HackerRank | Prepare Certify Compete Apply Search

All Contests > DAA_LAB > Quick-sort

Quick-sort

Problem Submissions Leaderboard Discussions

Sort a given set of n integer elements using Quick Sort method and compute its time complexity. Run the program for varied values of n> 5000 and record the time taken to sort. Plot a graph of the time taken versus non graph sheet.The elements can be read from a file or can be generated using the random number generator. Demonstrate using Java how the divide - and - conquer method works along with its time complexity analysis: worst case, average case and best case.

Input Format

5 0 0 4 3 1

Constraints

Size of the array should be always positive

Output Format

Before Sort: 0 0 4 3 1 After sort: 0 0 1 3 4

Sample Input 0

```
5
0
0
4
3
1
```

Sample Output 0

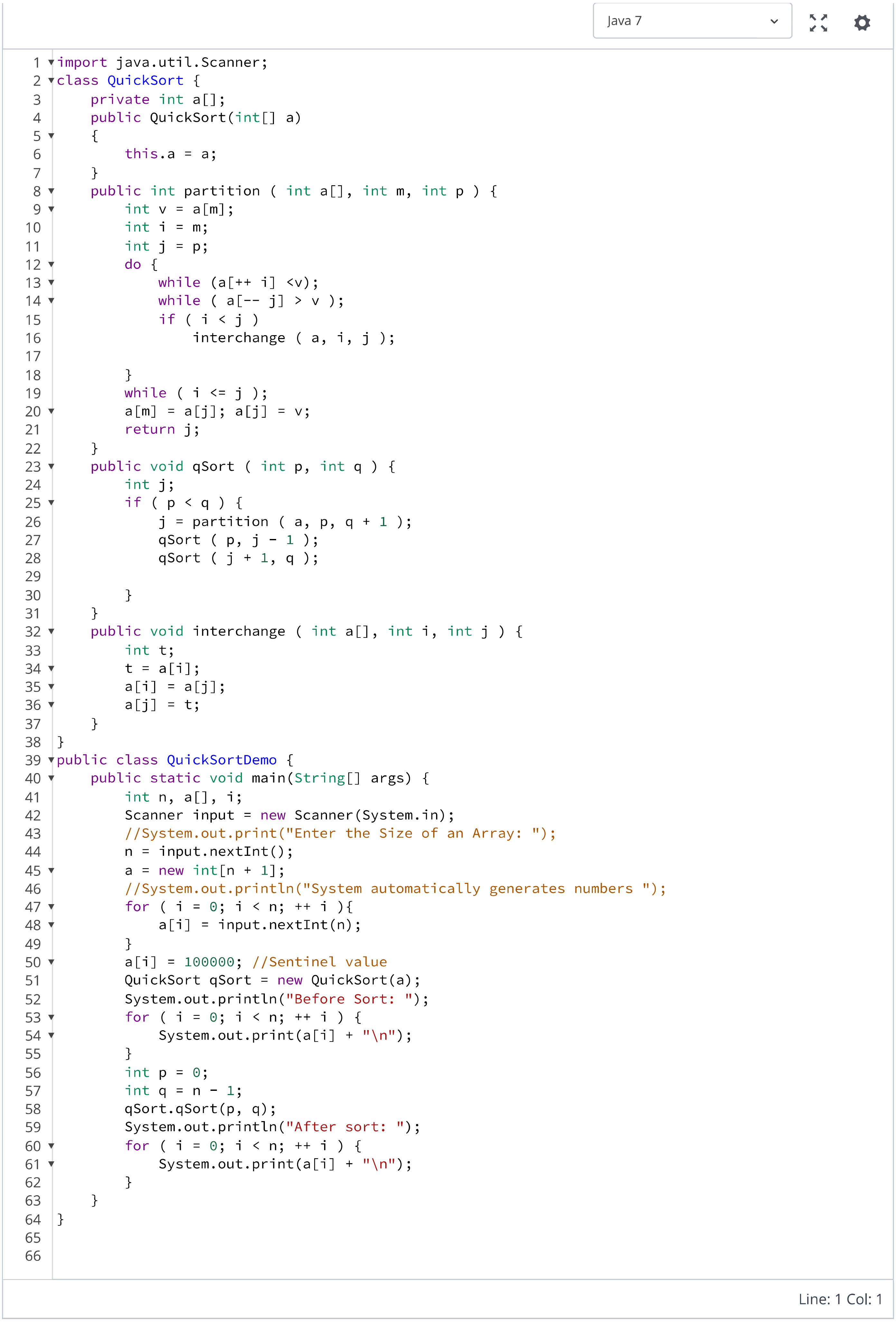
```
Before Sort:
0
0
4
3
1
After sort:
0
0
1
3
4
```

Contest ends in 9 days

Submissions: 103
Max Score: 10
Difficulty: Medium

Rate This Challenge:
☆☆☆☆☆

More



```java
import java.util.Scanner;
class QuickSort {
    private int a[];
    public QuickSort(int[] a)
    {
        this.a = a;
    }
    public int partition ( int a[], int m, int p ) {
        int v = a[m];
        int i = m;
        int j = p;
        do {
            while (a[++ i] <v);
            while ( a[-- j] > v );
            if ( i < j )
                interchange ( a, i, j );

        }
        while ( i <= j );
        a[m] = a[j]; a[j] = v;
        return j;
    }
    public void qSort ( int p, int q ) {
        int j;
        if ( p < q ) {
            j = partition ( a, p, q + 1 );
            qSort ( p, j - 1 );
            qSort ( j + 1, q );

        }
    }
    public void interchange ( int a[], int i, int j ) {
        int t;
        t = a[i];
        a[i] = a[j];
        a[j] = t;
    }
}
public class QuickSortDemo {
    public static void main(String[] args) {
        int n, a[], i;
        Scanner input = new Scanner(System.in);
        //System.out.print("Enter the Size of an Array: ");
        n = input.nextInt();
        a = new int[n + 1];
        //System.out.println("System automatically generates numbers ");
        for ( i = 0; i < n; ++ i ){
            a[i] = input.nextInt(n);
        }
        a[i] = 100000; //Sentinel value
        QuickSort qSort = new QuickSort(a);
        System.out.println("Before Sort: ");
        for ( i = 0; i < n; ++ i ) {
            System.out.print(a[i] + "\n");
        }
        int p = 0;
        int q = n - 1;
        qSort.qSort(p, q);
        System.out.println("After sort: ");
        for ( i = 0; i < n; ++ i ) {
            System.out.print(a[i] + "\n");
        }
    }
}


```

Upload Code as File □ Test against custom input Run Code Submit Code

Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
5
0
0
4
3
1
```

Your Output (stdout)

```
Before Sort:
0
0
4
3
1
After sort:
0
0
1
3
4
```

Expected Output

```
Before Sort:
0
0
4
3
1
After sort:
0
0
1
3
4
```

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |