

```
1 import java.util.ArrayList;
2 import java.util.Arrays;
3 import java.util.List;
4 import lettersort.LetterSorter;
5 import mergesort.MergeSorter;
6
7 class Main {
8     public static void main(String args[]) {
9
10         // List<Integer> input = Arrays.asList(6, 5, 12, 10, 9, 1, 7);
11         List<String> input = Arrays.asList("c", "b", "e", "A", "M", "i");
12
13         LetterSorter letterSorter = new LetterSorter();
14         MergeSorter mergeSorter = new MergeSorter();
15
16         List<Integer> numberList = letterSorter.convertLettersToNumbers(input);
17         List<Integer> sortedList = mergeSorter.mergeSort(numberList);
18         List<String> sortedLetterList = letterSorter.convertNumbersToLetters(sortedList);
19
20         System.out.println("Sorted Array:");
21         System.out.println(sortedLetterList.toString());
22     }
23 }
```

```

1  package mergesort;
2
3  import java.util.ArrayList;
4  import java.util.Arrays;
5  import java.util.List;
6
7  public class MergeSorter {
8
9      public List<Integer> mergeSort(List<Integer> inputArray) {
10         // There's only 1 element so return the array.
11         if (inputArray.size() < 2){
12             return inputArray;
13         }
14
15         // Find the midpoint and split into two lists.
16         int midpoint = inputArray.size() / 2;
17         List<Integer> left = inputArray.subList(0, midpoint);
18         List<Integer> right = inputArray.subList(midpoint, inputArray.size());
19
20         // Sort both lists.
21         List<Integer> leftSorted = mergeSort(left);
22         List<Integer> rightSorted = mergeSort(right);
23
24         // Merge the sorted lists by keeping a pointer to the first unused number in each list.
25         // Compare the first unused item in each list and add the lowest to the combined list.
26         // Stop when the end of either list has been reached.
27         ArrayList<Integer> combinedSorted = new ArrayList<Integer>();
28         int leftIndex = 0;
29         int rightIndex = 0;
30         while (leftIndex < leftSorted.size() && rightIndex < rightSorted.size()) {
31             if (leftSorted.get(leftIndex) <= rightSorted.get(rightIndex)){
32                 combinedSorted.add(leftSorted.get(leftIndex));
33                 leftIndex++;
34             } else {
35                 combinedSorted.add(rightSorted.get(rightIndex));
36                 rightIndex++;
37             }
38         }
39
40         // Add the remaining items in the list where the end was not reached.
41         if (leftIndex == leftSorted.size()) {
42             combinedSorted.addAll(rightSorted.subList(rightIndex, rightSorted.size()));
43         } else {
44             combinedSorted.addAll(leftSorted.subList(leftIndex, leftSorted.size()));
45         }
46
47         return combinedSorted;
48     }
49 }

```

```
1 package lettersort;
2
3 import java.util.ArrayList;
4 import java.util.Arrays;
5 import java.util.List;
6
7 public class LetterSorter {
8     public final List<String> CHAR_LIST = Arrays.asList(
9         "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m",
10        "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z",
11        "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M",
12        "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z");
13
14     public List<Integer> convertLettersToNumbers(List<String> letterList) {
15         List<Integer> numberList = new ArrayList<Integer>();
16         // Loop through the letters and convert it to a number.
17         for (int index = 0; index < letterList.size(); index++) {
18             for (int charIndex = 0; charIndex < CHAR_LIST.size(); charIndex++) {
19                 // If the character in the CHAR_LIST matches add the index to the numberList.
20                 if (CHAR_LIST.get(charIndex) == letterList.get(index)) {
21                     numberList.add(charIndex);
22                     continue;
23                 }
24             }
25
26             // The character was not found so throw an error.
27             if (numberList.size() != index + 1) {
28                 throw new RuntimeException(
29                     String.format("Invalid character: %s cannot be converted to a number.",
30                         letterList.get(index)));
31             }
32         }
33
34         return numberList;
35     }
36
37     public List<String> convertNumbersToLetters(List<Integer> numberList) {
38         List<String> letterList = new ArrayList<String>();
39         for (int index = 0; index < numberList.size(); index++) {
40             // The number does not match a character so throw an error.
41             if (numberList.get(index) > CHAR_LIST.size() || numberList.get(index) < 0) {
42                 throw new RuntimeException(
43                     String.format("Invalid number: %d cannot be converted to a letter.",
44                         numberList.get(index)));
45             }
46
47             // Look up the character that matches the number in the CHAR_LIST.
48             letterList.add(CHAR_LIST.get(numberList.get(index)));
49         }
50         return letterList;
51     }
52 }
```