Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 2 REPORT

ELİF AKGÜN
1801042251

Course Assistant: Ayşe Şerbetçi Turan

1 INTRODUCTION

1.1 Problem Definition

In this assignment, an ExperimentList class was created to track some machine learning experiments and results. The ExperimentList class is in the form of Single Linked List. The experiments are connected to each other in this structure. The first experiment of each day shows the other day, nex day of the other experiments show null. Experiments and days are listed in this structure.

A machine learning experiment consists of the following instance variables:

- setup (String): explains the experimental setup
- day(integer): represents the day of start
- time(Time): represents the time of start
- completed(boolean): indicates whether it is completed or not
- accuracy(float): represents the output (not a valid value if the experiment is not completed)

This structure must has addExp(Experiment), getExp(day, index), setExp(day, index,), removeExp(day, index), listExp(day), removeDay(day), orderDay(day), orderExperiments() methods.

1.2 System Requirements

The ExperimentList class methods requirements:

addExp(Experiment): The add method, which takes only one parameter, takes the Experiment object as a parameter and insert experiment to the end of the day.

getExp(day, index): This method takes two parameters, Experiment object and a integer which represents the day, and get the experiment with the given day and position.

setExp(day, index, Experiment): This method takes three parameters, Experiment object and two integer that represent the day and index, and sets the experiment with the given day and position. According to their index and day, statements can change.

removeExp(day, index): The removeExp method takes two integers as a parameter which represent day and index of experiment. This method remove the experiment specified as index from given day

listExp(day): listExp method takes a integer as a parameter that represents the day. This method lists all completed experiments in the given day.

removeDay(day): removeDay method takes a integer as a parameter that represents the day. This method remove all experiments in a given day.

orderDay(day): orderDay method takes a integer as a parameter that represents the day. This method sorts the experiments in a given day according to the accuracy, the changes will be done on the list.

orderExperiments(): orderExperiments takes no parameter and sorts all the experiments in the list according to the accuracy. List don't change at last.

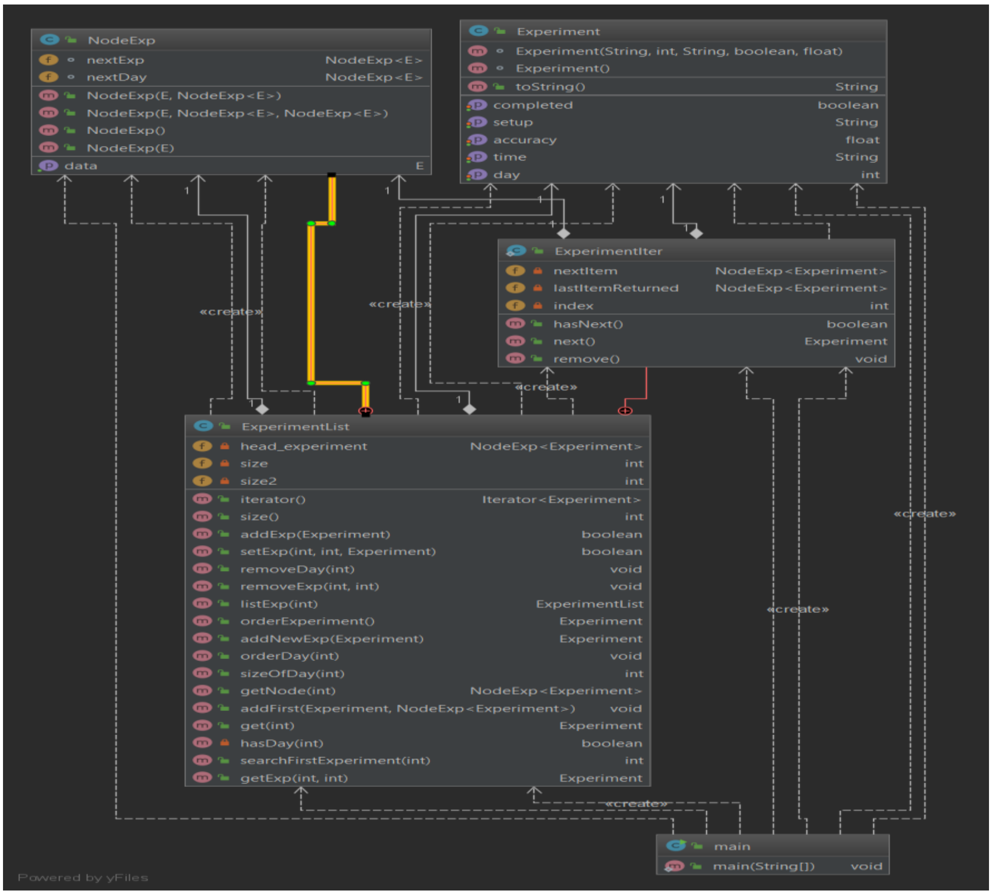
The ExperimentList class inner class requirements:

public class NodeExp<E>: This class builds block fort he ExperimentList

public static class ExperimentIter implements Iterator<Experiment>: This class implements Iterator class and overrides its methods.

2 METHOD

2.1 Class Diagrams



2.2 Use Case Diagrams

Add use case diagrams if required.

2.3 Other Diagrams (optional)

Add other diagrams if required.

2.4 Problem Solution Approach

During the project development phase, the objects that should be used first were considered and classes were created for those objects. Some additional methods were needed in this project.
sizeOfDay(int day): Takes a parameter that represents given day and returns size of the day.
getNode(int index): Find the node at a specified index and returns the node at index or null if it does not exist.

addFirs(Experiment item, NodeExp<Experiment> next_data): Insert an item as the first item of the list.

get(int index): Get the data value at index and returns the data at index.

hasDay(int day): If the list includes given day, returns true.

