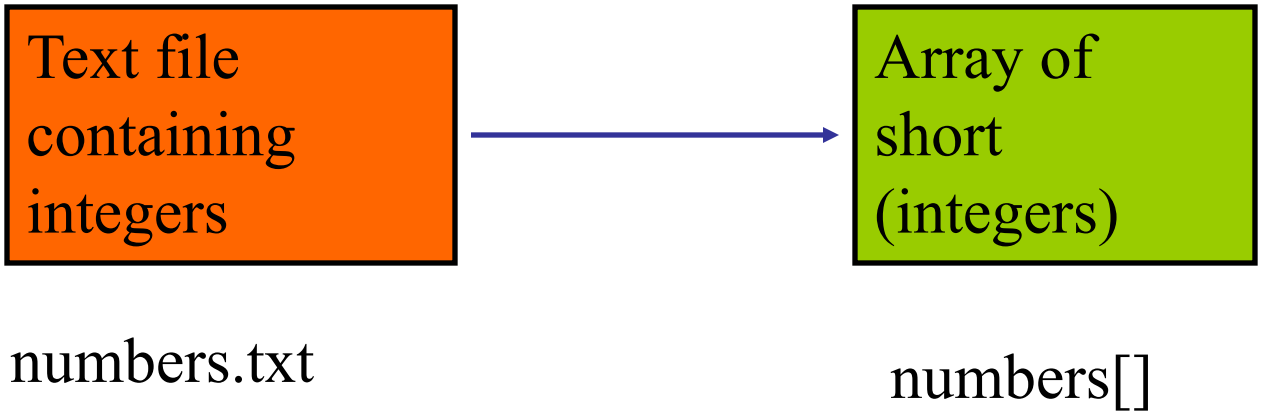
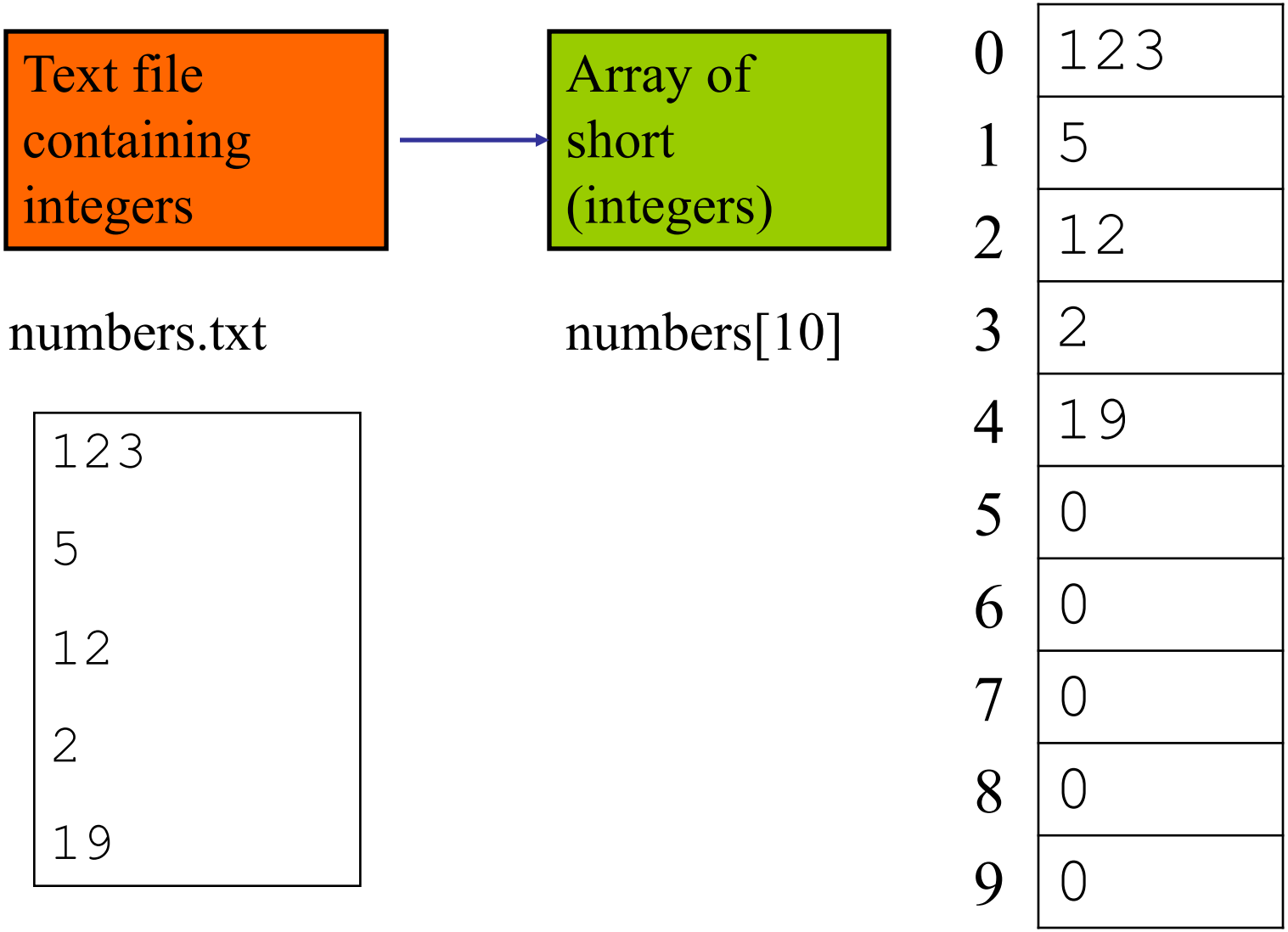
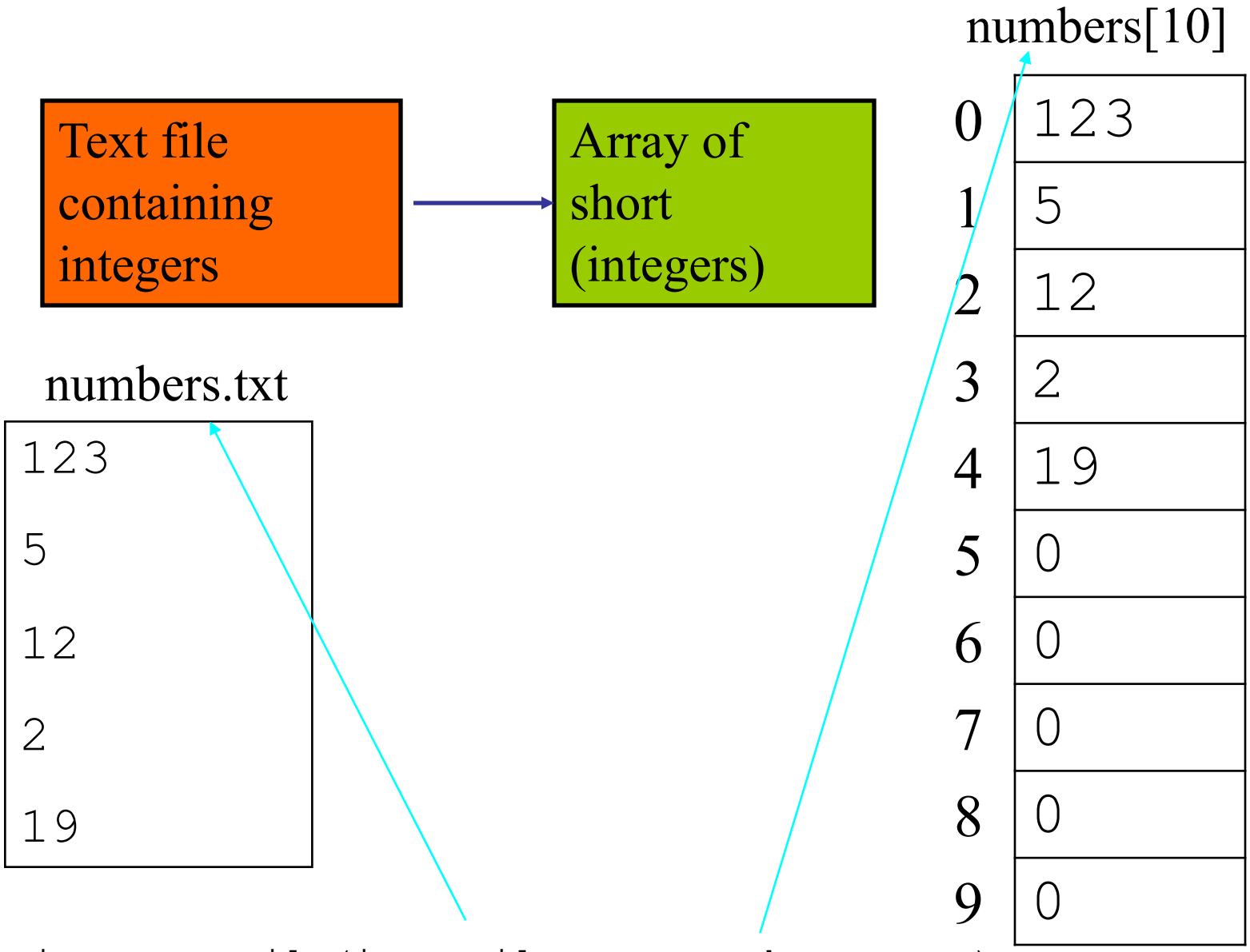
Arrays and Sorting

