Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 2 REPORT

Yusuf Can Kan
161044007

Course Assistant: Ayşe Şerbetçi Turan

1 INTRODUCTION

1.1 Problem Definition

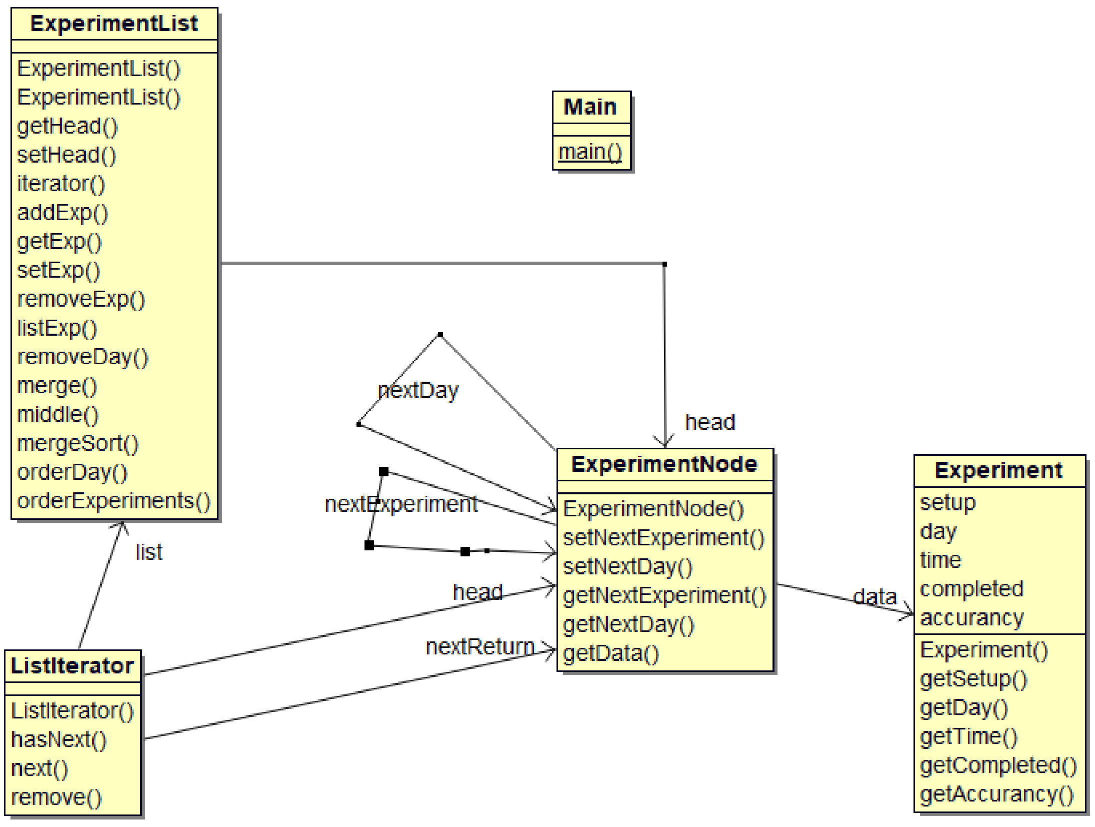
In this homework the problem is to build a experiment list structure. All experiments has experimental setup information, experiments start day, experiments start time, experiment completed or not information and accurancy(output of experiment) information. All this experiments must be linked with each other in order to day orders. Besides the day order, every days first experiment need to be link each other for make easy to reaching experiments. In addition we need to do some operations over the list. For example adding new experiments in a proper place, removing experiments, removing experiments with respect to their days, reaching experiments and making modifications of it, and sorting experiments in order to their accurancy rates.

1.2 System Requirements

Firstly we need to create experiments for the list and for this we need to experiment structure which can represent with class. For creating experiment structures we need some informations such as setup,accurancy, etc. which are mentioned in the problem definition. When data structure creating, if completed value is not true, accurancy shouldn't be valid value. Besides that we need to create determined time structure for preventing confusion. For linking experiments to each other we need to create node structure and inside of node structure we have create two linked list structure, one for linking experiments with each other, and the other for linking every days first experiments with each other. We need to make other experiments day structures to null for preventing the confusion. We need another structure that holds head of the experiment list nodes for making operations easily over the list. For making operations, this structue should have some methods which are using for adding experiments, removing experiments, listing experiments,sorting experiments,etc. For the last one, we need iterator structure for advancing forward direction over the list or making some operations such as removing some experiments.

2 METHOD

2.1 Class Diagrams



2.2 Use Case Diagrams

Use case diagram is not required for this homework.

2.3 Other Diagrams (optional)

Other diagrams are not required fort his homework.