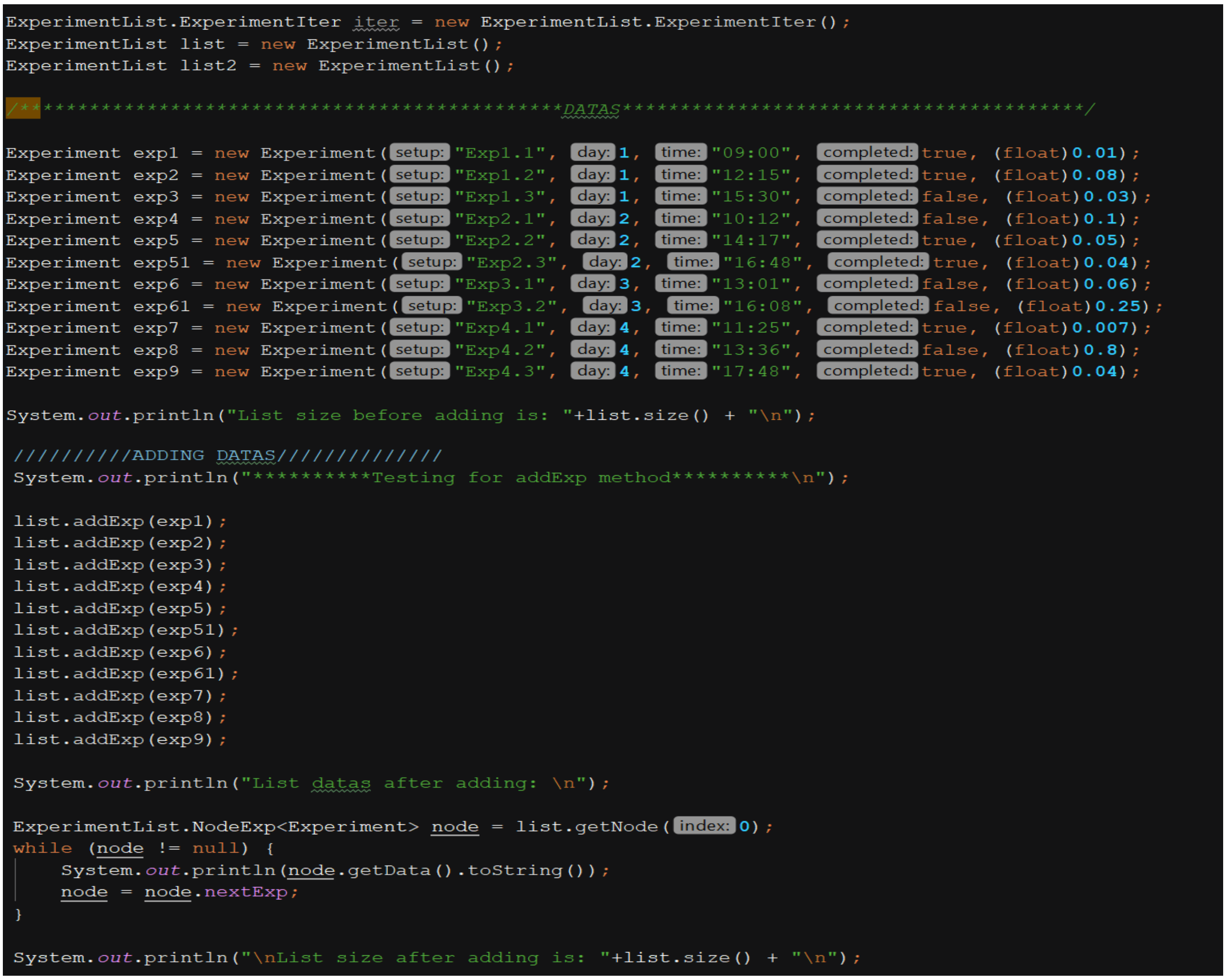
searcFirstExperiment(int day): Returns index that first index of given day.

And additional nested class is NodeExp<E> class builds block for the ExperimentList.

3 RESULT

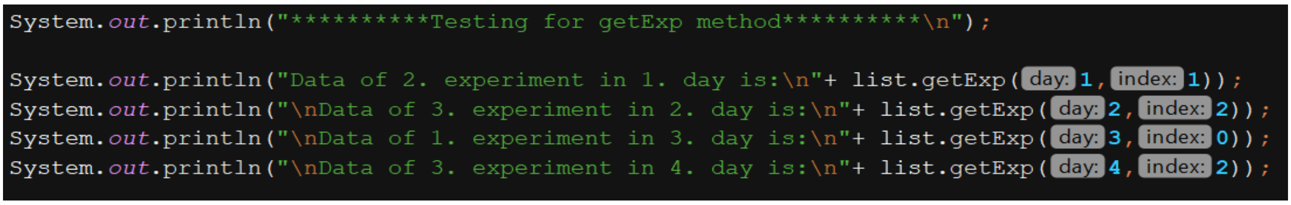
3.1 Test Cases

1)Test for addExp method: Firstly the objects that were created were sent to the addExp method and checks if they were listed correctly and the size.