Process

- Open a file that contains integers, one per line.
- Read each line, convert to *short* and store each into an array
- Sort the array
- Output the sorted array


Text file containing integers
Array of short (integers)
numbers.txt
numbers[]


Text file containing integers
Array of short (integers)
numbers.txt
numbers[10]
123
5
12
2
19
0 123
1 5
2 12
3 2
4 19
5 0
6 0
7 0
8 0
9 0




```
inputFromFile(inputFileName, numbersArray);
```

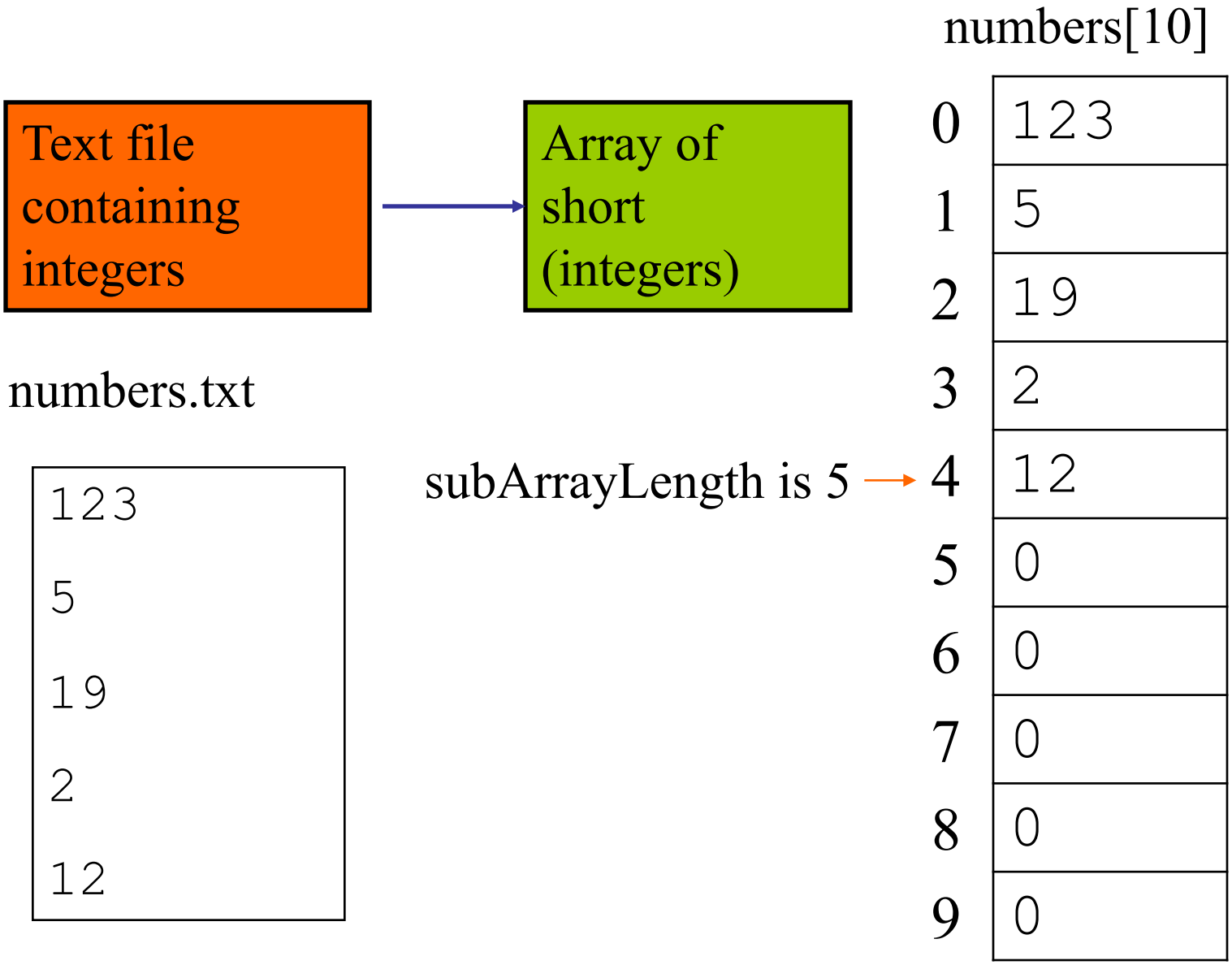
Main program calls

```
subArrayLength = inputFromFile(inputFileName, numbersArray);
```

```
private static int inputFromFile(String filename, short[] numbers){
   TextFileInput in = new TextFileInput(filename);
   int lengthFilled = 0;
   String line = in.readLine();
   while ( lengthFilled < numbers.length && line != null ) {
      numbers[lengthFilled++] = Short.parseShort(line);
      line = in.readLine();
   } // while

   if ( line != null ) {
      System.out.println("File contains too many numbers.");
      System.out.println("This program can process only " +
                          numbers.length + " numbers.");
      System.exit(1);
    } // if
   in.close();
   return lengthFilled;
} // method inputFromFile
```

Program 1.1


numbers[10]
Text file containing integers
Array of short (integers)
numbers.txt
123
5
19
2
12
subArrayLength is 5
0 123
1 5
2 19
3 2
4 12
5 0
6 0
7 0
8 0
9 0

numbers[10]

0	123
1	5
2	19
3	2
4	12
5	0
6	0
7	0
8	0
9	0

subArrayLength 5

Partially-filled array

NO:

```
for (int i =0;i<numbers.length; i ++) {
```

YES:

```
for (int i =0;i<subArrayLength; i ++) {
```

numbers[10]

0	123
1	5
2	19
3	2
4	12
5	0
6	0
7	0
8	0
9	0

```
//  average the numbers
int sum=0;
for (int i =0; i<subArrayLength; i ++)
   sum += numbers[i];
Average = sum/subArrayLength;
```

subArrayLength 5

numbers[10]

0	123
1	5
2	19
3	2
4	12
5	0
6	0
7	0
8	0
9	0

```
//  find the smallest number
short smallest = numbers[0];
for (int i =1; i<subArrayLength; i ++)
 smallest = Math.min(smallest,numbers[i];
```

subArrayLength 5

numbers[10]

0	123
1	5
2	19
3	2
4	12
5	0
6	0
7	0
8	0
9	0

```
//  find the index of the smallest number
int indexLowest = 0;
for ( int j = 1; j < subArrayLength; j++ )
    if ( array[j] < array[indexLowest] )
        indexLowest = j;
```

subArrayLength 5

numbers[10]

0	123
1	5
2	19
3	2
4	12
5	0
6	0
7	0
8	0
9	0

```
//  find the index of the smallest number
```

This is the basis of "Selection Sort"

numbers[10]

0	123
1	5
2	12
3	2
4	19
5	0
6	0
7	0
8	0
9	0

find the *index* of the smallest number

This is the basis of "Selection Sort"

Find the smallest number and swap it with the number at the top of the array

numbers[10]

0	2
1	5
2	12
3	123
4	19
5	0
6	0
7	0
8	0
9	0

numbers[10]

