Multi-column Sorting

Program -

```
package Algorithms;
import java.time.LocalDate;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;
import java.util.Objects;
import java.util.*;
import java.util.stream.Collectors;

class Employee implements Comparable<Employee> {
    private int id;
    private String name;
    private double salary;
    private LocalDate joiningDate;

    public Employee(int id, String name, double salary, LocalDate joiningDate) {
        this.id = id;
        this.name = name;
        this.salary = salary;
        this.joiningDate = joiningDate;
    }

    public int getId() {
```

```
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public double getSalary() {
    return salary;
}

public void setSalary(double salary) {
    this.salary = salary;
}

public LocalDate getJoiningDate() {
    return joiningDate;
```

```
}

public void setJoiningDate(LocalDate joiningDate) {
    this.joiningDate = joiningDate;
}
@Override
public int compareTo(Employee anotherEmployee) {
    return this.getId() - anotherEmployee.getId();
}
// Two employees are equal if their IDs are equal
@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    Employee employee = (Employee) o;
    return id == employee.id;
}
@Override
public int hashCode() {
    return Objects.hash(id);
}
@Override
public String toString() {
    return "Employee{" +
            "id=" + id +
```

```
                ", name='" + name + '\'' +
                ", salary=" + salary +
                ", joiningDate=" + joiningDate +
                '}';
    }
}
```

// Sorting stream on multiple fields -

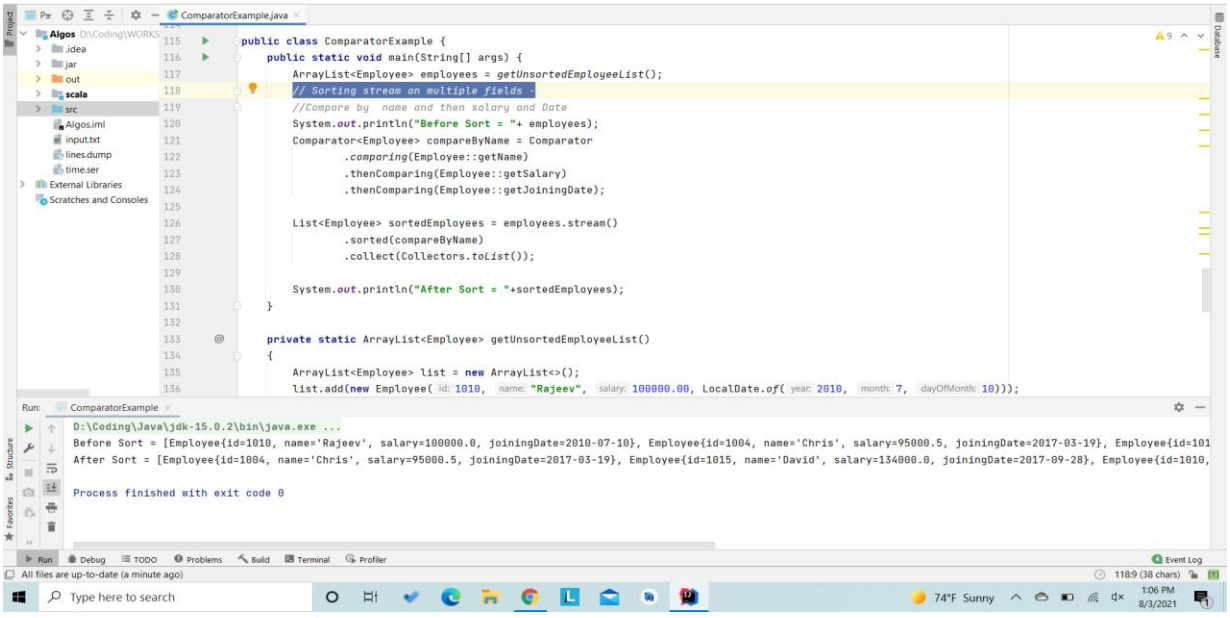
```
public class ComparatorExample {
    public static void main(String[] args) {
        ArrayList<Employee> employees = getUnsortedEmployeeList();
        // Sorting stream on multiple fields -
        //Compare by  name and then salary and Date
        System.out.println("Before Sort = "+ employees);
        Comparator<Employee> compareByName = Comparator
                .comparing(Employee::getName)
                .thenComparing(Employee::getSalary)
                .thenComparing(Employee::getJoiningDate);

        List<Employee> sortedEmployees = employees.stream()
                .sorted(compareByName)
                .collect(Collectors.toList());
        System.out.println("After Sort = "+sortedEmployees);
    }
    private static ArrayList<Employee> getUnsortedEmployeeList()
    {
```

```
        ArrayList<Employee> list = new ArrayList<>();
    list.add(new Employee(1010, "Rajeev", 100000.00, LocalDate.of(2010, 7, 10)));
     list.add(new Employee(1004, "Chris", 95000.50, LocalDate.of(2017, 3, 19)));
     list.add(new Employee(1015, "David", 134000.00, LocalDate.of(2017, 9, 28)));
      list.add(new Employee(1009, "Steve", 100000.00, LocalDate.of(2016, 5, 18)));
        return list;
    }
}
```

