

Lab 6 Analysis

To sort a list and place the first element in the correct position, placing it between the values smaller than it and those greater than it. The method `smallerBigger()` achieves this by walking through each element within given parameters for the starting position to an ending position. It singles out the first element in the list and compares it to all the other elements looking for those smaller than it and placing them in positions before it.

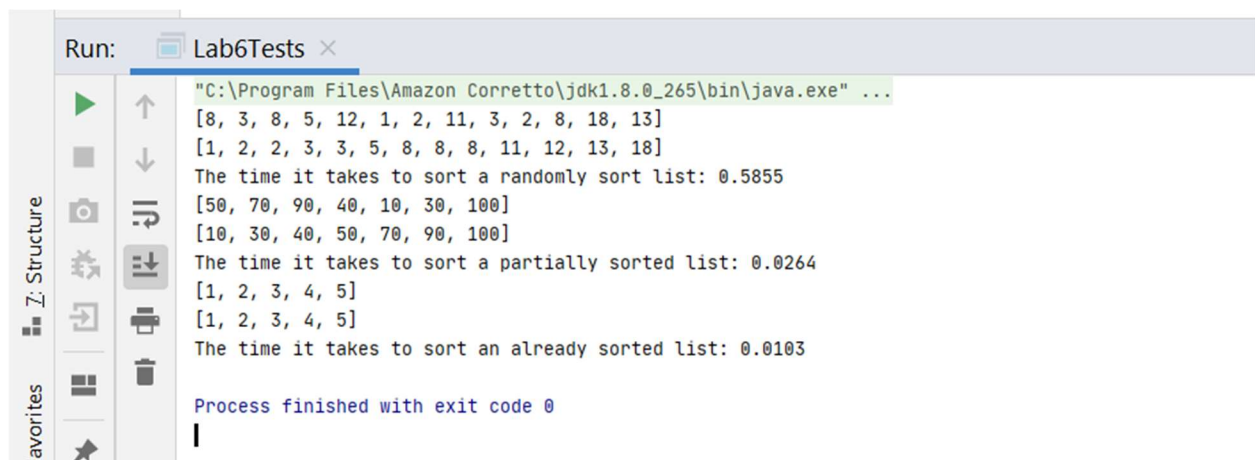
As well as moving elements greater than it to it right in the list. At the end of the method the elements less than the first element in the beginning of the list will be to the left of that element and the elements at the right of the first element will be those that are greater than it. The method was inspired from [GreekforGeeks](#) website cited in the comment section for the `smallerBigger()` method.

Results of benchmarking

The time it takes to sort a randomly sort list: 0.5855

The time it takes to sort a partially sorted list: 0.0264

The time it takes to sort an already sorted list: 0.0103

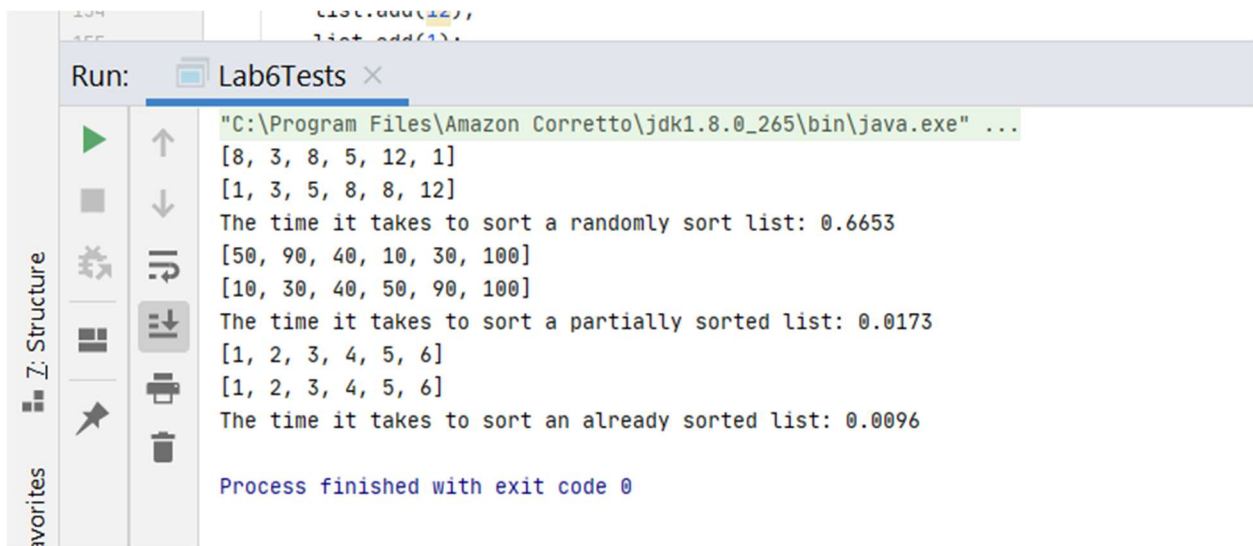


The screenshot shows an IDE's Run window for a project named 'Lab6Tests'. The output text is as follows:

```
Run: Lab6Tests x
"C:\Program Files\Amazon Corretto\jdk1.8.0_265\bin\java.exe" ...
[8, 3, 8, 5, 12, 1, 2, 11, 3, 2, 8, 18, 13]
[1, 2, 2, 3, 3, 5, 8, 8, 8, 11, 12, 13, 18]
The time it takes to sort a randomly sort list: 0.5855
[50, 70, 90, 40, 10, 30, 100]
[10, 30, 40, 50, 70, 90, 100]
The time it takes to sort a partially sorted list: 0.0264
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
The time it takes to sort an already sorted list: 0.0103

Process finished with exit code 0
|
```

The time it takes to sort the method changes with the starting condition on the list , The list that took the least amount of time was the list that was already sorted before the sorted method was called.



The screenshot shows an IDE's Run console window titled "Run: Lab6Tests". On the left, there is a vertical toolbar with icons for running (green play button), stopping (red square), debugging (bug icon), and other actions. The main area of the console displays the output of a Java program. The first line is the command: `"C:\Program Files\Amazon Corretto\jdk1.8.0_265\bin\java.exe" ...`. The output consists of several lines of text: `[8, 3, 8, 5, 12, 1]`, `[1, 3, 5, 8, 8, 12]`, `The time it takes to sort a randomly sort list: 0.6653`, `[50, 90, 40, 10, 30, 100]`, `[10, 30, 40, 50, 90, 100]`, `The time it takes to sort a partially sorted list: 0.0173`, `[1, 2, 3, 4, 5, 6]`, `[1, 2, 3, 4, 5, 6]`, and `The time it takes to sort an already sorted list: 0.0096`. The final line is `Process finished with exit code 0`.

```
"C:\Program Files\Amazon Corretto\jdk1.8.0_265\bin\java.exe" ...  
[8, 3, 8, 5, 12, 1]  
[1, 3, 5, 8, 8, 12]  
The time it takes to sort a randomly sort list: 0.6653  
[50, 90, 40, 10, 30, 100]  
[10, 30, 40, 50, 90, 100]  
The time it takes to sort a partially sorted list: 0.0173  
[1, 2, 3, 4, 5, 6]  
[1, 2, 3, 4, 5, 6]  
The time it takes to sort an already sorted list: 0.0096  
  
Process finished with exit code 0
```

Even when the lists are of the same time this remains true that the starting condition of the list changes how long it will take to sort it.