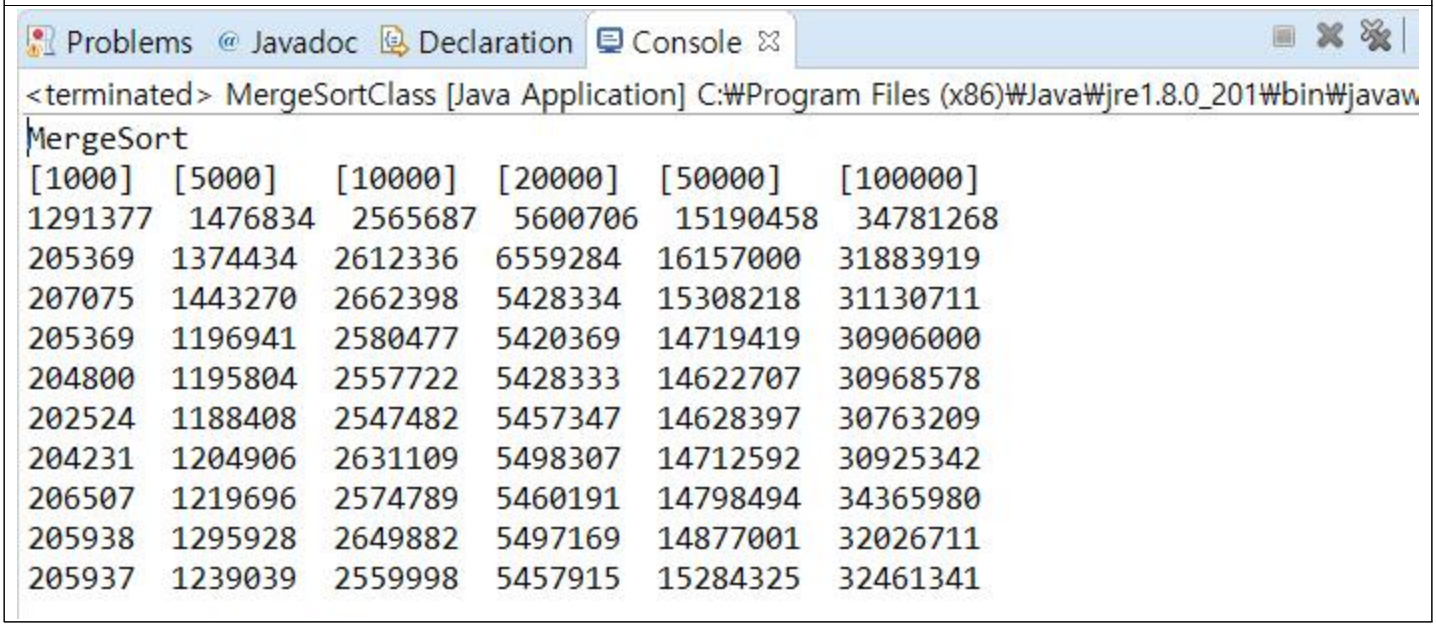
과목명	알고리즘
담당교수	우진운 교수님
학과	소프트웨어학과
학번	32153180
이름	이상민
제출일자	2019.04.15