2.4 Problem Solution Approach

First we must store experiments information so we have to create Experiments class that has some information fields such as setup,day,time,completed,accurancy. When we creating experiments class we have to carefull about accurancy and completed. If completed is false it does not have to accurancy information so it must set inside of constructor. After that we need to linked experiments to each other so we need to node structure(ExperimentNode class) which has next node for next experiments and for the day structure it must have additionally nextday field node. After that we have to implement new structure for storing list information(ExperimentList). It must hold head information and other methods which we can modify list with that. First method which is implemented is addExp() method. The thing which we have to careful in this method is, when we added new experiment in a days first element, we have to set day list to. In this situation we have to set back day node and old element next day node and new next day node.The other experiments are set and get experiments. We have to carefull about set experiment for the day structure. The others are removeExperiments, removeDay and listExperiment methods. The last ones are orderDay and orderExperiment which we have to write sort algorithm for that and it must be efficient. I choose merge sort which has complexity of O(nlogn). Besides that we have to implement iterator method inside of this class and for implementing this method we have to create our own iterator class. This class has iterator methods for list(next,hasNext,remove). We have to carefull about remove because it must remove the last element which returned by next method. So if next method doesn't call it doesn't remove anything. We have to careful about day list when we implementing remove method. When we implementing iterator methods and other list methods we have to carefull about list structures, for example if the modifying element is first element the head node must be changes and the nextday and next experiment nodes must be changes. OrderExperiments method doesn't modify the list because of the day list may be damage. So we have to copy the main list and we have to modify it and return it.

Complexity Table

Experiment(String...)	Constructor.	O(1)
getSetup()	Getter method.	O(1)
getDay()	Getter method.	O(1)
getTime()	Getter method.	O(1)
getCompleted()	Getter method.	O(1)
getAccurancy()	Getter method.	O(1)
ExperimentNode (Experiment)	Constructor	O(1)
setNextExperiment (Experiment)	Setter method	O(1)
setNextDay (ExperimentNode)	Setter method	O(1)
getNextExperment()	Getter method.	O(1)
getNextDay()	Getter method.	O(1)
getData()	Getter method.	O(1)
ListIterator (ExperimentNode, ExperimentList)	Constructor	O(1)
hasNext()	Just uses equality situation. O(1)	O(1)
next()	Inside of this method it uses one if statement O(1), one get method O(1) and assignments O(1).	O(1)
remove()	Inside of this method there is if else statement, O(1), if it goes with if statement the solution is O(1), if it goes with else statement there is one while loop (O(n)), and there are other if else statements,etc. which all has complexity O(1). So the complexity is the bigger one, O(n).	Best Case: Ω(1) Worst Case: O(n)
ExperimentList()	Constructor	O(1)
ExperimentList (ExperimentNode)	Constructor	O(1)
getHead()	Getter method.	O(1)

setHead()	Setter method.	O(1)
iterator()	It just creates new IteratorList class and returns it. Creating class O(1), returning O(1).	O(1)
addExp()	First if program goes with first the if statement and the else if statement which comes after that, the complexiy is become Ω(1). So the best case is Ω(1). If program goes with third else if stement it must go inside of while statements, which has O(n) complexity. If program goes with third else if statement, the first while loop time complexiy is O(n) which n is the day number which we want to reach. After that while loop we have one if statment which has O(1) complexity. After that if statement there is another while loop which time complexity is O(n) which n is the experiment which we want to reach. So in this second outer else if statement we obtain O(n) + O(n) =O(n) time complexity. If program goes with else statement we have another if-else if statement, if statement has one while loop which has O(n) complexity, which n is the day number which we want to reach, the else if statement has one loop which is the same as above loop. So in this else statement we obtain O(n) + O(n) = O(n) time complexity.	Best Case: Ω(1) Worst Case: O(n)
	First if program goes with if and else if statement the complexity is Ω(1) which is the best case for this method.	

getExperiment()	If program goes with second if statement, we have one while loop which has O(n) complexity which n is the number of given index.
	
	If program goes with third else if statement we have one while loop which has O(n) complexity which n is the day number which we want to reach. After that there is another if-else if-else statement, the if statement has O(1) complexity and the else if statement has one while loop which has O(n) complexity which n is the number of index, and the else statement has O(1) complexity. So this parts compexity is O(n).
	
	The whole method complexity is; O(n) , Ω(n)
		Best Case:
		Ω(1)
		
		Worst Case:
		O(1)

setExp()	First if program goes with if statemenr and second else if statement it has Ω(1) complexity which is the best case for this method.
	
	The second else if statement has another if else statement inside of it and if statement has O(1) complexity and else statement has one while loop inside of it and it has O(n) complexity which n is the number of index which we want to reach. So this parts complexity is O(n).
	
	The third else if statement has while loop which has O(n) complexity which n is te day we want to reach and we have if elseif else statement. İf statement has O(1) complexity,
		Best Case:
		Ω(1)
		
		Worst Case:
		O(1)

	else if statement has another if else statement and in this if else statement, if has one while loop, O(n) complexity which n is the day we want to reach, and else statement has one while loop also have O(n) complexity which n is the index number. The outer else statement has O(1) complexity. So this method has O(n) complexity.	
removeExp()	In this method we have if elseif else statements, as same as above method the first if, andfirst else if methods has O(1) complexity which is the best case for this method. The second else if statement has another if else statements which if has O(1) complety and else has while loop inside of it which has O(n) complexity which n is the index number.So finally it has O(n) complexity. The third else if statement has one while loop which it has O(n) complexity which n is the number of day which we want to reach. After while loop it has another if elseif else statement, if has O(1) complexity, and else if has another if else statement which as summary all has O(n) complexity. And we have else statement which it has O(n) complexity which n is the number of index. To summary this method has O(n) complexity.	Best Case:Ω(1) Worst Case: O(1)

