Data Structures and Algorithms Laboratory		
Laboratory 11: Basic Search and Sorting	**School of Information Technology**	
Name:Sai Hae Naing Lay	**ID:6531501208**	**Section:4**
Date: 4 May 2023	**Due date: on LMS**	

Objective

- To perform binary search
- To perform basic sorting i.e. bubble sort, selection sort and insertion sort
- To implement sorting based on divide and conquer i.e. merge sort and quick sort
- To estimate and compare efficiency of sorting algorithms

Exercise 1: **(Homework)** Given a sorted integer array, complete the binary-search codes below to get the results as shown. Note that you must NOT use the Java built-in method!

Expected result:

```
Found 22 at position 10
```

Codes:

```
public class BinarySearch

{

public static void main(String[] args)

{

int[] data = { 2, 4, 5, 7, 8, 9, 12, 14, 17, 19, 22, 25, 27, 28, 33,
37 };

//searching value

int value = 22;

//searching

int position = search(data, value);

System.out.println("Found " + value + " at position " + position);

}

public static int search(int[] data, int value)

{
```

```
int low = 0;

int high = data.length -1;

while(low <= high)

{

int mid = (low + high)/2;

if(data[mid] == value)

{

return mid;

}

else if(data[mid] < value)

{

low = mid + 1;

}

else

{

high = mid -1;

}

}

return -1;

}

}
```

Exercise 2: (Homework) Given an unsorted array below, implement a **bubble sort** to get the following result.

Expected result:

```
Bubble Sort
[77, 44, 99, 66, 33, 55, 88, 22]
```

```
[22, 33, 44, 55, 66, 77, 88, 99]
```

Codes:

```
public class BubbleSorting

{

public static void main(String[] args)

{

int[] a = {77,44,99,66,33,55,88,22};

System.out.println("Bubble Sort");

print(a);

bubbleSort(a);

print(a);

}

//print method

private static void print(int[] a)

{

int b = a.length;

for(int i = 0; i < b; i++)

{

System.out.print(a[i] + " ");

}

System.out.println();

}

//swap() method

private static void swap(int[] a, int i, int j)
{

int temp = a[i];

a[i] = a[j];

a[j] = temp;

}

//bubbleSort() method

public static void bubbleSort(int[] a)

{

int n = a.length;

for(int i = 0; i < n; i++)

{

for(int j = 0; j < n-i-1; j++)

{

if(a[j] > a[j+1])

{

swap(a,j,j+1);

}

}

}

}

}
```

Exercise 3 (Homework): Fill in the missing codes for a **selection sort**.

Expected result:

```
Selection Sort
[77, 44, 99, 66, 33, 55, 88, 22]
[22, 33, 44, 55, 66, 77, 88, 99]
```

Codes:

```java
public class SelectionSorting

{

public static void main(String[] args)

{

int[] data = {77,44,99,66,33,55,88,22};

System.out.println("Selection Sort");

print(data);

selectionSort(data);

print(data);

}

//print() method

private static void print(int[] a)

{

System.out.print("[");

for(int i = 0; i < a.length; i++)

{

System.out.print(a[i] + ",");

}

System.out.print("]");

System.out.println();

}
//swap() method

private static void swap(int[] a, int i, int j)

{

int temp = a[i];

a[i] = a[j];

a[j] = temp;

}

public static void selectionSort(int[] data)

{

int n = data.length;

for(int i = 0; i < n-1; i++)

{

int minIndex = i;

for(int j = i + 1; j < n; j++)

{

if(data[j] < data[minIndex])

{

minIndex = j;

}

}

swap(data,i,minIndex);

}

}

}
```

<u>Exercise 4 (Homework):</u> Implement a **quick sort**.