MergeSort code

```
package homework;
import java.util.Scanner;

public class MergeSortClass {
        private int a[], b[] ;
    private int aSize ;

    public MergeSortClass(int arr[], int n) {
        a = arr;
        aSize = n;
        b = new int[aSize+1];
    }

    public int[] MergeSortCall() {
        MergeSort(1, aSize);
        return a ;
    }

    public void MergeSort(int low, int high) {
        if (low < high) {
                int mid = (low + high)/2;
            MergeSort(low, mid);
            MergeSort(mid + 1, high);
            Merge(low, mid, high);
        }
    }
    public void Merge(int low, int mid, int high) {
        int h = low, i = low, j = mid+1, k;
        while ((h <= mid) && (j <= high)) {
            if (a[h] <= a[j]) {
                    b[i] = a[h];
                    h++; }
            else {
                    b[i] = a[j];
                    j++;
                    }
            i++;
        }
        if (h > mid)
                for (k=j; k<=high; k++) {
                        b[i] = a[k];
                        i++;
                        }
        else
                for (k=h; k<=mid; k++) {
                b[i] = a[k];
                i++;
             }
        for (k=low; k<=high; k++)
                a[k] = b[k];
    }

    public static void main(String args[]) {
                int size[] = { 1000, 5000, 10000, 20000, 50000, 100000 };
                int store[][] = new int[6][];
// 정렬 후 저장할 배열

                for (int i = 0; i < 6; i++)
                        store[i] = new int[size[i] + 10];                                       // 배열에
대한 사이즈

                System.out.println("MergeSort");
                System.out.println("[1000]   [5000]    [10000]   [20000]   [50000]    [100000]");
                for (int i = 0; i < 10; i++) {
// 데이터 10개 테스트
                        for (int j = 0; j < 6; j++)
        // 각각의 크기만큼 6번 수행
                                for (int k = 0; k < size[j]; k++)                               //
할당받은 크기만큼 반복
                                        store[j][k] = (int)(java.lang.Math.random() * size[j]);
```

```
                                // 난수 입력

                        for (int l = 0; l < 6; l++) {
                                MergeSortClass merge = new MergeSortClass(store[l], size[l]);
                                long before = System.nanoTime();
// 합병 정렬 시작 시간
                                store[l] = merge.MergeSortCall();
// 합병 정렬
                                long after = System.nanoTime();
// 합병 정렬 끝난 시간
                                long execute = after - before;
// 합병 정렬 걸린 시간 구하기
                                System.out.print(execute + "   ");
                        }
                        System.out.println("");
                }
                System.out.println();
        }
}
```