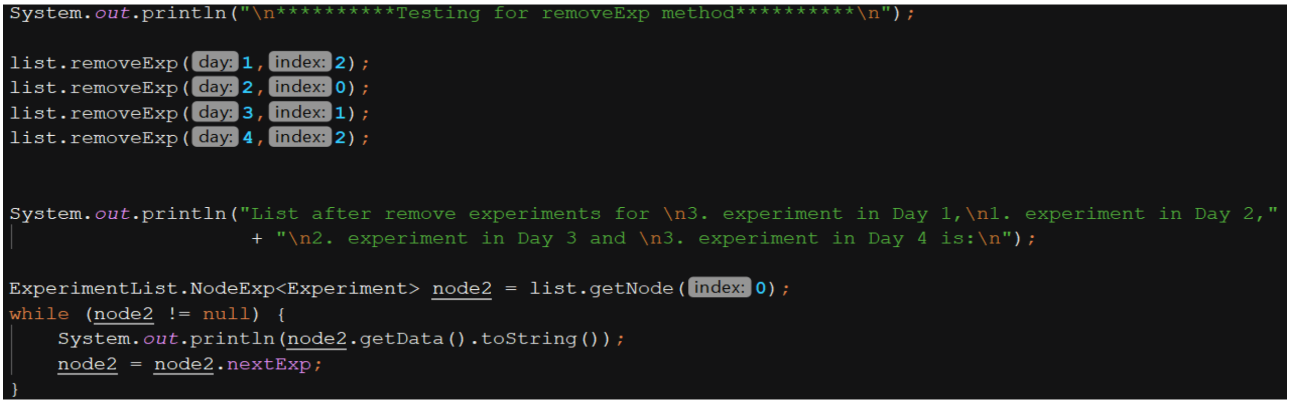
```
ExperimentList.ExperimentIter iter = new ExperimentList.ExperimentIter();
ExperimentList list = new ExperimentList();
ExperimentList list2 = new ExperimentList();

/**********************************************DATAS*****************************************/

Experiment exp1 = new Experiment(setup: "Exp1.1", day: 1, time: "09:00", completed: true, (float)0.01);
Experiment exp2 = new Experiment(setup: "Exp1.2", day: 1, time: "12:15", completed: true, (float)0.08);
Experiment exp3 = new Experiment(setup: "Exp1.3", day: 1, time: "15:30", completed: false, (float)0.03);
Experiment exp4 = new Experiment(setup: "Exp2.1", day: 2, time: "10:12", completed: false, (float)0.1);
Experiment exp5 = new Experiment(setup: "Exp2.2", day: 2, time: "14:17", completed: true, (float)0.05);
Experiment exp51 = new Experiment(setup: "Exp2.3", day: 2, time: "16:48", completed: true, (float)0.04);
Experiment exp6 = new Experiment(setup: "Exp3.1", day: 3, time: "13:01", completed: false, (float)0.06);
Experiment exp61 = new Experiment(setup: "Exp3.2", day: 3, time: "16:08", completed: false, (float)0.25);
Experiment exp7 = new Experiment(setup: "Exp4.1", day: 4, time: "11:25", completed: true, (float)0.007);
Experiment exp8 = new Experiment(setup: "Exp4.2", day: 4, time: "13:36", completed: false, (float)0.8);
Experiment exp9 = new Experiment(setup: "Exp4.3", day: 4, time: "17:48", completed: true, (float)0.04);

System.out.println("List size before adding is: "+list.size() + "\n");

 //////////ADDING DATAS//////////////
 System.out.println("**********Testing for addExp method**********\n");

 list.addExp(exp1);
 list.addExp(exp2);
 list.addExp(exp3);
 list.addExp(exp4);
 list.addExp(exp5);
 list.addExp(exp51);
 list.addExp(exp6);
 list.addExp(exp61);
 list.addExp(exp7);
 list.addExp(exp8);
 list.addExp(exp9);

 System.out.println("List datas after adding: \n");

 ExperimentList.NodeExp<Experiment> node = list.getNode(index: 0);
 while (node != null) {
     System.out.println(node.getData().toString());
     node = node.nextExp;
 }

 System.out.println("\nList size after adding is: "+list.size() + "\n");
```

2)Test for getExp method: Given day and index information is sent to the method and checks whether the printed datas were correct.



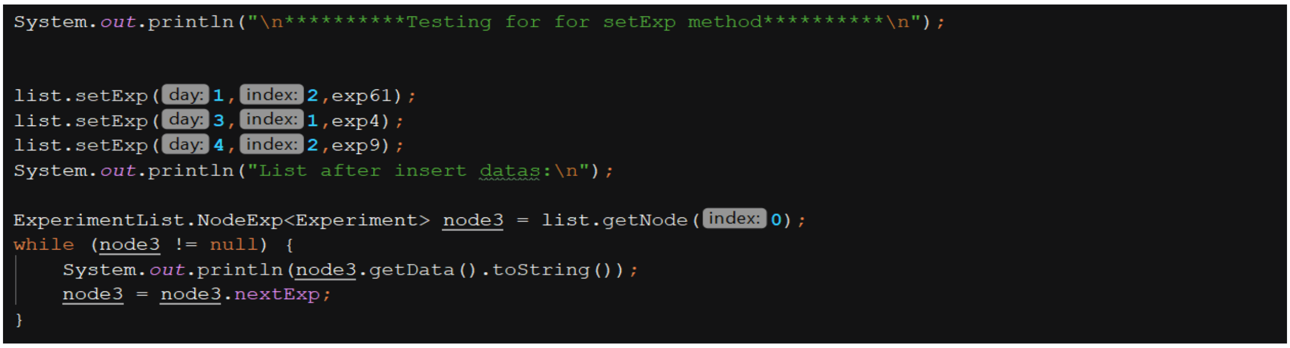
```
System.out.println("**********Testing for getExp method**********\n");

System.out.println("Data of 2. experiment in 1. day is:\n"+ list.getExp( day: 1, index: 1));
System.out.println("\nData of 3. experiment in 2. day is:\n"+ list.getExp( day: 2, index: 2));
System.out.println("\nData of 1. experiment in 3. day is:\n"+ list.getExp( day: 3, index: 0));
System.out.println("\nData of 3. experiment in 4. day is:\n"+ list.getExp( day: 4, index: 2));
```

3)Test for removeExp method: Given day and index information were sent to the method and checks whether the correct datas were deleted.



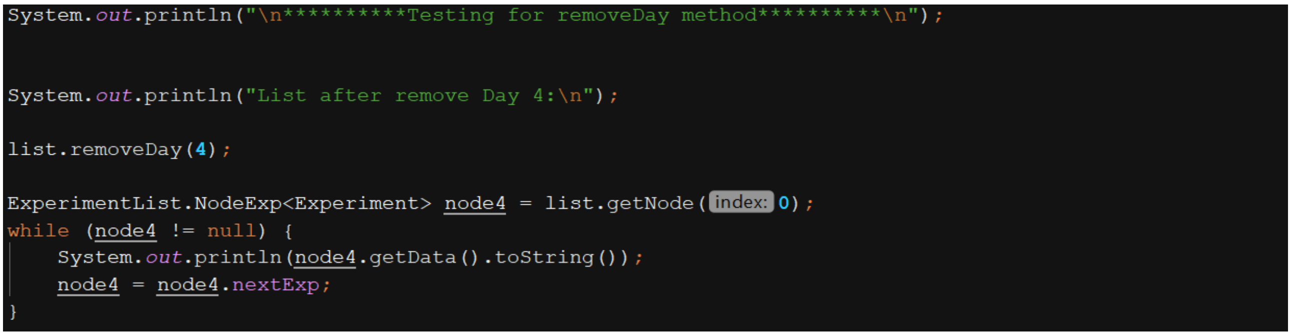
```
System.out.println("\n**********Testing for removeExp method**********\n");

list.removeExp( day: 1, index: 2);
list.removeExp( day: 2, index: 0);
list.removeExp( day: 3, index: 1);
list.removeExp( day: 4, index: 2);


System.out.println("List after remove experiments for \n3. experiment in Day 1,\n1. experiment in Day 2,"
        + "\n2. experiment in Day 3 and \n3. experiment in Day 4 is:\n");

ExperimentList.NodeExp<Experiment> node2 = list.getNode( index: 0);
while (node2 != null) {
    System.out.println(node2.getData().toString());
    node2 = node2.nextExp;
}
```

4)Testing for setExp method: Given day, index and Experiment information were sent to the method and checks whether the correct data was added.



