```java
class Employee implements Comparable<Employee> {

int id;
String name;

public Employee(int eid, String ename) {
id = eid;
name = ename;
}

@Override
public int compareTo(Employee emp) {
return name.compareTo(emp.name);
}

@Override
public String toString() {
return "Employee [name=" + name + ", id=" + id + "]";
}
}
```

```java
Employee emp1=new Employee(4, "A");
Employee emp2=new Employee(2, "C");
Employee emp3=new Employee(1, "B");
Employee emp4=new Employee(3, "D");

ArrayList<Employee> list=new ArrayList<Employee>();
list.add(emp1);
list.add(emp2);
list.add(emp3);
list.add(emp4);
System.out.println("Insertion Order in List: \n"+list);

Collections.sort(list);
System.out.println("After Sorting Names in Sorting Order : \n"+list);

Collections.reverse(list);
System.out.println("After Sorting Names in Reverse Order : \n"+list);

}
```

Insertion Order in List:
[Employee [name=A, id=4], Employee [name=C, id=2], Employee [name=B, id=1], Employee [name=D, id=3]]
After Sorting Names in Sorting Order :
[Employee [name=A, id=4], Employee [name=B, id=1], Employee [name=C, id=2], Employee [name=D, id=3]]
After Sorting Names in Reverse Order :
[Employee [name=D, id=3], Employee [name=C, id=2], Employee [name=B, id=1], Employee [name=A, id=4]]

```
class Employee implements Comparable<Employee> {

int id;
String name;

public Employee(String ename, int eid) {
name = ename;
id = eid;
}

@Override
public int compareTo(Employee emp) {
return emp.name.compareTo(name);
}

@Override
public String toString() {
return "Employee [id=" + id + ", name=" + name + "]";
}
}
```

```
Employee emp1 = new Employee("A", 4);
Employee emp2 = new Employee("C", 2);
Employee emp3 = new Employee("B", 1);
Employee emp4 = new Employee("D", 3);

ArrayList<Employee> list = new ArrayList<Employee>();
list.add(emp1);
list.add(emp2);
list.add(emp3);
list.add(emp4);
System.out.println("Insertion Order in List: \n" + list);

Collections.sort(list);
System.out.println("After Sorting Names in Desending  Order :  \n" +
list);

Collections.reverse(list);
System.out.println("After Sorting Names in Reverse Order :  \n" + list);
```

Insertion Order in List:
[Employee [name=A, id=4], Employee [name=C, id=2], Employee [name=B, id=1], Employee [name=D, id=3]]
After Sorting Names in Desending Order :
[Employee [name=D, id=3], Employee [name=C, id=2], Employee [name=B, id=1], Employee [name=A, id=4]]
After Sorting Names in Reverse Order :
[Employee [name=A, id=4], Employee [name=B, id=1], Employee [name=C, id=2], Employee [name=D, id=3]]

```java
class Employee implements Comparator<Employee> {
int id;
String name;

public Employee(int id, String name) {
this.id = id;
this.name = name;
}
public Employee() {
}
@Override
public String toString() {
return "Employee [id=" + id + ", name=" + name + "]";
}
@Override
public int compare(Employee o1, Employee o2) {
return o1.name.compareTo(o2.name); //ascending
}
}
```

```java
Employee emp1 = new Employee(4, "D");
Employee emp3 = new Employee(1, "A");
Employee emp2 = new Employee(2, "B");
Employee emp4 = new Employee(3, "C");

ArrayList<Employee> list = new ArrayList<Employee>();
list.add(emp1);
list.add(emp2);
list.add(emp3);
list.add(emp4);
System.out.println("Insertion  Order: \n" + list);

Collections.sort(list, new Employee());
System.out.println("Sorting Names in Asending Order : \n" + list);

Collections.reverse(list);
System.out.println("Sorting Names in Reverse Order : \n" + list);
```

Insertion Order:
[Employee [id=4, name=D], Employee [id=2, name=B], Employee [id=1, name=A], Employee [id=3, name=C]]
Sorting Names in Asending Order :
[Employee [id=1, name=A], Employee [id=2, name=B], Employee [id=3, name=C], Employee [id=4, name=D]]
Sorting Names in Reverse Order :
[Employee [id=4, name=D], Employee [id=3, name=C], Employee [id=2, name=B], Employee [id=1, name=A]]

```java
class Employee implements Comparator<Employee> {

int id;
String name;

public Employee(int id, String name) {
this.id = id;
this.name = name;
}
public Employee() {
}
@Override
public String toString() {
return "Employee [id=" + id + ", name=" + name + "]";
}
@Override
public int compare(Employee o1, Employee o2) {
return o2.name.compareTo(o1.name); //descending
}}
```

```java
Employee emp1 = new Employee(4, "D");
Employee emp3 = new Employee(1, "A");
Employee emp2 = new Employee(2, "B");
Employee emp4 = new Employee(3, "C");

ArrayList<Employee> list = new ArrayList<Employee>();
list.add(emp1);
list.add(emp2);
list.add(emp3);
list.add(emp4);
System.out.println("Insertion  Order: \n" + list);

Collections.sort(list, new Employee());
System.out.println("Sorting Names in Desending Order : \n" + list);

Collections.reverse(list);
System.out.println("Sorting Names in Reverse Order : \n" + list);
```

Insertion Order:
[Employee [id=4, name=D], Employee [id=2, name=B], Employee [id=1, name=A], Employee [id=3, name=C]]
Sorting Names in Desending Order :
[Employee [id=4, name=D], Employee [id=3, name=C], Employee [id=2, name=B], Employee [id=1, name=A]]
Sorting Names in Reverse Order :
[Employee [id=1, name=A], Employee [id=2, name=B], Employee [id=3, name=C], Employee [id=4, name=D]]

```java
public class Employee  {

int id;
String name;

public Employee() {
}

public Employee(int id, String name) {
this.id = id;
this.name = name;
}

@Override
public String toString() {
return "Employee [id=" + id + ", name=" + name + "]";
}
}
```

```java
public class CompareIds implements Comparator<Employee> {

// Ascending Order Id`s
@Override
public int compare(Employee o1, Employee o2) {
Integer i = o1.id;
return i.compareTo(o2.id);
}
}
```

```java
public class CompareNames  implements Comparator<Employee>{

//Ascending Order
@Override
public int compare(Employee o1, Employee o2) {
return o1.name.compareTo(o2.name);
}
}
```

```
public class Client {

public static void main(String[] args) {

Employee emp1 = new Employee(4, "A");
Employee emp2 = new Employee(2, "C");
Employee emp3 = new Employee(1, "B");
Employee emp4 = new Employee(5, "E");
Employee emp5 = new Employee(3, "D");

ArrayList<Employee> list = new ArrayList<Employee>();
list.add(emp1);
list.add(emp2);
list.add(emp3);
list.add(emp4);
list.add(emp5);

System.out.println("Insertion Order: " + list);

Collections.sort(list, new CompareIds());
System.out.println("Asending Order Id's: " + list);

Collections.sort(list, new CompareNames());
System.out.println("Asending Order Names: : " + list);
}
}
```

```
Insertion Order: [Employee [id=4, name=A], Employee [id=2, name=C], Employee [id=1, name=B], Employee [id=5, name=E], Employee [id=3, name=D]]
Asending Order Id's: [Employee [id=1, name=B], Employee [id=2, name=C], Employee [id=3, name=D], Employee [id=4, name=A], Employee [id=5, name=E]]
Asending Order Names: : [Employee [id=4, name=A], Employee [id=1, name=B], Employee [id=2, name=C], Employee [id=3, name=D], Employee [id=5, name=E]]
```