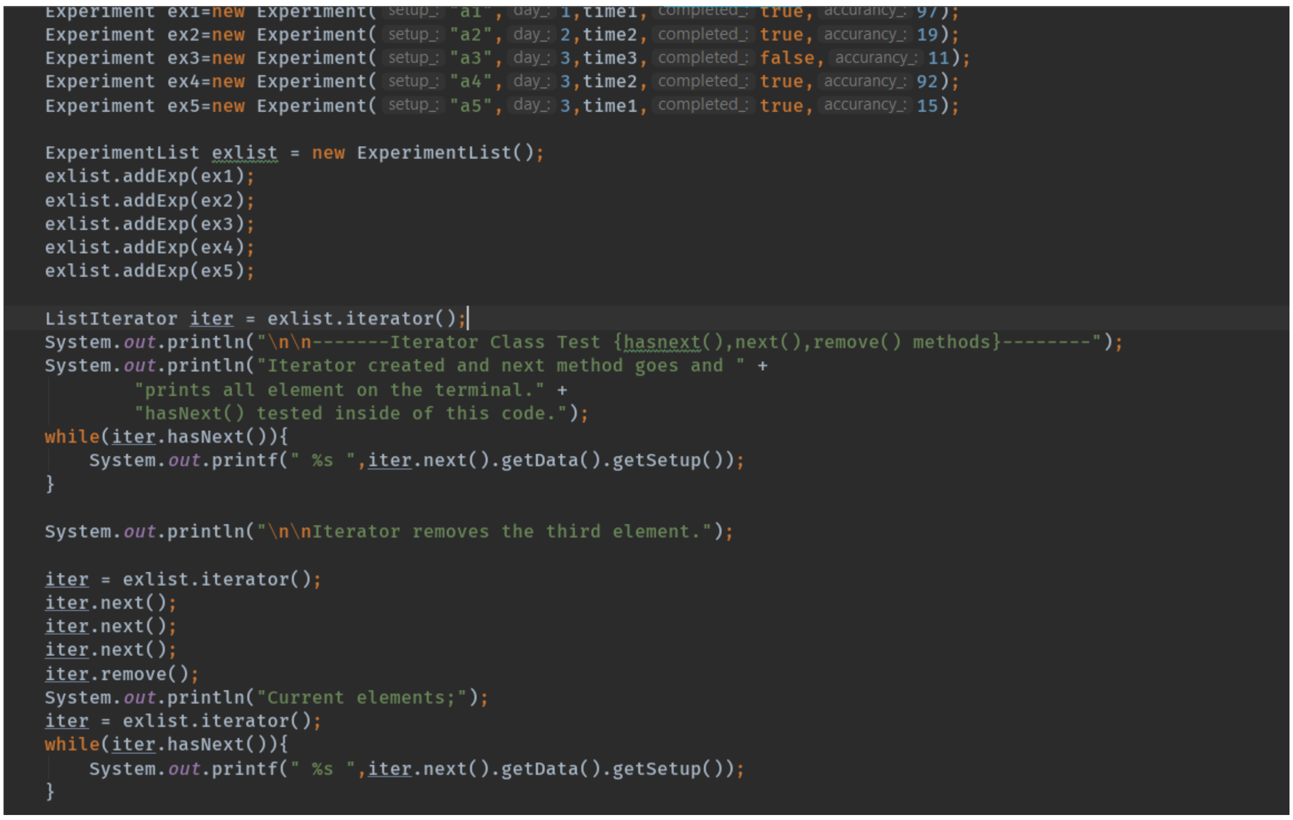
listExperiment()	This method has one while loop which it has O(n) time complexity which n is the day which we want to reach, and after that it has if else statement which if has O(1) complexity and else has another while loop which it has O(n) complexity which n is the number of experiment. So this method has O(n) complexity.	O(n)
removeDay()	This method has if else statement which if has Ω(1) complexity which is the best case fort his method and the else statement has while loop which has O(n) complexity which n is the number of experiment which we want to reach.	Best Case: Ω(1) Worst Case: O(n)
All Merge Sort Methods	Merge sort.	Has Θ(n*log(n)) All three cases (worst,best,average)
orderDay()	It has if statements which time complexity is O(1) and below that there is another if else statement which if has two while loop which both has O(n) complexity(one n is for experiment number we want to reach for storing tail information,etc. ,the other one sorted experiment number.). So this if part has O(n) complexity. The else part has one while loop which has O(n) complexity which n is the number of day and below that we have another if else statement which if has O(1) complexity and else has 3 while loops which all has O(n) complexity which two for experiments, one for sorted list experiments. In addition to all this else statement uses merge sort algroithm so it has O(n*logn)	Best Case: Ω(1) Worst Case: O(n*log(n))

	complexity. So finally we have O(n) + O(nlogn) = O(nlogn) time complexity.	
orderExperiments()	This method has one while loop which has O(n) complexity and it uses merge sort algorithm which has O(nlogn) complexity so the whole method has O(n*log(n)) complexity.	O(n*log(n))

3 RESULT

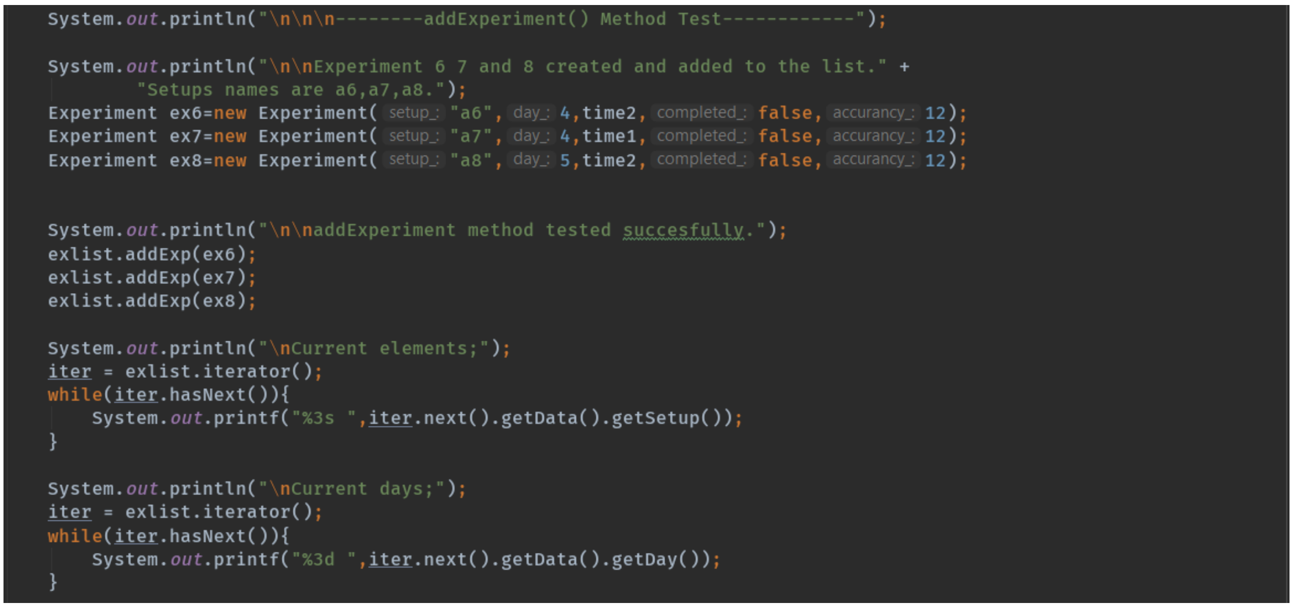
3.1 Test Cases

*****ListIIterator class next(),hasnext(),remove() methods test.*****



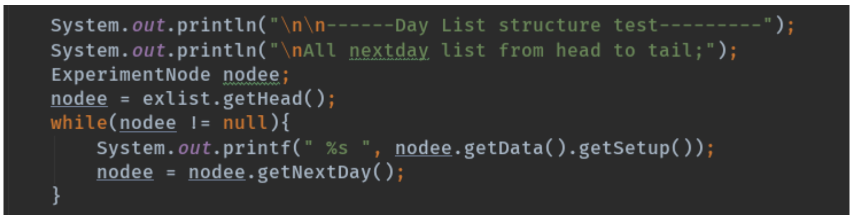
```
Experiment ex1=new Experiment( setup_: "a1", day_: 1,time1, completed_: true, accurancy_: 97);
Experiment ex2=new Experiment( setup_: "a2", day_: 2,time2, completed_: true, accurancy_: 19);
Experiment ex3=new Experiment( setup_: "a3", day_: 3,time3, completed_: false, accurancy_: 11);
Experiment ex4=new Experiment( setup_: "a4", day_: 3,time2, completed_: true, accurancy_: 92);
Experiment ex5=new Experiment( setup_: "a5", day_: 3,time1, completed_: true, accurancy_: 15);

ExperimentList exlist = new ExperimentList();
exlist.addExp(ex1);
exlist.addExp(ex2);
exlist.addExp(ex3);
exlist.addExp(ex4);
exlist.addExp(ex5);

ListIterator iter = exlist.iterator();
System.out.println("\n\n-------Iterator Class Test {hasnext(),next(),remove() methods}--------");
System.out.println("Iterator created and next method goes and " +
        "prints all element on the terminal." +
        "hasNext() tested inside of this code.");
while(iter.hasNext()){
    System.out.printf(" %s ",iter.next().getData().getSetup());
}

System.out.println("\n\nIterator removes the third element.");

iter = exlist.iterator();
iter.next();
iter.next();
iter.next();
iter.remove();
System.out.println("Current elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf(" %s ",iter.next().getData().getSetup());
}
```