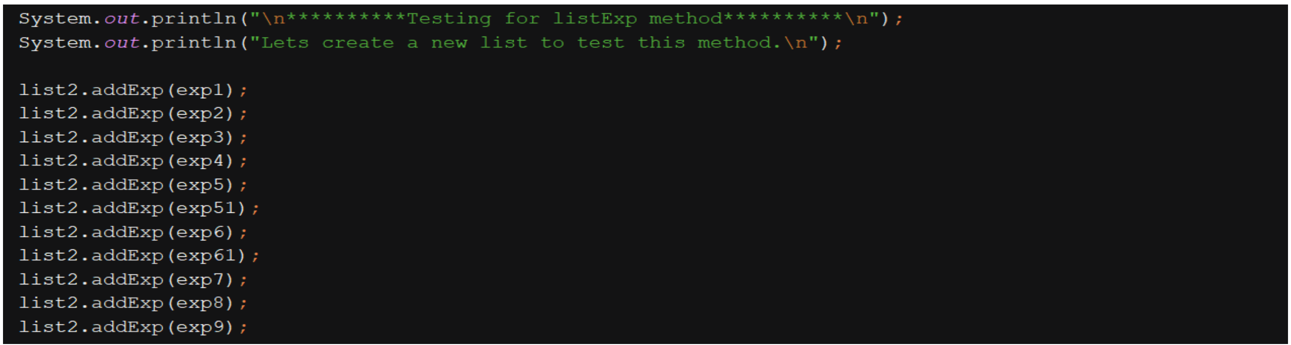
```
System.out.println("\n**********Testing for for setExp method**********\n");


list.setExp( day: 1, index: 2,exp61);
list.setExp( day: 3, index: 1,exp4);
list.setExp( day: 4, index: 2,exp9);
System.out.println("List after insert datas:\n");

ExperimentList.NodeExp<Experiment> node3 = list.getNode( index: 0);
while (node3 != null) {
    System.out.println(node3.getData().toString());
    node3 = node3.nextExp;
}
```

5)Testing for removeDay method: Given the day information was sent to the method and checks whether the correct day deleted.

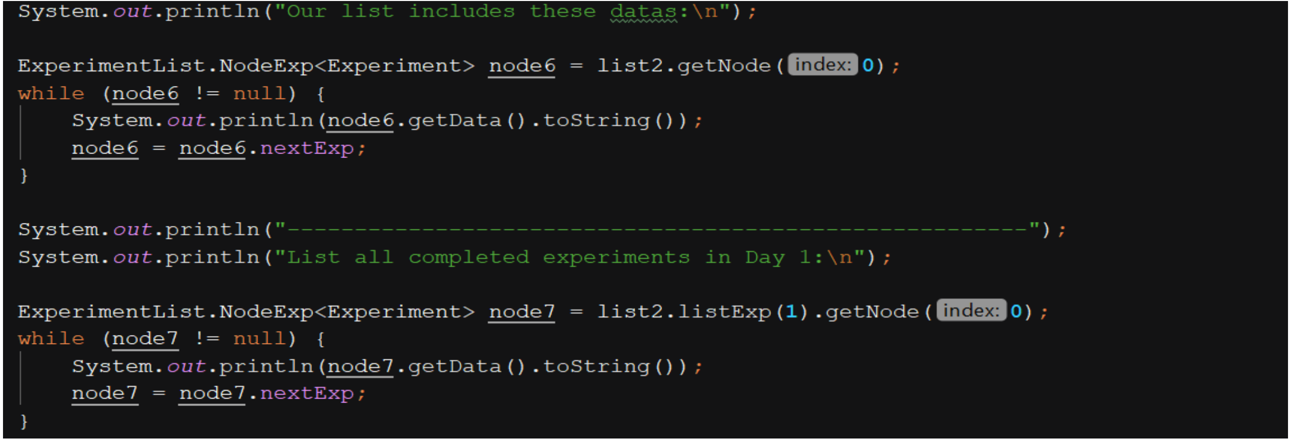


```
System.out.println("\n**********Testing for removeDay method**********\n");


System.out.println("List after remove Day 4:\n");

list.removeDay(4);

ExperimentList.NodeExp<Experiment> node4 = list.getNode( index: 0);
while (node4 != null) {
    System.out.println(node4.getData().toString());
    node4 = node4.nextExp;
}
```

6)Testing for listExp method: Given day information was sent to the method and checks whether the correct list all completed experiments in a given day.

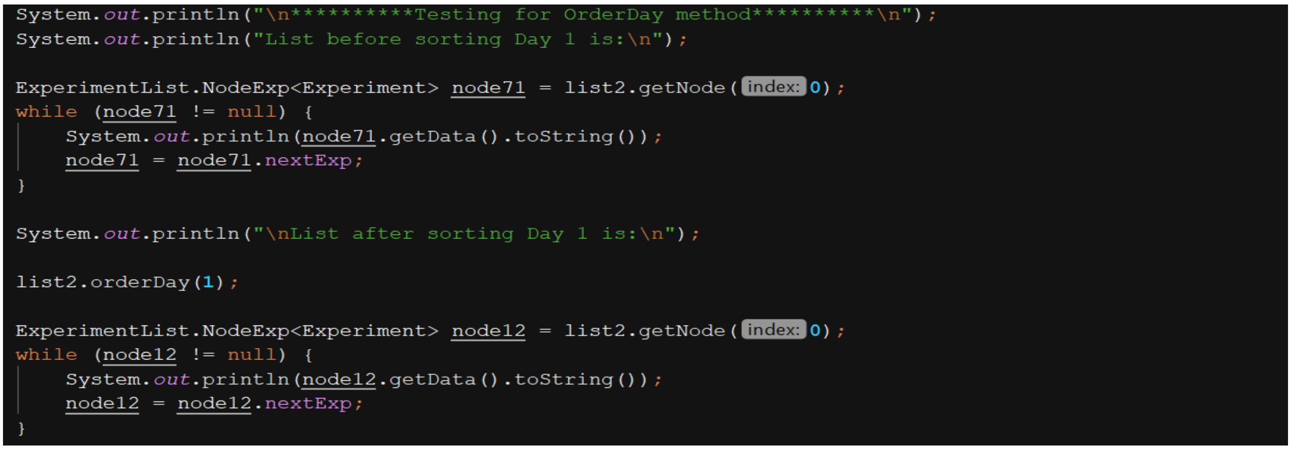


```
System.out.println("\n**********Testing for listExp method**********\n");
System.out.println("Lets create a new list to test this method.\n");

list2.addExp(exp1);
list2.addExp(exp2);
list2.addExp(exp3);
list2.addExp(exp4);
list2.addExp(exp5);
list2.addExp(exp51);
list2.addExp(exp6);
list2.addExp(exp61);
list2.addExp(exp7);
list2.addExp(exp8);
list2.addExp(exp9);
```



```
System.out.println("Our list includes these datas:\n");

ExperimentList.NodeExp<Experiment> node6 = list2.getNode( index: 0);
while (node6 != null) {
    System.out.println(node6.getData().toString());
    node6 = node6.nextExp;
}

System.out.println("----------------------------------------------------");
System.out.println("List all completed experiments in Day 1:\n");

ExperimentList.NodeExp<Experiment> node7 = list2.listExp(1).getNode( index: 0);
while (node7 != null) {
    System.out.println(node7.getData().toString());
    node7 = node7.nextExp;
}
```