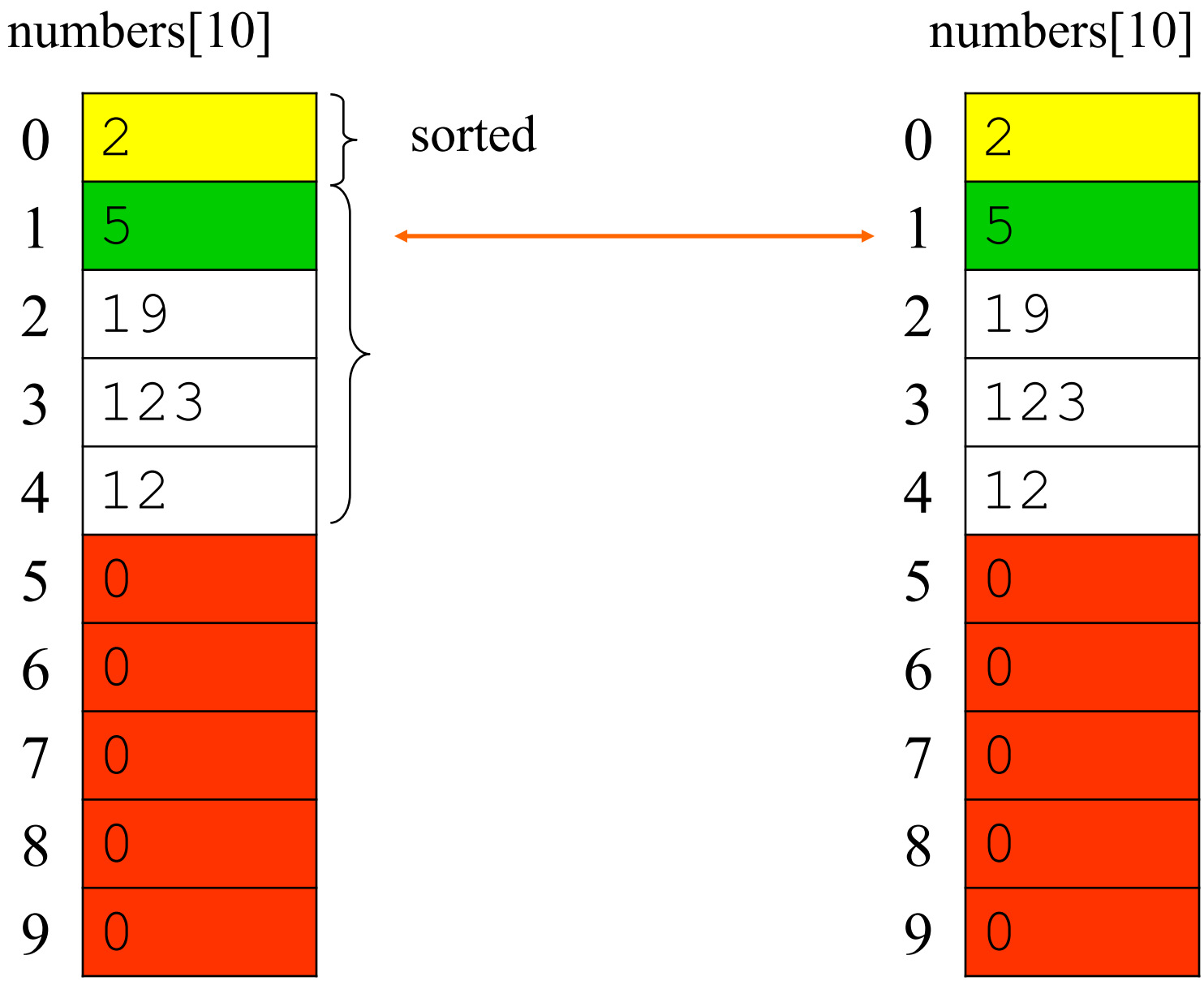
numbers[10]