*****addExperiment() method test.*****



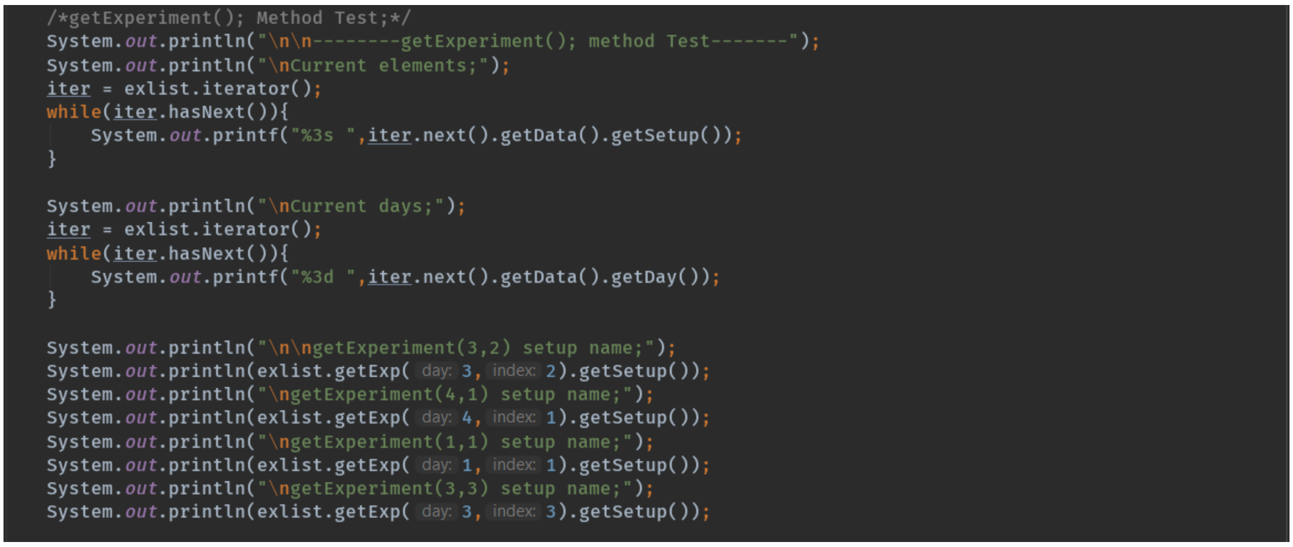
```java
System.out.println("\n\n\n--------addExperiment() Method Test------------");

System.out.println("\n\nExperiment 6 7 and 8 created and added to the list." +
        "Setups names are a6,a7,a8.");
Experiment ex6=new Experiment( setup_: "a6", day_: 4,time2, completed_: false, accurancy_: 12);
Experiment ex7=new Experiment( setup_: "a7", day_: 4,time1, completed_: false, accurancy_: 12);
Experiment ex8=new Experiment( setup_: "a8", day_: 5,time2, completed_: false, accurancy_: 12);


System.out.println("\n\naddExperiment method tested succesfully.");
exlist.addExp(ex6);
exlist.addExp(ex7);
exlist.addExp(ex8);

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}
```

*****Day list structure test.*****



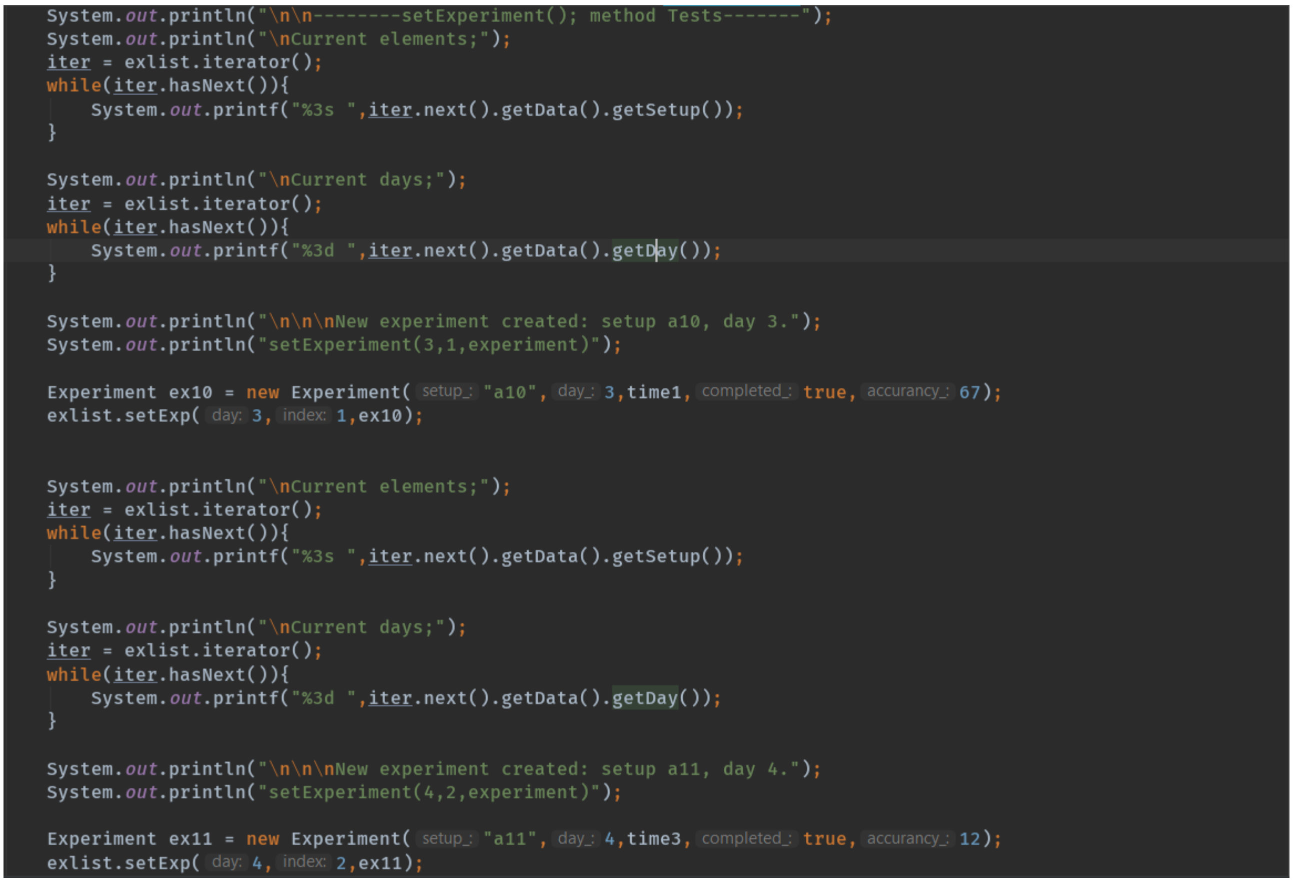
```java
System.out.println("\n\n------Day List structure test---------");
System.out.println("\nAll nextday list from head to tail;");
ExperimentNode nodee;
nodee = exlist.getHead();
while(nodee != null){
    System.out.printf(" %s ", nodee.getData().getSetup());
    nodee = nodee.getNextDay();
}
```