Output -

Before Sort = [Employee{id=1010, name='Rajeev', salary=100000.0, joiningDate=2010-07-10},
Employee{id=1004, name='Chris', salary=95000.5, joiningDate=2017-03-19},
Employee{id=1015, name='David', salary=134000.0, joiningDate=2017-09-28},
Employee{id=1009, name='Steve', salary=100000.0, joiningDate=2016-05-18}]

After Sort = [Employee{id=1004, name='Chris', salary=95000.5, joiningDate=2017-03-19},
Employee{id=1015, name='David', salary=134000.0, joiningDate=2017-09-28},
Employee{id=1010, name='Rajeev', salary=100000.0, joiningDate=2010-07-10},
Employee{id=1009, name='Steve', salary=100000.0, joiningDate=2016-05-18}]



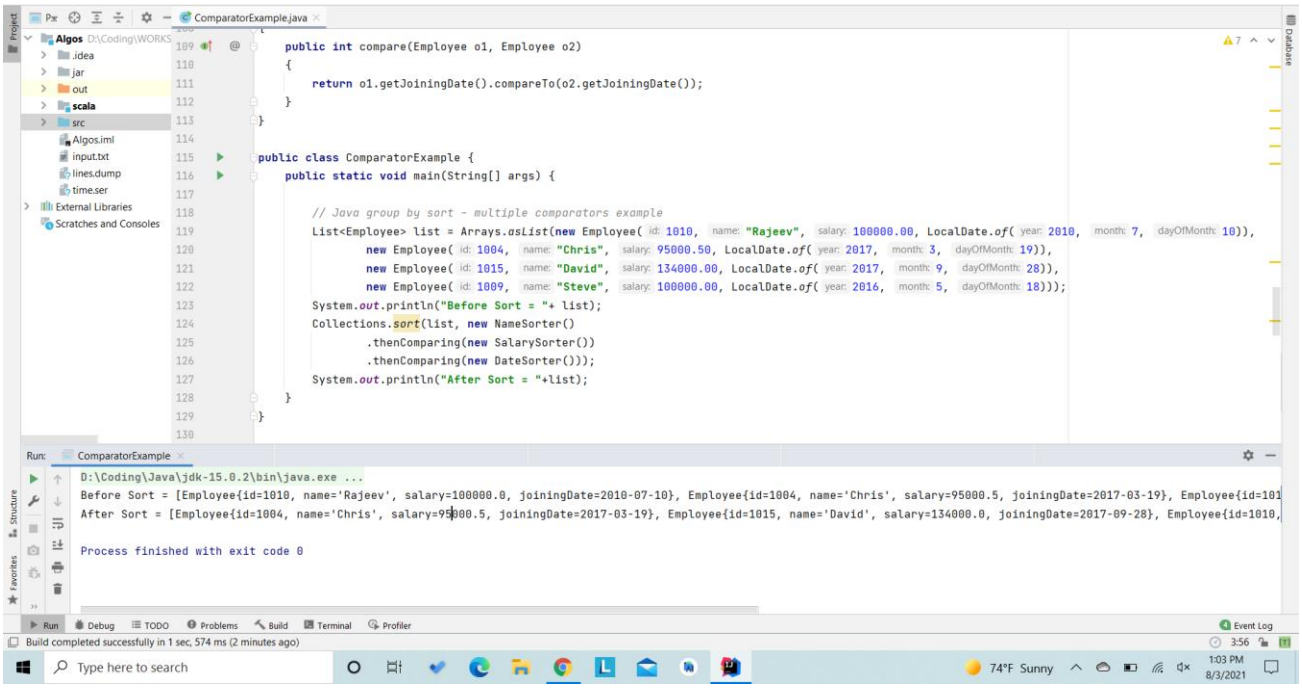
// Java group by sort – multiple comparators example

```
class NameSorter implements Comparator<Employee>
{
   public int compare(Employee o1, Employee o2)
   {
      return o1.getName().compareTo(o2.getName());
   }
}

class SalarySorter implements Comparator<Employee>
{
   public int compare(Employee e1, Employee e2) {
      if(e1.getSalary() < e2.getSalary()) {
         return -1;
      } else if (e1.getSalary() > e2.getSalary()) {
         return 1;
      } else {
         return 0;
      }
   }
}

class DateSorter implements Comparator<Employee>
{
   public int compare(Employee o1, Employee o2)
   {
```

```
        return o1.getJoiningDate().compareTo(o2.getJoiningDate());
    }
}
public class ComparatorExample {
    public static void main(String[] args) {
        // Java group by sort – multiple comparators example
        List<Employee> list = Arrays.asList(new Employee(1010, "Rajeev",
100000.00, LocalDate.of(2010, 7, 10)),
                new Employee(1004, "Chris", 95000.50, LocalDate.of(2017, 3, 19)),
                new Employee(1015, "David", 134000.00, LocalDate.of(2017, 9, 28)),
                new Employee(1009, "Steve", 100000.00, LocalDate.of(2016, 5, 18)));
        System.out.println("Before Sort = "+ list);
        Collections.sort(list, new NameSorter()
                .thenComparing(new SalarySorter())
                .thenComparing(new DateSorter()));
        System.out.println("After Sort = "+list);
}
```

