

MergeSortWithQueue.java

```
package sort;

import java.util.List;
import java.util.Queue;
import java.util.concurrent.ConcurrentLinkedQueue;
import myLib.utils.Utils;

/**
 * Merge Sort
 *
 * @author tadaki
 * @param <T>
 */
public class MergeSortWithQueue<T extends Comparable<T>>
    extends AbstractSort<T> {

    public MergeSortWithQueue(T[] d) {
        super(d);
    }

    @Override
    public List<T> sort() {
        return sortSub(list);
    }

    /**
     * 処理の再帰呼び出し
     *
     * @param tList
     * @return
     */
    private List<T> sortSub(List<T> tList) {
        Queue<List<T>> queue = new ConcurrentLinkedQueue<>();
        //すべての要素をリストにして待ち行列へ
        tList.stream().forEach(t -> {
            List<T> tt = Utils.createList();
            tt.add(t);
            queue.add(tt);
        });
        //二つ毎にマージ
        while (queue.size() > 1) {
            mergeListWithQueue(queue);
        }
        return queue.poll();
    }
}
```

MergeSortWithQueue.java

```
/**
 * 待ち行列から二つ取り出して、マージした結果を追加
 * @param <T>
 * @param queue
 * @return
 */
public static <T extends Comparable<T>> Queue<List<T>>
    mergeListWithQueue(Queue<List<T>> queue) {
    List<T> t1 = queue.poll();
    List<T> t2 = queue.poll();
    List<T> tOut = mergeList(t1, t2);
    queue.add(tOut);
    return queue;
}

/**
 * 二つのリストのmerge
 *
 * @param <T>
 * @param a
 * @param b
 * @return
 */
public static <T extends Comparable<T>> List<T>
    mergeList(List<T> a, List<T> b) {
    List<T> c = Utils.createList();
    while ((!a.isEmpty()) || (!b.isEmpty())) {
        T t;
        if (!a.isEmpty()) {
            if (!b.isEmpty()) {
                if (a.get(0).compareTo(b.get(0)) < 0) {
                    t = a.remove(0);
                } else {
                    t = b.remove(0);
                }
            } else {
                t = a.remove(0);
            }
        } else {
            t = b.remove(0);
        }
        c.add(t);
    }
    return c;
}
```

MergeSortWithQueue.java

```
/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    Integer data[] = {3, 6, 2, 9, 1, 6, 2, 8};
    MergeSortWithQueue<Integer> sort = new MergeSortWithQueue<>(data);
    List<Integer> list = sort.sort();
    AbstractSort.printList(list);
}
}
```