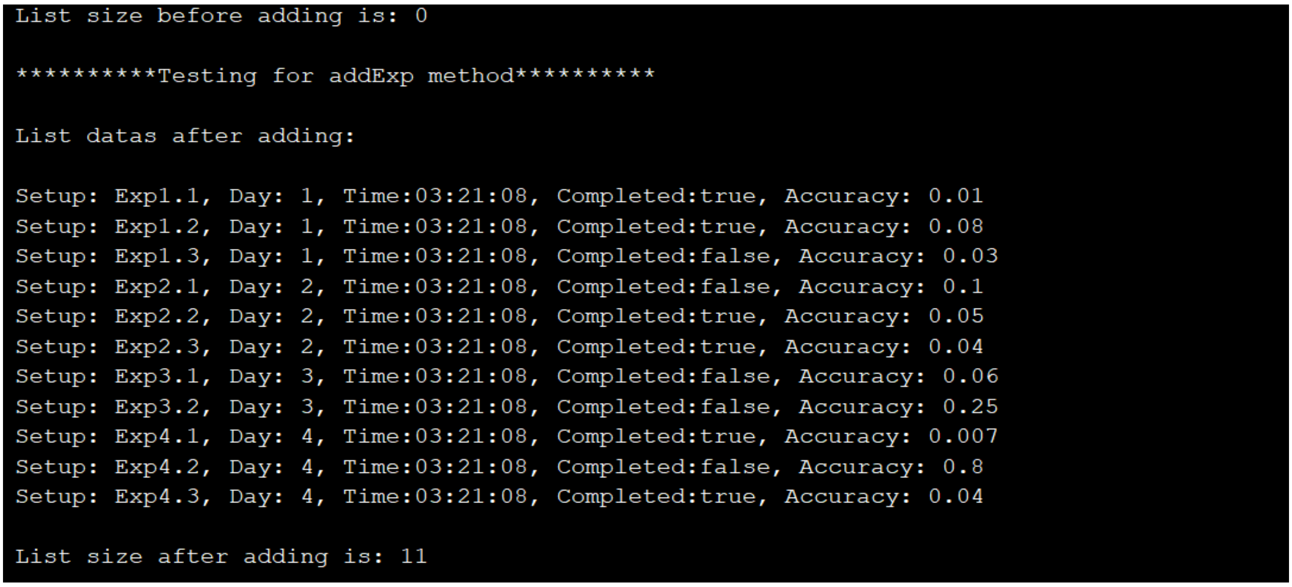
7)Testing for orderDay method: Given day information were sent to the method and checked whether correct sorts the datas.



```
System.out.println("\n**********Testing for OrderDay method**********\n");
System.out.println("List before sorting Day 1 is:\n");

ExperimentList.NodeExp<Experiment> node71 = list2.getNode( index: 0);
while (node71 != null) {
    System.out.println(node71.getData().toString());
    node71 = node71.nextExp;
}

System.out.println("\nList after sorting Day 1 is:\n");

list2.orderDay(1);

ExperimentList.NodeExp<Experiment> node12 = list2.getNode( index: 0);
while (node12 != null) {
    System.out.println(node12.getData().toString());
    node12 = node12.nextExp;
}
```

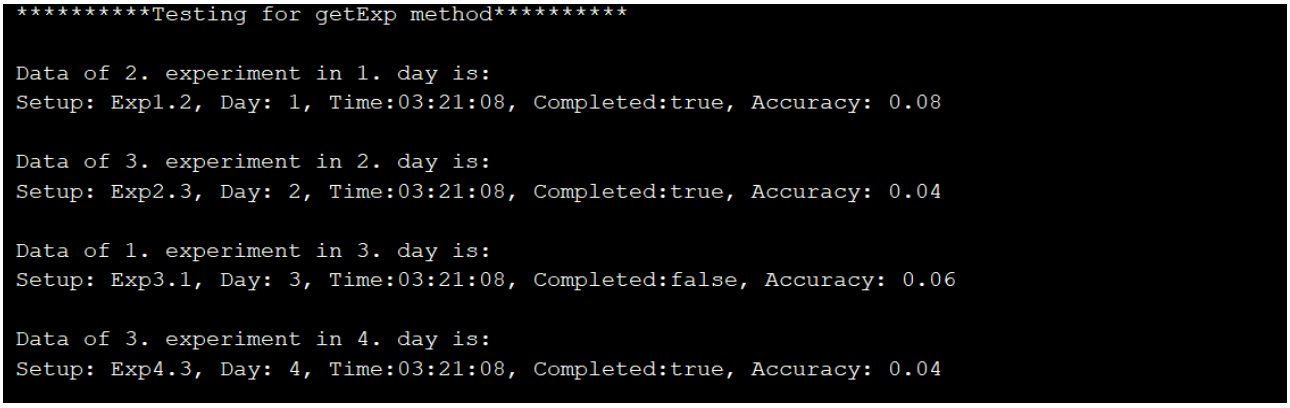
3.2 Running Results

1)Result of addExp method:



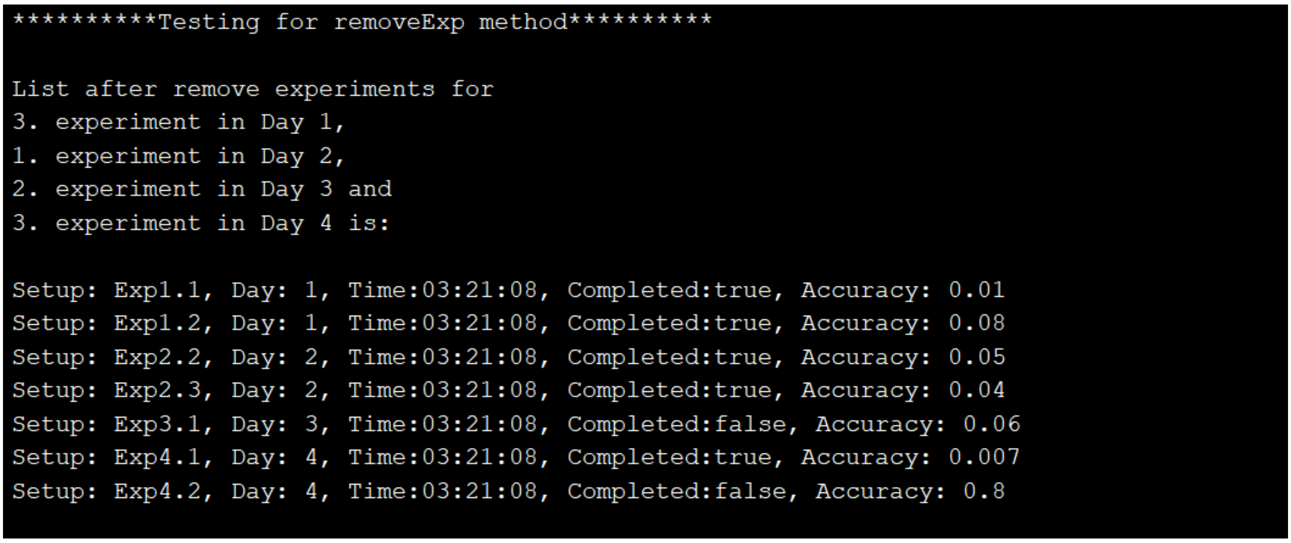
```
List size before adding is: 0

**********Testing for addExp method**********

List datas after adding:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
Setup: Exp1.3, Day: 1, Time:03:21:08, Completed:false, Accuracy: 0.03
Setup: Exp2.1, Day: 2, Time:03:21:08, Completed:false, Accuracy: 0.1
Setup: Exp2.2, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.05
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06
Setup: Exp3.2, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.25
Setup: Exp4.1, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.007
Setup: Exp4.2, Day: 4, Time:03:21:08, Completed:false, Accuracy: 0.8
Setup: Exp4.3, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.04

List size after adding is: 11
```

2)Result of getExp method:



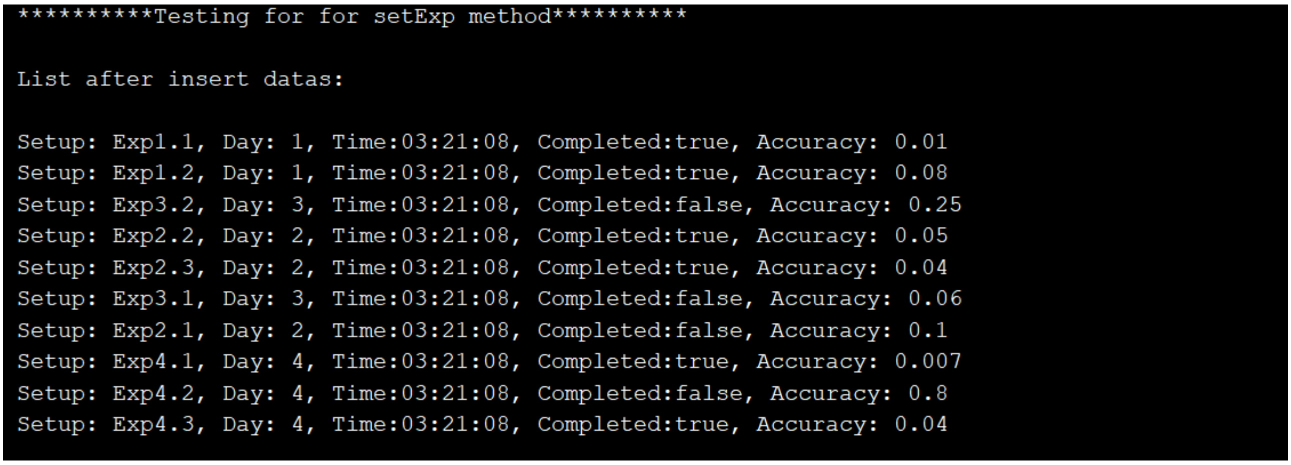
```
**********Testing for getExp method**********

Data of 2. experiment in 1. day is:
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08

Data of 3. experiment in 2. day is:
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04

Data of 1. experiment in 3. day is:
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06

Data of 3. experiment in 4. day is:
Setup: Exp4.3, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.04
```

3) Result of removeExp method:



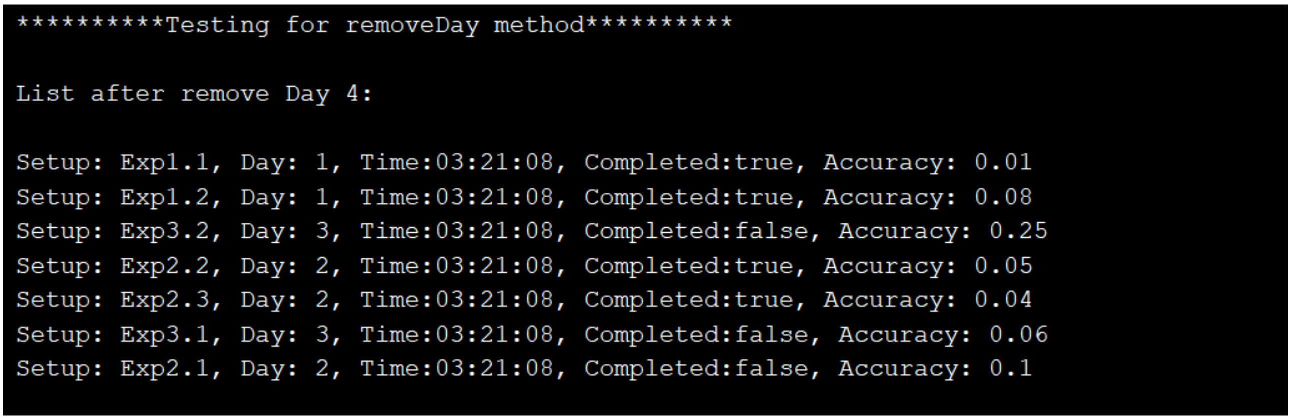
```
**********Testing for removeExp method**********

List after remove experiments for
3. experiment in Day 1,
1. experiment in Day 2,
2. experiment in Day 3 and
3. experiment in Day 4 is:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
Setup: Exp2.2, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.05
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06
Setup: Exp4.1, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.007
Setup: Exp4.2, Day: 4, Time:03:21:08, Completed:false, Accuracy: 0.8
```

4) Result of setExp method:



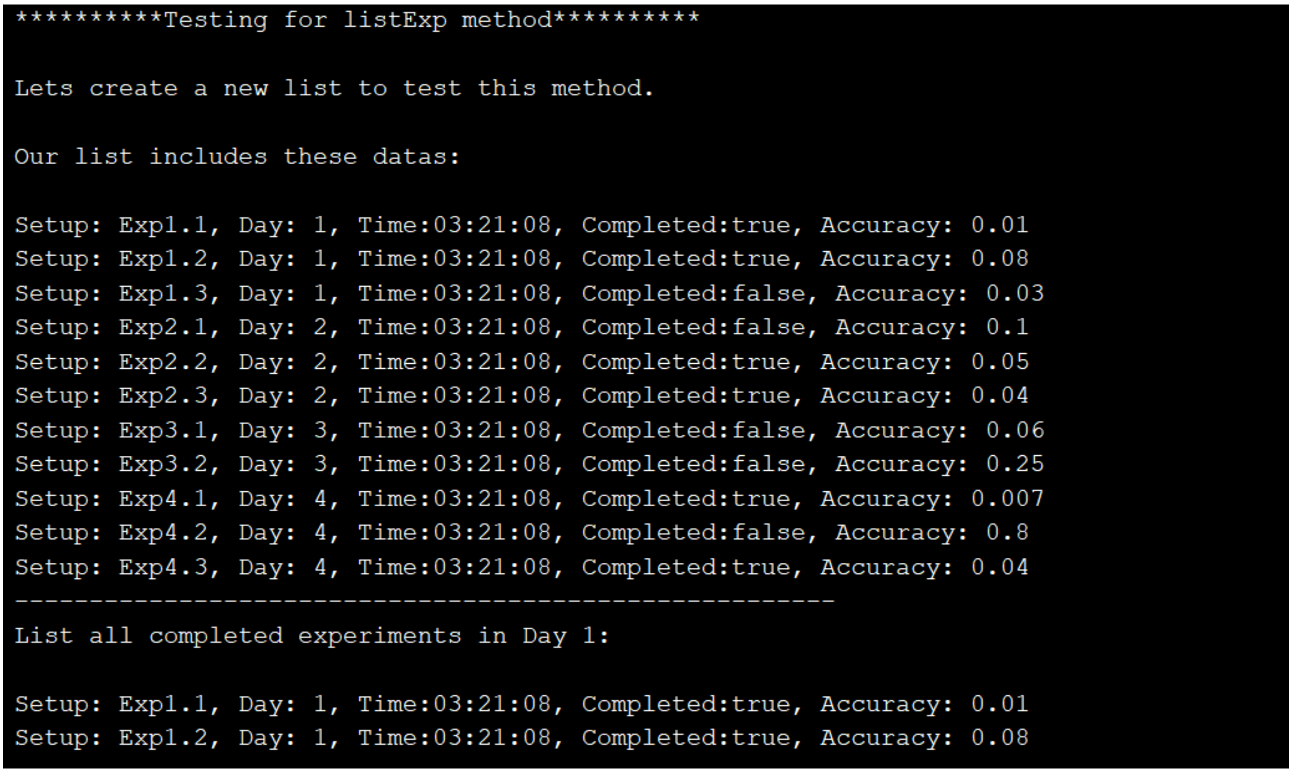
```
**********Testing for for setExp method**********

List after insert datas:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
Setup: Exp3.2, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.25
Setup: Exp2.2, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.05
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06
Setup: Exp2.1, Day: 2, Time:03:21:08, Completed:false, Accuracy: 0.1
Setup: Exp4.1, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.007
Setup: Exp4.2, Day: 4, Time:03:21:08, Completed:false, Accuracy: 0.8
Setup: Exp4.3, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.04
```

5) Result of removeDay method:



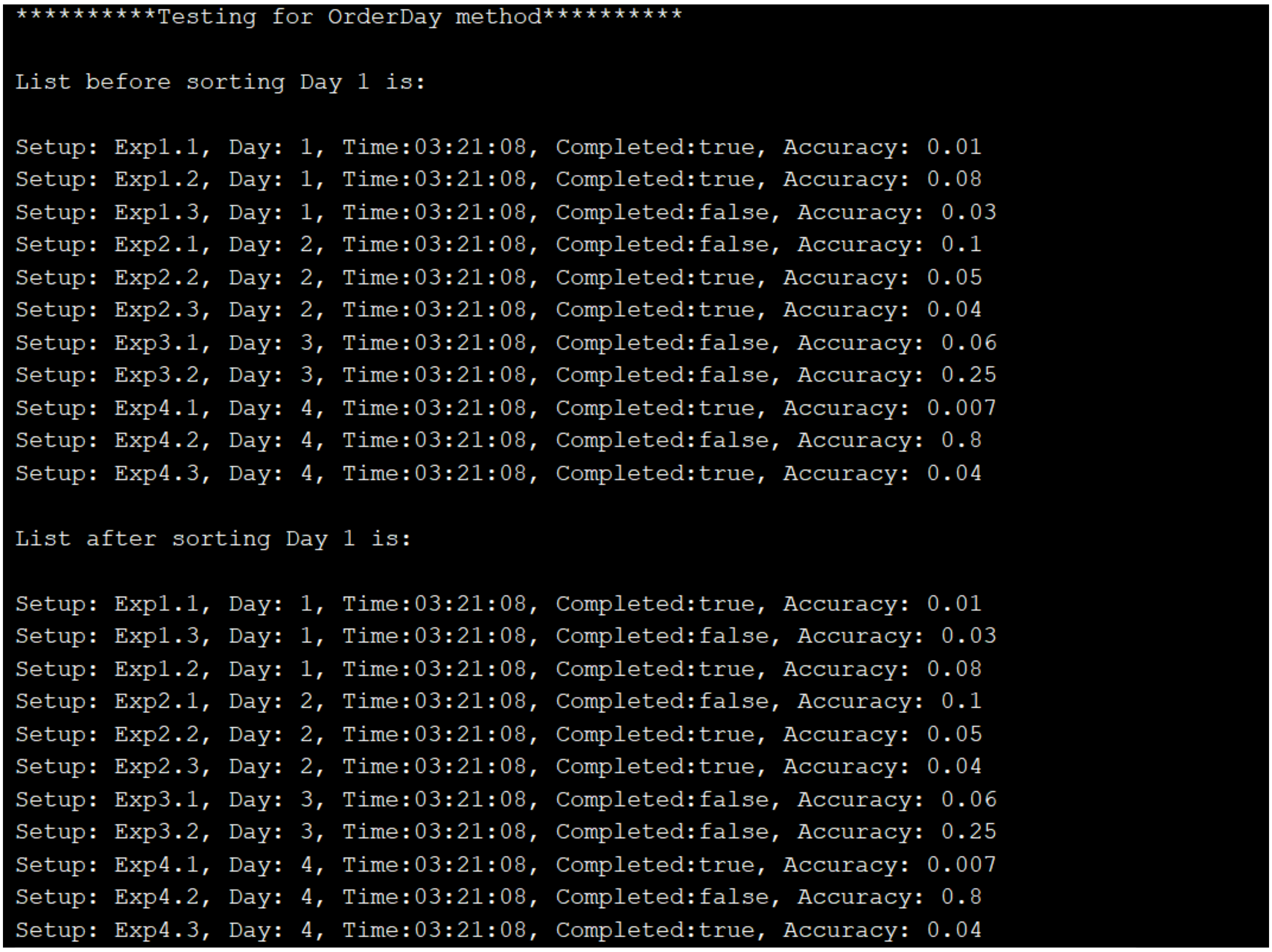
```
**********Testing for removeDay method**********

List after remove Day 4:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
Setup: Exp3.2, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.25
Setup: Exp2.2, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.05
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06
Setup: Exp2.1, Day: 2, Time:03:21:08, Completed:false, Accuracy: 0.1
```

6) Result of listExp method:



```
**********Testing for listExp method**********

Lets create a new list to test this method.

Our list includes these datas:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
Setup: Exp1.3, Day: 1, Time:03:21:08, Completed:false, Accuracy: 0.03
Setup: Exp2.1, Day: 2, Time:03:21:08, Completed:false, Accuracy: 0.1
Setup: Exp2.2, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.05
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06
Setup: Exp3.2, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.25
Setup: Exp4.1, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.007
Setup: Exp4.2, Day: 4, Time:03:21:08, Completed:false, Accuracy: 0.8
Setup: Exp4.3, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.04
-------------------------------------------------------
List all completed experiments in Day 1:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
```

7) Result of orderDay method:



```
**********Testing for OrderDay method**********

List before sorting Day 1 is:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
Setup: Exp1.3, Day: 1, Time:03:21:08, Completed:false, Accuracy: 0.03
Setup: Exp2.1, Day: 2, Time:03:21:08, Completed:false, Accuracy: 0.1
Setup: Exp2.2, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.05
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06
Setup: Exp3.2, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.25
Setup: Exp4.1, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.007
Setup: Exp4.2, Day: 4, Time:03:21:08, Completed:false, Accuracy: 0.8
Setup: Exp4.3, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.04

List after sorting Day 1 is:

Setup: Exp1.1, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.01
Setup: Exp1.3, Day: 1, Time:03:21:08, Completed:false, Accuracy: 0.03
Setup: Exp1.2, Day: 1, Time:03:21:08, Completed:true, Accuracy: 0.08
Setup: Exp2.1, Day: 2, Time:03:21:08, Completed:false, Accuracy: 0.1
Setup: Exp2.2, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.05
Setup: Exp2.3, Day: 2, Time:03:21:08, Completed:true, Accuracy: 0.04
Setup: Exp3.1, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.06
Setup: Exp3.2, Day: 3, Time:03:21:08, Completed:false, Accuracy: 0.25
Setup: Exp4.1, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.007
Setup: Exp4.2, Day: 4, Time:03:21:08, Completed:false, Accuracy: 0.8
Setup: Exp4.3, Day: 4, Time:03:21:08, Completed:true, Accuracy: 0.04
```

3.3 TIME COMPLEXITY ANALYSIS

METHOD	EXPLAIN
addExp(Experiment)	This method includes if-else statements so its complexity is O(1).
getExp(day,index)	This method includes if-else statements and a for loop. If-else statements complexity are O(1) and for loop is O(n). So its complexity is O(1)+O(n)=O(n).
setExp(day, index,Experiment)	This method includes if-else statements and a for loop. If-else statements complexity are O(1) and for loop is O(n). So its complexity is O(1)+O(n)=O(n). Best case is O(1) and worst case is O(n).
removeExp(day, index)	This method includes if-else statements and a for loops. If-else statements complexity are O(1) and for loops are O(n). So its complexity is O(1)+O(n)=O(n).
listExp(day)	This method includes if-else statements so its complexity is O(1).
removeDay(day)	This method includes if-else statements so its complexity is O(1).

orderDay(day):	This method includes two nested for loops. So its complexity is O(n)*O(n)=O(n^2).
orderExperiments()	This method includes loop so its complexity is O(n).
getNode(int index)	This method includes for loop so its complexity is O(n). Best case and worst case are also O(n).
addFirst(Experimet e, NodeExp data)	This method includes if-else statements so its complexity is O(1).
get(int index)	This method invode getNode method so its complexity is O(n).
hasDay(int day)	This method includes if-else statements so its complexity is O(1).
searcFirstExperiment(int day)	This method includes if-else statements so its complexity is O(1).