*****getExp() method test.*****

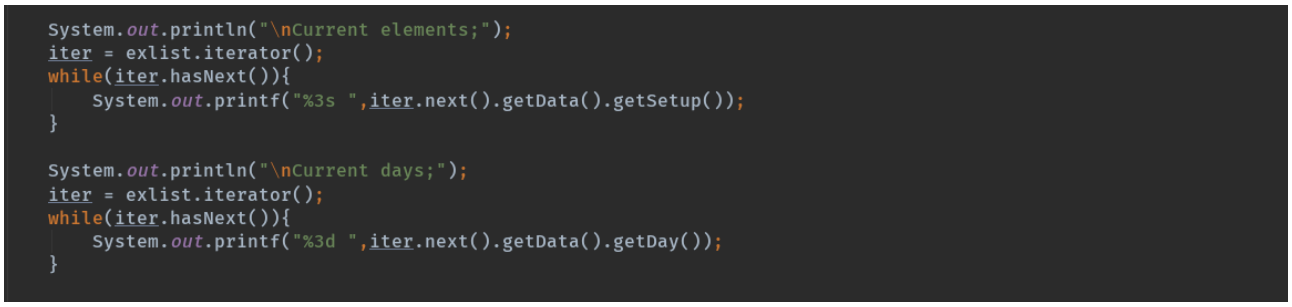


```java
/*getExperiment(); Method Test;*/
System.out.println("\n\n--------getExperiment(); method Test-------");
System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}

System.out.println("\n\ngetExperiment(3,2) setup name;");
System.out.println(exlist.getExp( day: 3, index: 2).getSetup());
System.out.println("\ngetExperiment(4,1) setup name;");
System.out.println(exlist.getExp( day: 4, index: 1).getSetup());
System.out.println("\ngetExperiment(1,1) setup name;");
System.out.println(exlist.getExp( day: 1, index: 1).getSetup());
System.out.println("\ngetExperiment(3,3) setup name;");
System.out.println(exlist.getExp( day: 3, index: 3).getSetup());
```

*****setExp() method test.*****

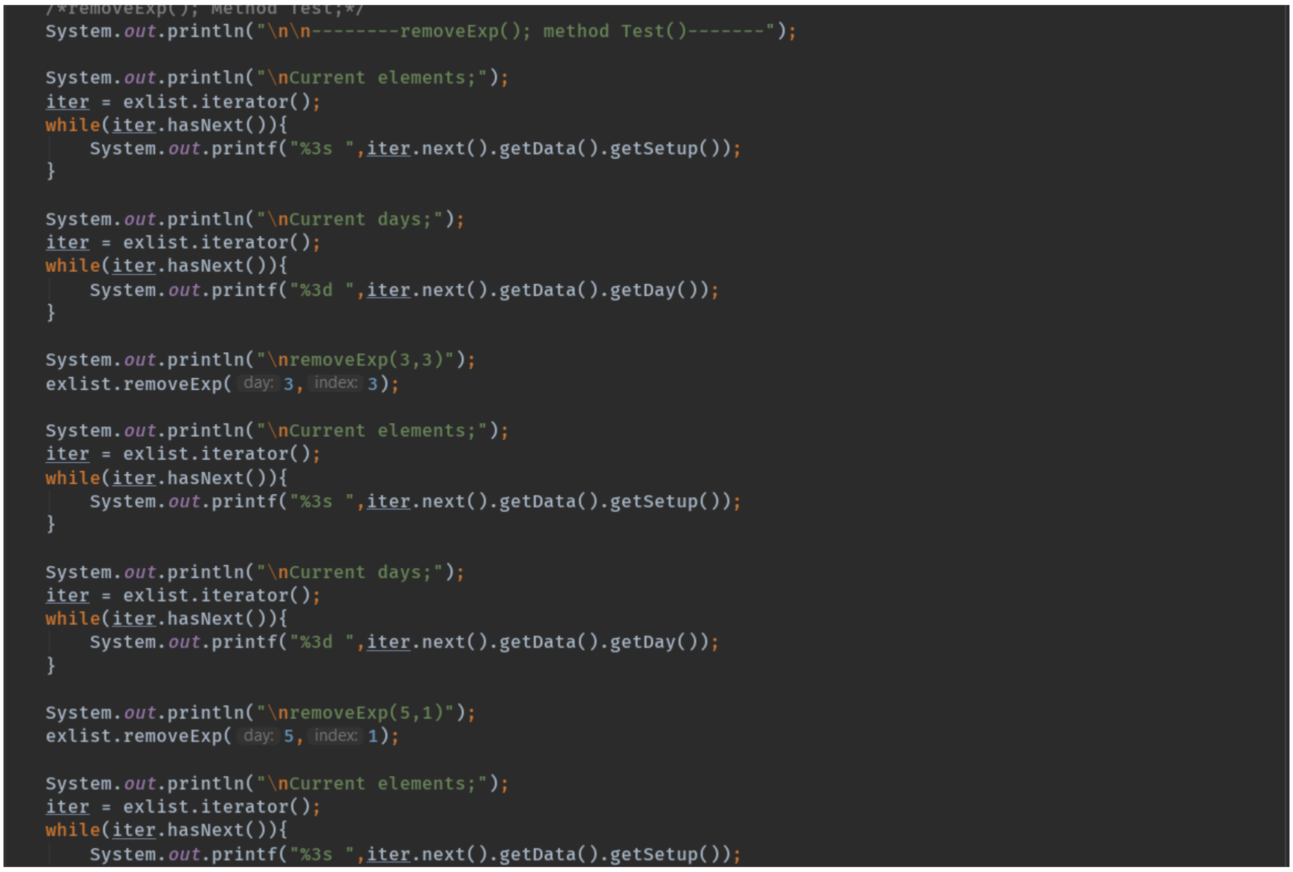


```java
System.out.println("\n\n--------setExperiment(); method Tests-------");
System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}

System.out.println("\n\n\nNew experiment created: setup a10, day 3.");
System.out.println("setExperiment(3,1,experiment)");

Experiment ex10 = new Experiment( setup_: "a10", day_: 3,time1, completed_: true, accurancy_: 67);
exlist.setExp( day: 3, index: 1,ex10);


System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}

System.out.println("\n\n\nNew experiment created: setup a11, day 4.");
System.out.println("setExperiment(4,2,experiment)");

Experiment ex11 = new Experiment( setup_: "a11", day_: 4,time3, completed_: true, accurancy_: 12);
exlist.setExp( day: 4, index: 2,ex11);
```

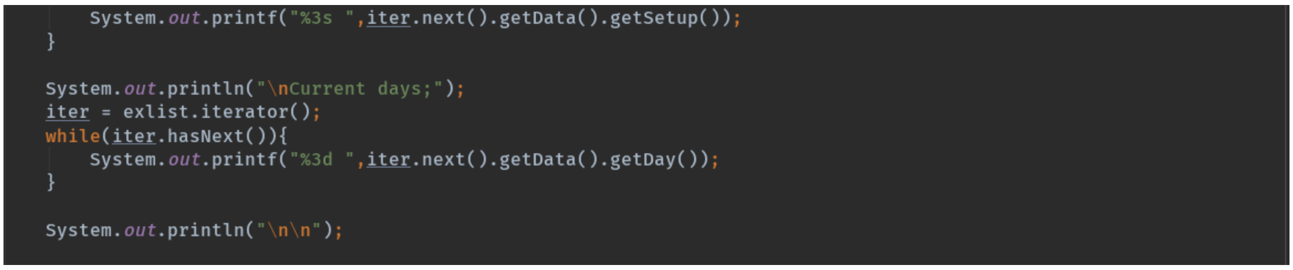


```java
System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}
```

*****removeExp() method test.*****

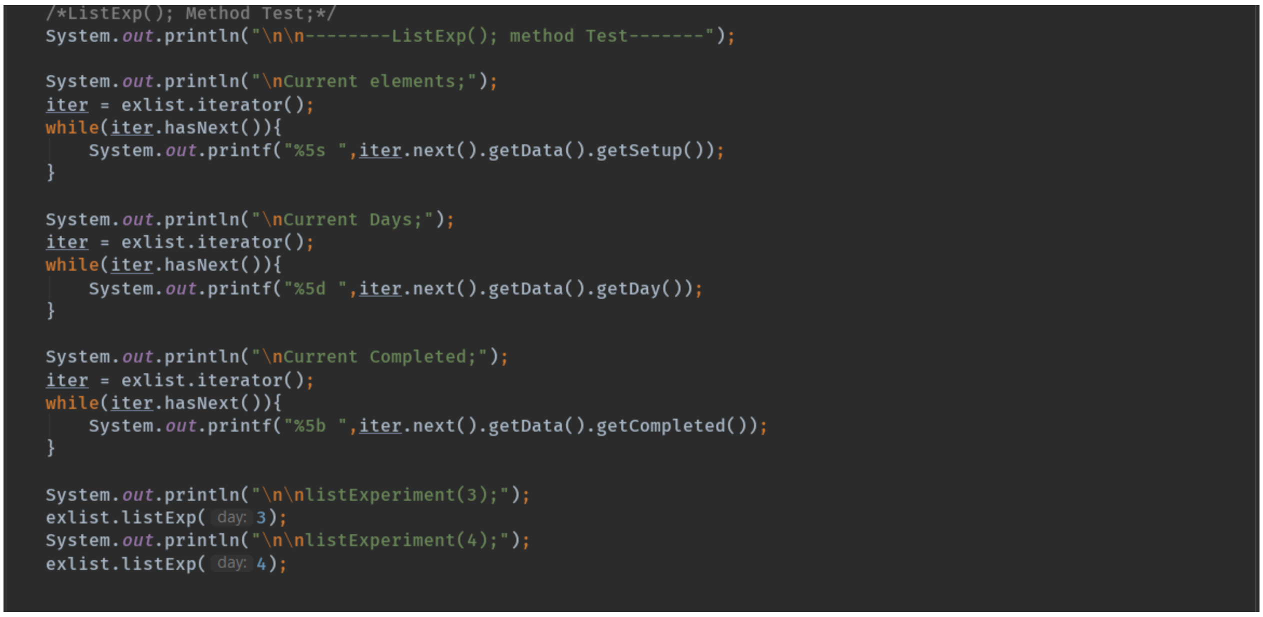


```
/*removeExp(); Method Test;*/
System.out.println("\n\n--------removeExp(); method Test()-------");

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}

System.out.println("\nremoveExp(3,3)");
exlist.removeExp( day: 3, index: 3);

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}

System.out.println("\nremoveExp(5,1)");
exlist.removeExp( day: 5, index: 1);

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
```