실행결과

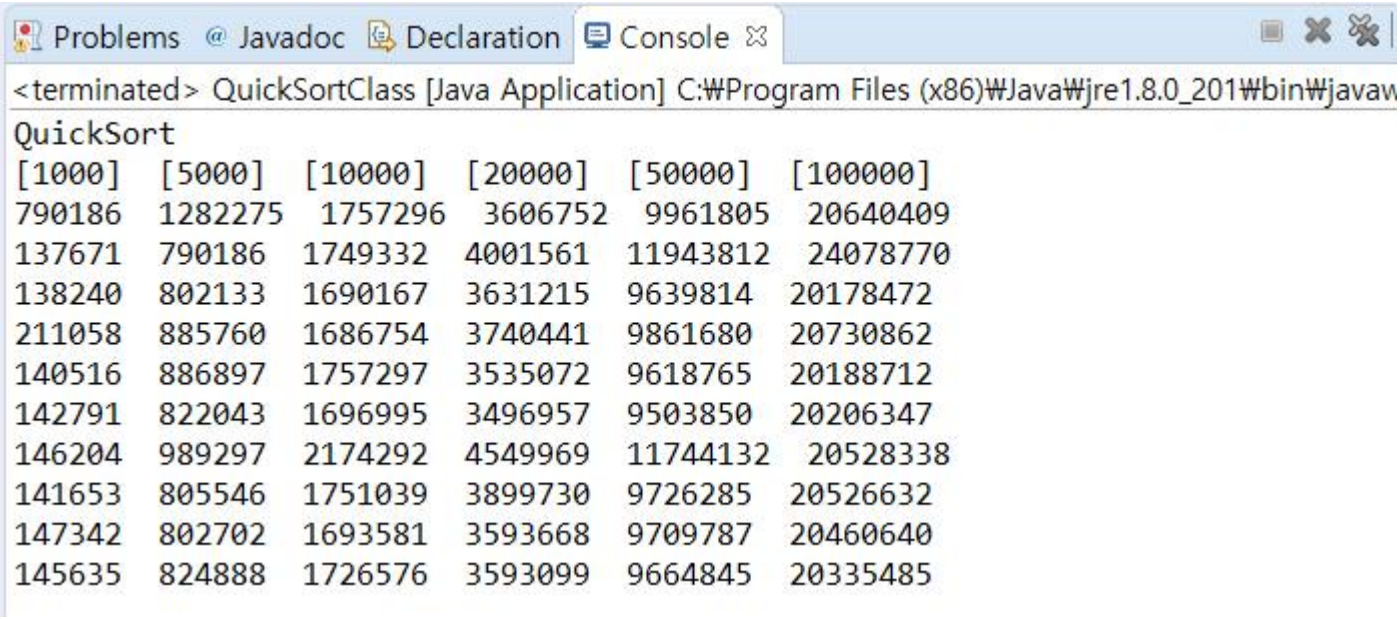


QuickSort code

```
package homework;
import java.util.Scanner;

public class QuickSortClass {
        private int a[];
    private int aSize ;

    public QuickSortClass(int arr[], int n) {
        a = arr;
        aSize = n;
                a[n+1] = Integer.MAX_VALUE ;
     }

    public int[] QuickSortCall() {
        QuickSort(1, aSize);
        return a ;
    }

    void QuickSort(int p, int q) {
                if (p < q) {
                      int j = Partition(a, p, q+1);
                      QuickSort(p, j-1);
                      QuickSort(j+1,q);
                }
            }

    int Partition(int a[], int m, int p) {
                int v=a[m];
                int i=m, j=p;
                do {
                        do i++;
                        while (a[i] < v);
                        do j--;
                        while (a[j] > v);
                        if (i < j)
                            Interchange(a, i, j);
                    } while (i < j);
                    a[m] = a[j];
                    a[j] = v;
                    return(j);
            }

    void Interchange(int a[], int i, int j) {
                    int temp = a[i];
                    a[i] = a[j];
                    a[j] = temp;
          }

    public static void main(String args[]) {
                int size[] = { 1000, 5000, 10000, 20000, 50000, 100000 };
                int store[][] = new int[6][];
// 정렬 후 저장할 배열

                for (int i = 0; i < 6; i++)
                        store[i] = new int[size[i] + 10];                                    // 배열에
대한 사이즈

                System.out.println("QuickSort");
                System.out.println("[1000]  [5000]  [10000]  [20000]  [50000]  [100000]");
                for (int i = 0; i < 10; i++) {
// 데이터 10개 테스트
                        for (int j = 0; j < 6; j++)
        // 각각의 크기만큼 6번 수행
                                for (int k = 0; k < size[j]; k++)                            //
할당받은 크기만큼 반복
                                        store[j][k] = (int)(java.lang.Math.random() * size[j]);

                                // 난수 입력

                        for (int l = 0; l < 6; l++) {
                                QuickSortClass quick = new QuickSortClass(store[l], size[l]);
                                long before = System.nanoTime();
// 퀵 정렬 시작 시간
                                store[l] = quick.QuickSortCall();                            // 퀵
정렬
```

```
                                long after = System.nanoTime();
// 퀵 정렬 끝난 시간
                                long execute = after - before;
// 퀵 정렬 걸린 시간 구하기
                                System.out.print(execute + "  ");
                        }
                        System.out.println("");
                }
                System.out.println();
        }
}
```

실행결과



<합병정렬>

합병	1	2	3	4	5	6	7	8	9	10
1000	1291377	205369	207075	205369	204800	202524	204231	206507	205938	205937
5000	1476834	1374434	1443270	1196941	1195804	1188408	1204906	1219696	1295928	1239039
10000	2565687	2612336	2662398	2580477	2557722	2547482	2631109	2574789	2649882	2559998
20000	5600706	6559284	5428334	5420369	5428333	5457347	5498307	5460191	5497169	5457915
50000	15190458	16157000	15308218	14719419	14622707	14628397	14712592	14798494	14877001	15284325
100000	34781268	31883919	31130711	30906000	30968578	30763209	30925342	34365980	32026711	32461341

<퀵정렬>

퀵	1	2	3	4	5	6	7	8	9	10
1000	790186	137671	138240	211058	140516	142791	146204	141653	147342	145635
5000	1282275	790186	802133	885760	886897	822043	989297	805546	802702	824888
10000	1757296	1749332	1690167	1686754	1757297	1696995	2174292	1751039	1693581	1726576
20000	3606752	4001561	3631215	3740441	3535072	3496957	4549969	3899730	3593668	3593099
50000	9961805	11943812	9639814	9861680	9618765	9503850	11744132	9726285	9709797	9664845
100000	20640409	24078770	20178472	20730862	20188712	20206347	20528338	20526632	20460640	20335485

<평균>

	1000	5000	10000	20000	50000	100000
합병정렬	313912.70	1283526.00	2594188.00	5580795.50	15029861.10	32021305.90
퀵정렬	214129.60	889172.70	1768332.90	3764846.40	10137478.50	20787466.70