


```
class MergeSort
```

```
{
```

```
    static void mergesort (int[] x, int p, int r)
```

```
    {
```

```
        // sort the elements x[p], x[p+1], ... , x[r-1]
```

```
        if (p < r-1) // at least two elements to be sorted
```

```
        {
```

```
            int q = (p+r)/2;
```

```
            mergesort (x, p, q);
```

```
            mergesort (x, q, r);
```

```
            merge (x, p, q, r);
```

```
        }
```

```
    }
```

```
    static void merge (int[] x, int p, int q, int r)
```

```
        // merge the two sorted sequences
```

```
        // x[p],... , x[q-1]    and    x[q], ... , x[r-1]
```

```
    {
```

```
        int[] scratchpad = new int[q-p];
```

```
        // copy the first half to scratchpad
```

```
        for (int i=p; i < q; i++)
```

```
            scratchpad[i-p] = x[i];
```

```
        int lower = 0; // index to scratchpad
```

```
        int upper = q ;
```

```
for (int i=p; i<r ; i++)
{
    if (upper >= r || scratchpad[lower] <= x[upper])
        // upper part empty, or lower element next
        {
            x[i] = scratchpad[lower];
            lower++;
            if (lower >= q-p) // scratchpad empty
                break; // all done!
        }
    else
    {
        x[i] = x[upper];
        upper++;
    }
}
}
}
```