Conclusion:

First of all, In the both merge1 and merge2 we have a third array which its size is equal the summation sizes of array 1 and array 2. In the first merge, we are inserting the data of first and second array by the insert binary in the final array. But is important to know insert binary belong to in order array which means when we are trying to insert them in a third array as the final we are doing something very inefficient. We need to do all calculation of the insert binary which means to compare any new element with previous ones before inserting, while we know that they are in order.

In the second merge we are doing something much better and more efficient because before we want to insert any new element we are comparing the position of each element by seeing which is greater, less or equal. Then after comparing we insert them in the third or final array and then we increment the pointer by one.

The terms which makes the merging much more efficient:

1. In the first merge, we could copy the array1 in the final ones and then using the insert binary for array2 which means we are eliminate one inefficient binary from one of the loops.
2. Second way is declaring the array 1 and array2 in type of regular array ([]) instead of ordArray. Because the ordArrya makes them in order and when we want to use the insert binary we are using something against the same case that causes very inefficiency in our process. So, we declare them in the type of regular array and then use the insert binary, first to make them in order and then insert the new elements.