Expected result:

```
Quick Sort
[77, 44, 99, 66, 33, 55, 88, 22]
[22, 33, 44, 55, 66, 77, 88, 99]
```

Codes:

```java
public class QuickSort

{

public static void main(String[] args)

{

int[] data = {77,44,99,66,33,55,88,22};

System.out.println("Quick Sort");

print(data);

qSort(data, 0, data.length- 1);

print(data);

}

//print() method

private static void print(int[] a)

{

System.out.print("[");

for(int i = 0; i < a.length; i++)

{

System.out.print(a[i] + ",");

}

System.out.print("]");

System.out.println();
}

private static void qSort(int[] a, int low, int high)

{

if(low < high)

{

int pivotIndex = partition(a, low, high);

qSort(a, low, pivotIndex - 1);

qSort(a, pivotIndex + 1, high);

}

}

private static int partition(int[] a, int low, int high)

{

int pivotValue = a[high];

int pivotIndex = low;

for(int i = low; i < high; i++)

{

if(a[i] < pivotValue)

{

swap(a, i, pivotIndex);

pivotIndex++;

}

}

swap(a, pivotIndex, high);

return pivotIndex;

}
```

```
public static void swap(int[] a, int i, int j)

{

int temp = a[i];

a[i] = a[j];

a[j] = temp;

}

}
```

<u>Exercise 5 (Homework):</u> Implement an **insertion sort**.

Expected result:

```
Insertion Sort
[77, 44, 99, 66, 33, 55, 88, 22]
[22, 33, 44, 55, 66, 77, 88, 99]
```

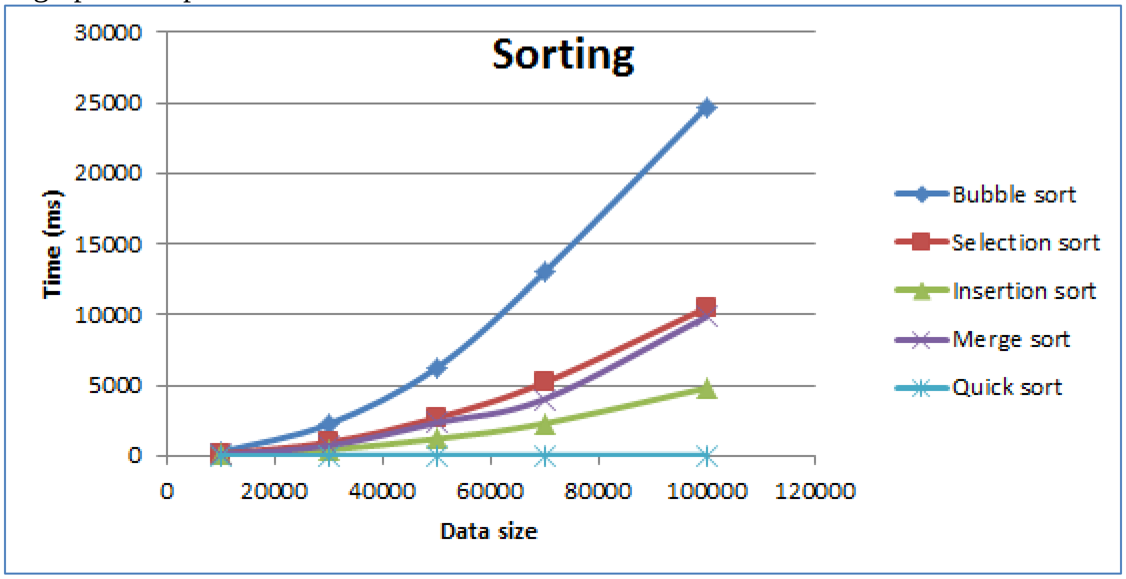
Codes:

```
public class InsertionSorting

{

public static void main(String[] args)

{

int[] a = {77,44,99,66,33,55,88,22};

System.out.println("Insertion Sort");

print(a);

insertionSort(a);

print(a);

}

//print() method

private static void print(int[] a)

{

System.out.print("[");

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

System.out.print(a[i] + ",");

}

System.out.print("]");

System.out.println();

}

//swap() method
private static void swap(int[] a, int i, int j)

{

int temp = a[i];

a[i] = a[j];

a[j] = temp;

}

//bubbleSort() method

public static void insertionSort(int[] data)

{

for(int i = 1; i < data.length; i++)

{

int current = data[i];

int j = i - 1;

while(j >= 0 && data[j] > current)

{

data[j+1] = data[j];

j--;

}

data[j+1] = current;

}

}

}
```

Exercise 6 (Homework): Complete all sorting functions in the previous exercises. Then, change/replace the source code by using the given ***print()*** and ***main()*** methods below. After that try to import the java library (`import java.util.Random;`). Then, let change the number of MAX variable to 10000, 30000, 50000, 70000 and 100000 and fill the time used for each sorting technique in the table below. Also use this table to plot a graph.