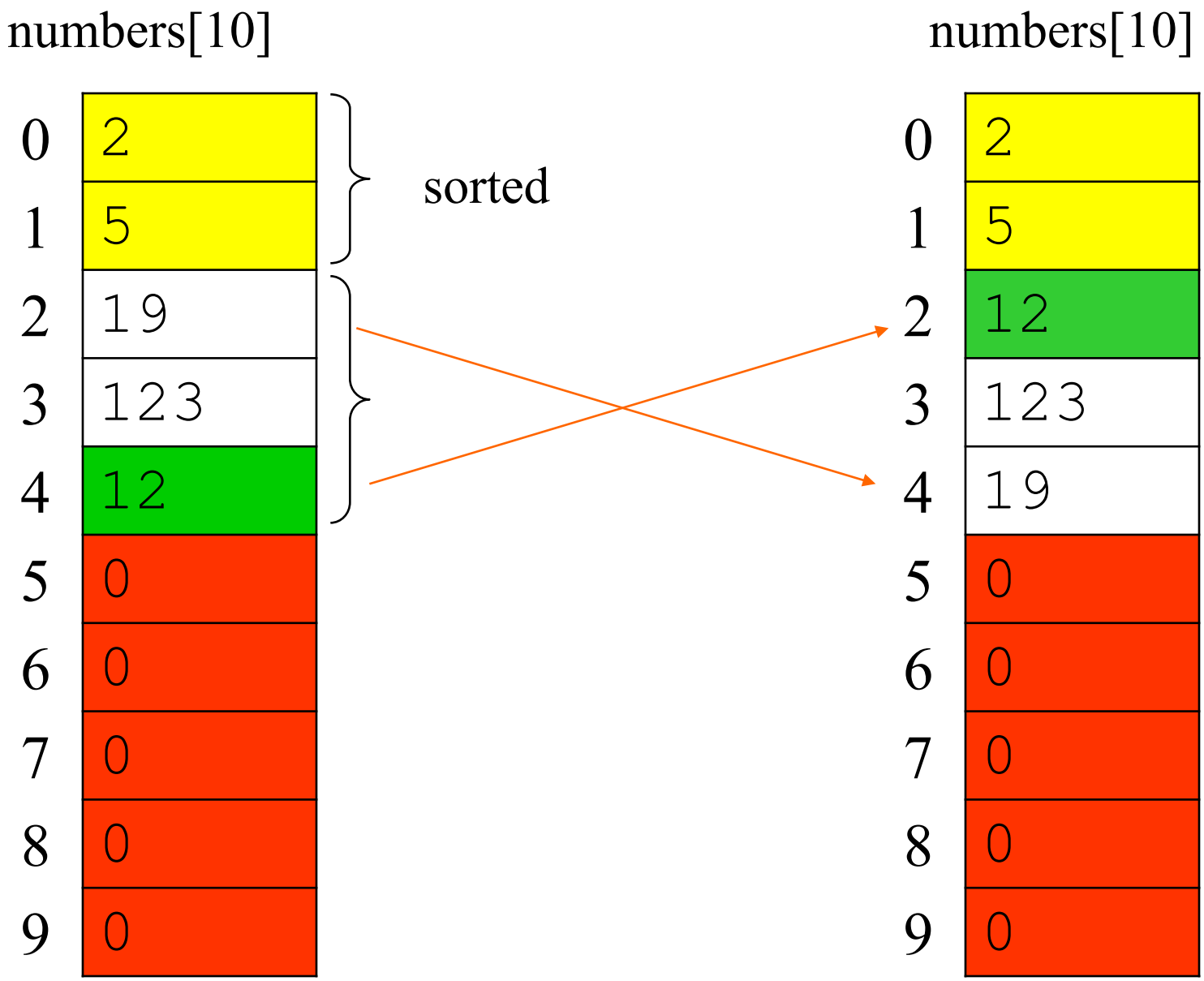
0	2
1	5
2	19
3	123
4	12
5	0
6	0
7	0
8	0
9	0

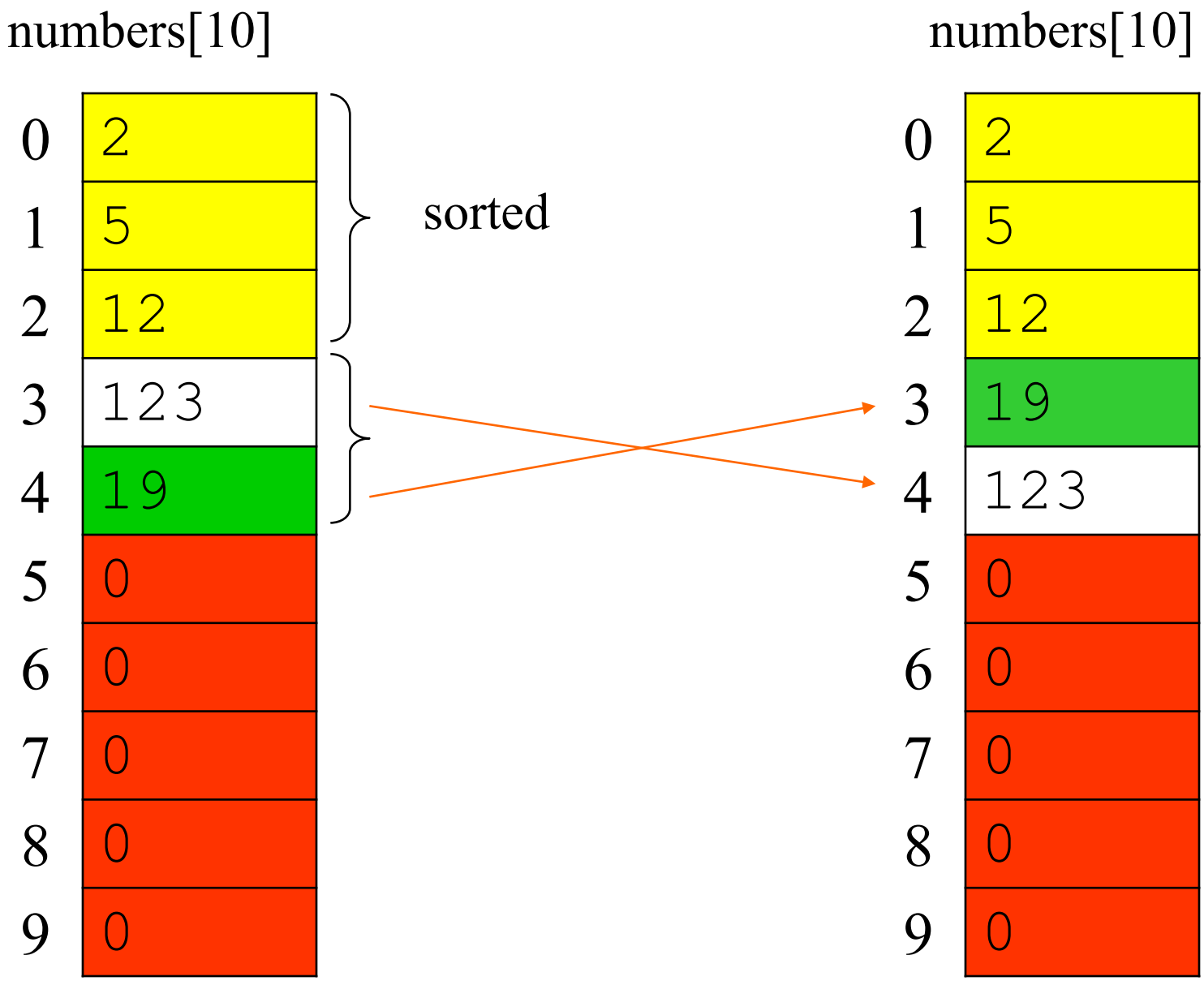
sorted

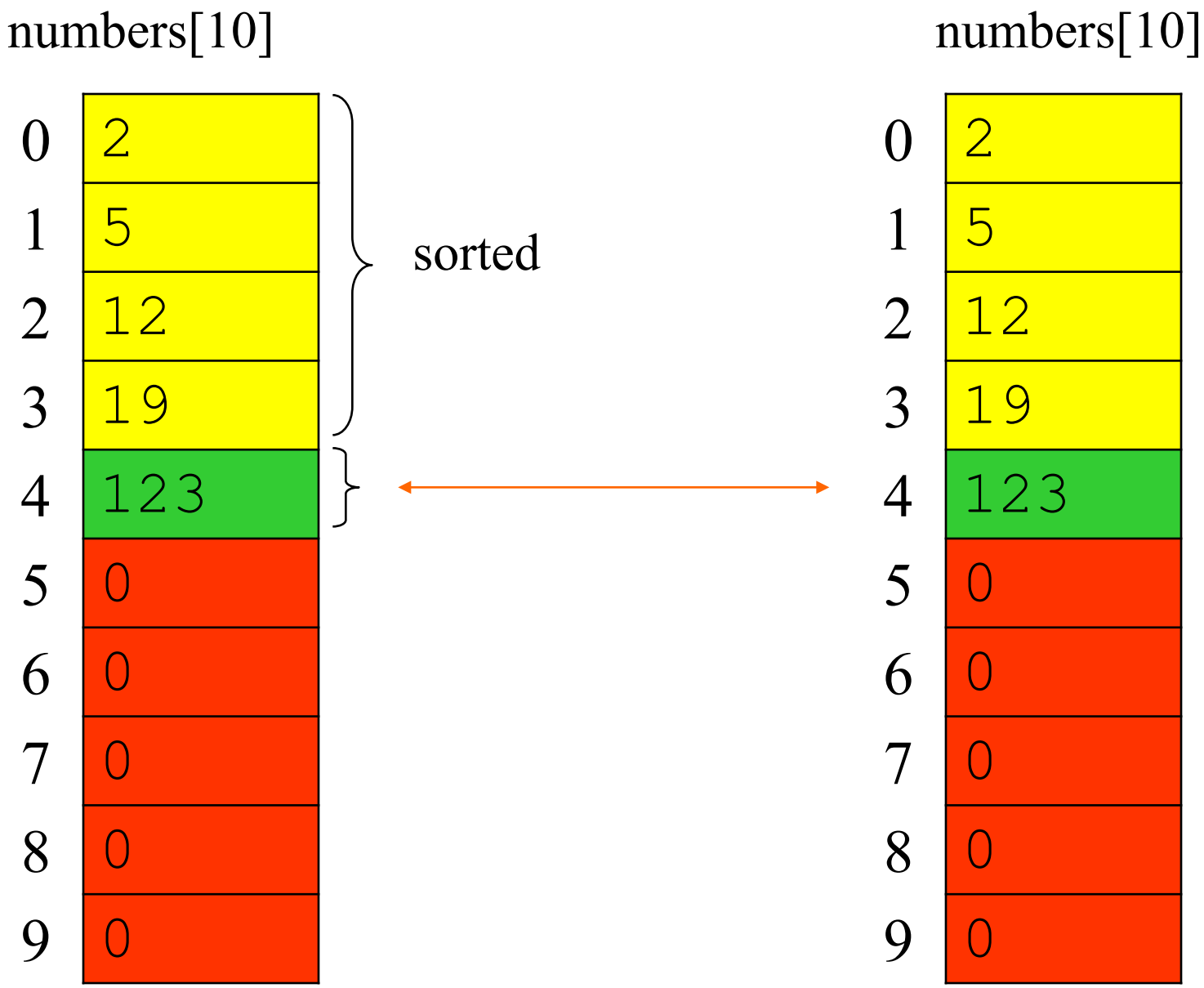
Not sorted;