Output -

```
Before Sort = [Employee{id=1010, name='Rajeev', salary=100000.0, joiningDate=2010-07-10},
Employee{id=1004, name='Chris', salary=95000.5, joiningDate=2017-03-19},
Employee{id=1015, name='David', salary=134000.0, joiningDate=2017-09-28},
Employee{id=1009, name='Steve', salary=100000.0, joiningDate=2016-05-18}]

After Sort = [Employee{id=1004, name='Chris', salary=95000.5, joiningDate=2017-03-19},
Employee{id=1015, name='David', salary=134000.0, joiningDate=2017-09-28},
Employee{id=1010, name='Rajeev', salary=100000.0, joiningDate=2010-07-10},
Employee{id=1009, name='Steve', salary=100000.0, joiningDate=2016-05-18}]
```


ComparatorExample.java
Algos D:\Coding\WORKS
.idea
jar
out
scala
src
Algos.iml
input.txt
lines.dump
time.ser
External Libraries
Scratches and Consoles
public int compare(Employee o1, Employee o2)
{
return o1.getJoiningDate().compareTo(o2.getJoiningDate());
}
}
public class ComparatorExample {
public static void main(String[] args) {
// Java group by sort - multiple comparators example
List<Employee> list = Arrays.asList(new Employee(id: 1010, name: "Rajeev", salary: 100000.00, LocalDate.of(year: 2010, month: 7, dayOfMonth: 10)),
new Employee(id: 1004, name: "Chris", salary: 95000.50, LocalDate.of(year: 2017, month: 3, dayOfMonth: 19)),
new Employee(id: 1015, name: "David", salary: 134000.00, LocalDate.of(year: 2017, month: 9, dayOfMonth: 28)),
new Employee(id: 1009, name: "Steve", salary: 100000.00, LocalDate.of(year: 2016, month: 5, dayOfMonth: 18)));
System.out.println("Before Sort = "+ list);
Collections.sort(list, new NameSorter()
.thenComparing(new SalarySorter())
.thenComparing(new DateSorter()));
System.out.println("After Sort = "+list);
}
}
Run: ComparatorExample
D:\Coding\Java\jdk-15.0.2\bin\java.exe ...
Before Sort = [Employee{id=1010, name='Rajeev', salary=100000.0, joiningDate=2010-07-10}, Employee{id=1004, name='Chris', salary=95000.5, joiningDate=2017-03-19}, Employee{id=101
After Sort = [Employee{id=1004, name='Chris', salary=95000.5, joiningDate=2017-03-19}, Employee{id=1015, name='David', salary=134000.0, joiningDate=2017-09-28}, Employee{id=1010,
Process finished with exit code 0
Run
Debug
TODO
Problems
Build
Terminal
Profiler
Event Log
Build completed successfully in 1 sec, 574 ms (2 minutes ago)
Type here to search
74°F Sunny
1:03 PM
8/3/2021