CSSE 220

Merge Sort
Comparable/Comparator

Checkout *MergeSortSimple* project from SVN

Today's Plan

- Merge sort
- How to use Java's sort functions (Comparable and Comparator)

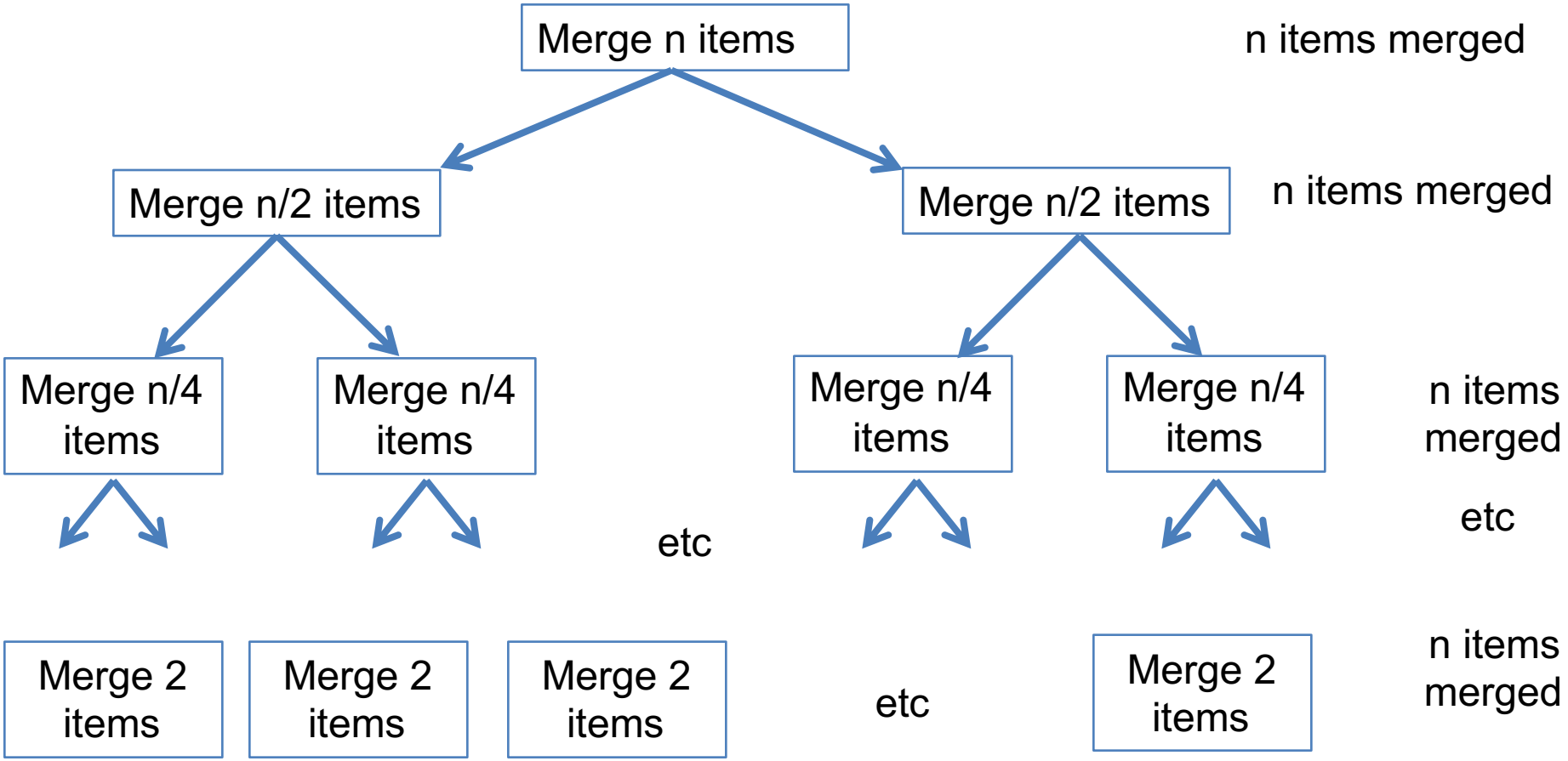
Merge Sort

- Basic recursive idea:
  - If list is length 0 or 1, then it's already sorted
  - Otherwise:
    - Divide list into two halves
    - Recursively sort the two halves
    - **Merge** the sorted halves back together

Analyzing Merge Sort

If list is length 0 or 1,
then it's already sorted

- Otherwise:
  - Divide list into two halves
  - Recursively sort the two halves
  - **Merge** the sorted halves back together



How to Sort in Java

- For arrays:

```
Arrays.sort(myArray);
```

- For ArrayLists or other stuff:

```
Collections.sort(myArrayList)
```

- For stuff like Strings and ints, the expected sorting is already built in. But what if you have a new class you want to sort?

When Your Object is Sortable

- You should implement the Comparable<YourObjectType> interface
- You need to implement 1 method: compareTo
- See section 10.3 of your text for details
- Let’s do an example

A Sort of a Different Order

- Java libraries provide efficient sorting algorithms
  - `Arrays.sort(…)` and `Collections.sort(…)`

- But suppose we want to sort by something other than the "natural order" given by `compareTo()`

- *Function objects* to the rescue!

Function Objects

- Objects defined to just “wrap up” functions so we can pass them to other (library) code

- For sorting we can create a function object that implements `Comparator`

- `Let’s try it!`