Find the smallest number here
and swap it with numbers[1]

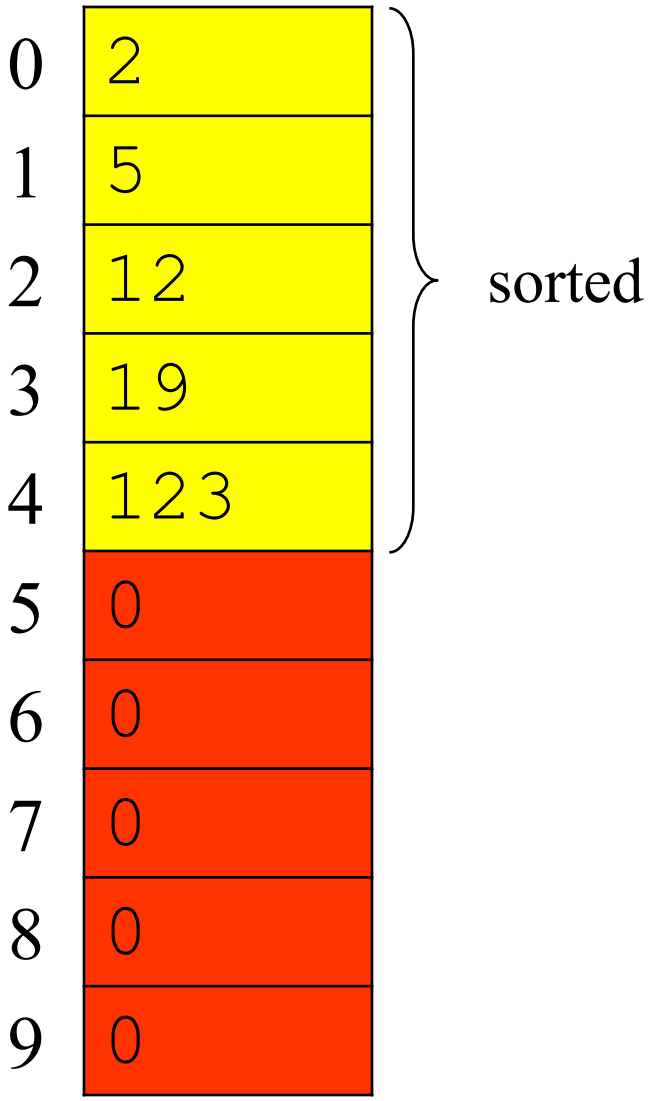

numbers[10]
numbers[10]
sorted
0 2
1 5
2 19
3 123
4 12
5 0
6 0
7 0
8 0
9 0
0 2
1 5
2 19
3 123
4 12
5 0
6 0
7 0
8 0
9 0


numbers[10]
0 2
1 5
2 19
3 123
4 12
5 0
6 0
7 0
8 0
9 0
sorted
numbers[10]
0 2
1 5
2 12
3 123
4 19
5 0
6 0
7 0
8 0
9 0


numbers[10]
0 2
1 5
2 12
3 123
4 19
5 0
6 0
7 0
8 0
9 0
sorted
numbers[10]
0 2
1 5
2 12
3 19
4 123
5 0
6 0
7 0
8 0
9 0


numbers[10]
0 2
1 5
2 12
3 19
4 123
5 0
6 0
7 0
8 0
9 0
sorted
numbers[10]
0 2
1 5
2 12
3 19
4 123
5 0
6 0
7 0
8 0
9 0

numbers[10]



```
private static void selectionSort
                                (short[] array, int length) {
    for ( int i = 0; i < length - 1; i++ ) {
        int indexLowest = i;
        for ( int j = i + 1; j < length; j++ )
            if ( array[j] < array[indexLowest] )
                indexLowest = j;
        if ( array[indexLowest] != array[i] ) {
            short temp = array[indexLowest];
            array[indexLowest] = array[i];
            array[i] = temp;
        }  // if
    } // for i
} // method selectionSort
```