run on every comparison?
  - Change it to run only when needed?

Specifics

- The worst/expected case for N and L, (number of strings, length of strings)
  - The worst case for N and L in this case would be O(L * N Log N)
  - The average/expected case for N and L would be O(L * N Log N)
- There were no issues with correctness reported.
- The data is stored as strings in an array, would not be considered objects

Starting Code

```
public int compare(String s1, String s2) {
    // first criterion: the number of bits different from the target string:

    int count1 = distanceToTarget(s1);
    int count2 = distanceToTarget(s2);

    BigInteger n1 = new BigInteger(s1, 2); // converting a binary number into a BigInteger
    BigInteger n2 = new BigInteger(s2, 2); // converting a binary number into a BigInteger


    if (count1 - count2 != 0) { // if the difference from the target string aren't the same

        return (count1 - count2); // return a number < 0 if the first string is closer, > 0 if the second one is
    }

        // if the two counts are the same
    return n1.compareTo(n2);
```

Final Code

```
public int compare(String s1, String s2) {
    // first criterion: the number of bits different from the target string:

    int count1 = distanceToTarget(s1);
    int count2 = distanceToTarget(s2);

    if (count1 - count2 != 0) { // if the difference from the target string aren't the same

        return (count1 - count2); // return a number < 0 if the first string is closer, > 0 if the second one is
    }
    else {

        BigInteger n1 = new BigInteger(s1, 2); // converting a binary number into a BigInteger
        BigInteger n2 = new BigInteger(s2, 2); // converting a binary number into a BigInteger
        // if the two counts are the same
        return n1.compareTo(n2);

    }
```

Results

- Round 1
  - Median: 762.0, 4th
- Round 2
  - Median: 1953.0, 7th
- Round 3
  - Median: 861.0, 11th

Other Ideas

- Radix Sort in regards to the target string