Algorithm	Sorting Time (ms)				
Data size	10000	30000	50000	70000	100000
Bubble sort	**84**	**1063**	**3021**	**5939**	**12309**
Selection sort	**28**	**191**	**490**	**1138**	**2229**
Insertion sort	**34**	**292**	**666**	**1373**	**2693**
Merge sort	**2**	**10**	**12**	**17**	**25**
Quick sort	**3**	**5**	**8**	**9**	**18**

A graph example of all sorts via Microsoft Excel



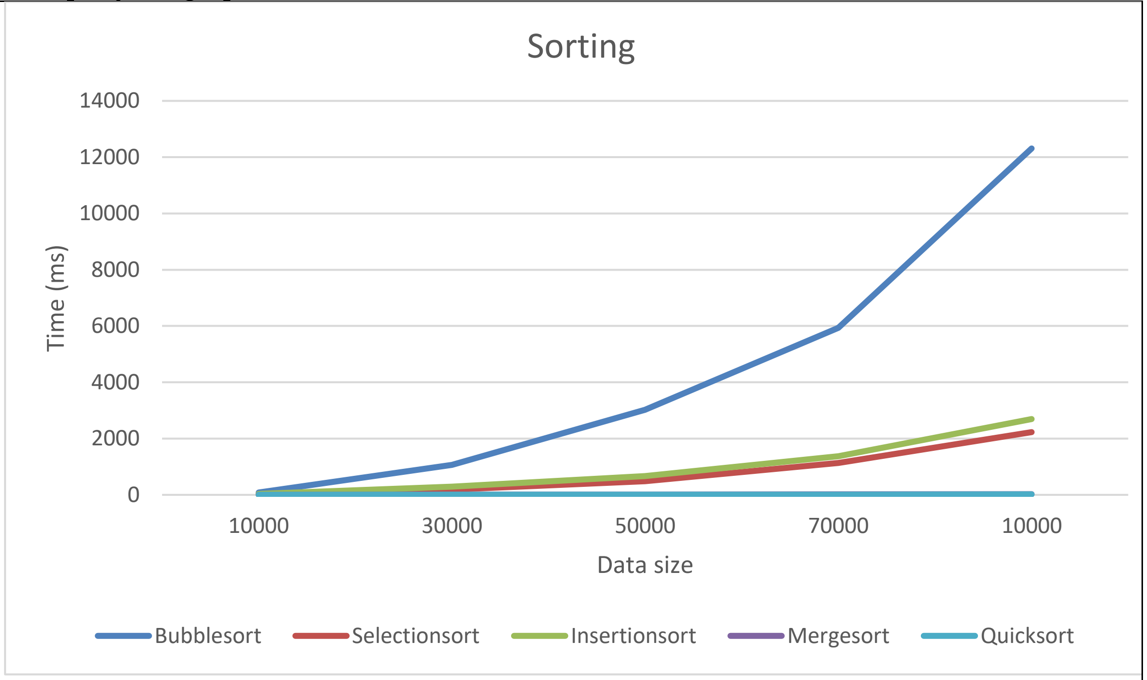
Note that these sorting times will be varied according to a computer. So you result must be different from the example. **You should NOT run all algorithms at the same time**. Running each algorithm at a time is recommended.

Given methods to replace the old one.

```
        //new print() method
          private static void print(int[] a){
                System.out.println("length= "+a.length);
                for (int i=0;i<a.length;i++){
             System.out.print(" "+a[i]);
           }
           System.out.println();
          }

          //new main() method
          public static void main(String[] args){
                //int[] data = {77,44,99,66,33,55,88,22};
               int MAX=10000;
```

```
Random rand = new Random();
int[] data = new int[MAX];
for(int i=0;i<MAX;i++) {
      data[i]=rand.nextInt(MAX)+1;
}
  System.out.println("Bubble Sort");
  print(data);

  //Algorithm Running Time (ms)
  long startTime = System.currentTimeMillis();
  bubbleSort(data);
  long endTime = System.currentTimeMillis();
  //

  print(data);
  System.out.println("Elapsed time = "+(endTime-startTime));
}
```

Paste/put your graph here.


Sorting
Time (ms)
Data size
Bubblesort
Selectionsort
Insertionsort
Mergesort
Quicksort