**Name : Ali Hassaan Mughal Reg: 173627**

**Advanced Programming Lab 3:**

**Code:**

**MainClass.java**

**public** **class** MainClass {

**public** **static** **void** main(String[] args) {

String[] elements = {"beta", "chi", "alpha","zeta","nu","mu" };

SortFactory factory = **new** NonRecursive();

Integer[] elem = {9,8,7,6,5,4,3,2,1};

factory.setFactory("Selection");

SortingAlgorithm algorithm = factory.getAlgorithm();

algorithm.sort(elements);

algorithm.sort(elem);

System.***out***.println("The Sorted strings are: ");

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

{

System.***out***.println(elements[i]);

}

System.***out***.println("The Sorted Integers are: ");

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

{

System.***out***.println(elem[i]);

}

}

}

**SortFactory.java**

/\*\*

\* A factory for sorting algorithms.

\*/

**public** **abstract** **class** SortFactory {

**protected** String algorithm;

/\*\*

\* The type of sorting algorithm we want to generate

\*/

**public** **void** setFactory(String algorithm) {

**this**.algorithm = algorithm;

}

/\*\*

\* This must generate the appropriate sorting algorithm

\* according to what was set in setFactory().

\*/

**public** **abstract** SortingAlgorithm getAlgorithm();

}

**NonRecursive.java**

**public** **class** NonRecursive **extends** SortFactory {

**public** SortingAlgorithm getAlgorithm() {

**if** (**this**.algorithm.equals("Selection")) {

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

**return** **new** SelectionSort();

}

**else** **if** (**this**.algorithm.equals("Insertion")) {

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

**return** **new** InsertionSort();

}

**return** **null**;

}

}

**SortingAlgorithm.java**

/\*\*

\* An interface for a sorting algorithm.

\*/

**public** **interface** SortingAlgorithm {

**public** **abstract** **void** sort(Object[] items);

}

**SelectionSort.java**

/\*\*

\* An implementation of the selection sort algorithm.

\*/

**public** **class** SelectionSort **implements** SortingAlgorithm

{

**public** **void** sort(Object[] items) {

**int** i, j, minPosition;

**for** (i = 0; i < items.length - 1; i++) {

minPosition = i;

**for** (j = i + 1; j < items.length; j++) {

**if**( ((Comparable) items[j]).compareTo(items[minPosition]) < 0)

minPosition = j;

}

// exchange items[i] and items[minPosition]

Object t = items[i];

items[i] = items[minPosition];

items[minPosition] = t;

}

}

}

**InsertionSort.java**

**public** **class** InsertionSort **implements** SortingAlgorithm {

**public** **void** sort(Object[] items) {

**int** j;

**for** (**int** p=1;p<items.length;p++) {

Object temp = items[p];

**for** (j=p; j > 0 &&((Comparable) items[j-1]).compareTo(temp) > 0; j--)

items[j] = items[j-1];

items[j] = temp;

}

}

}

**Output:**