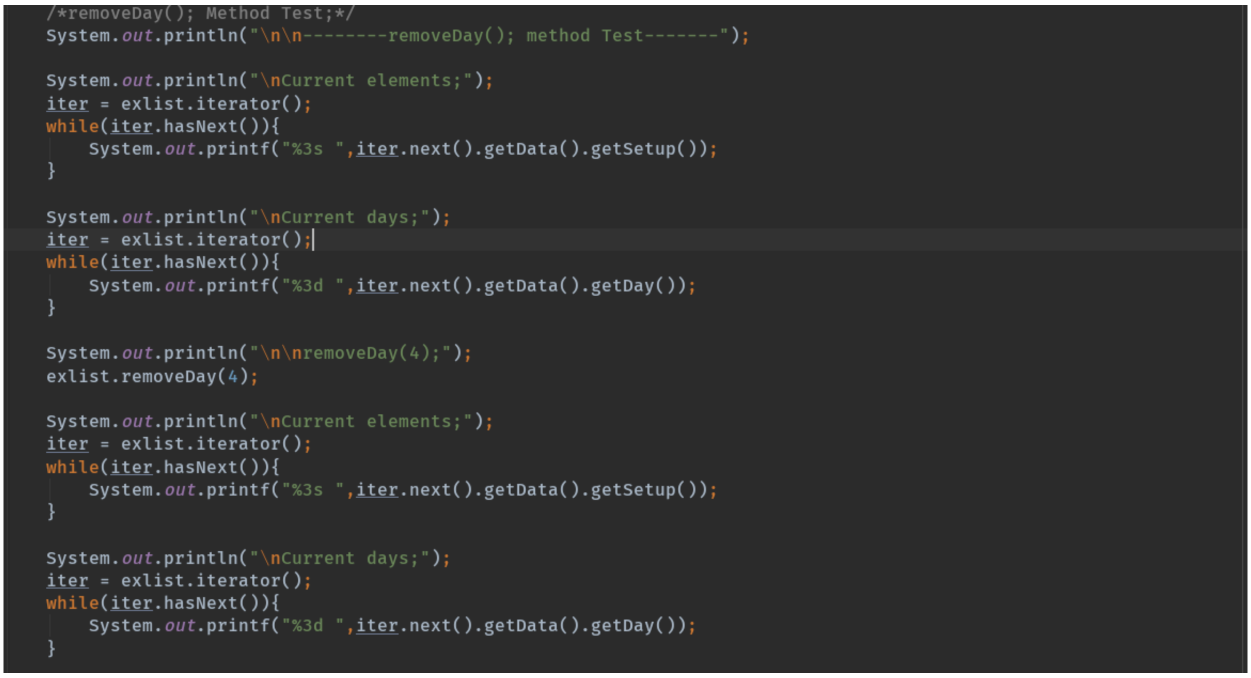
```
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}

System.out.println("\n\n");
```

*****ListExp() method test.*****

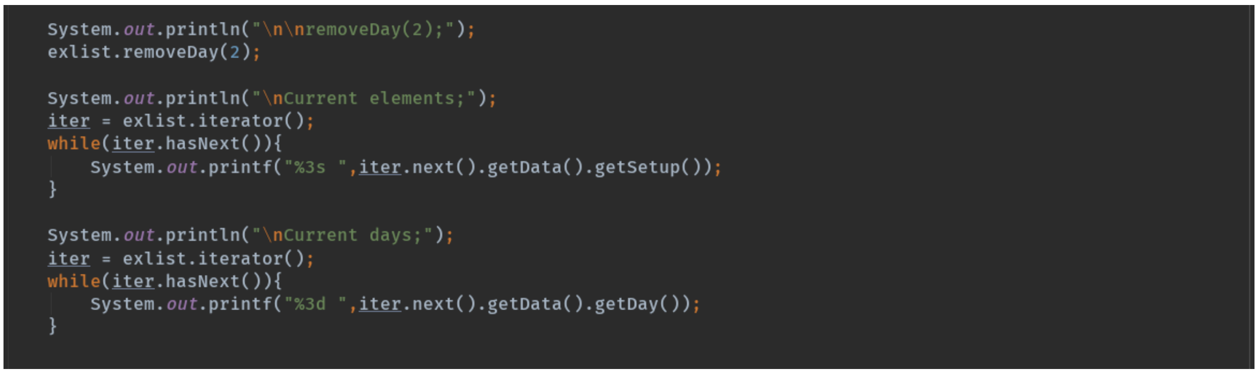


```
/*ListExp(); Method Test;*/
System.out.println("\n\n--------ListExp(); method Test-------");

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent Days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5d ",iter.next().getData().getDay());
}

System.out.println("\nCurrent Completed;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5b ",iter.next().getData().getCompleted());
}

System.out.println("\n\nlistExperiment(3);");
exlist.listExp( day: 3);
System.out.println("\n\nlistExperiment(4);");
exlist.listExp( day: 4);
```

*****removeDay() method test.*****

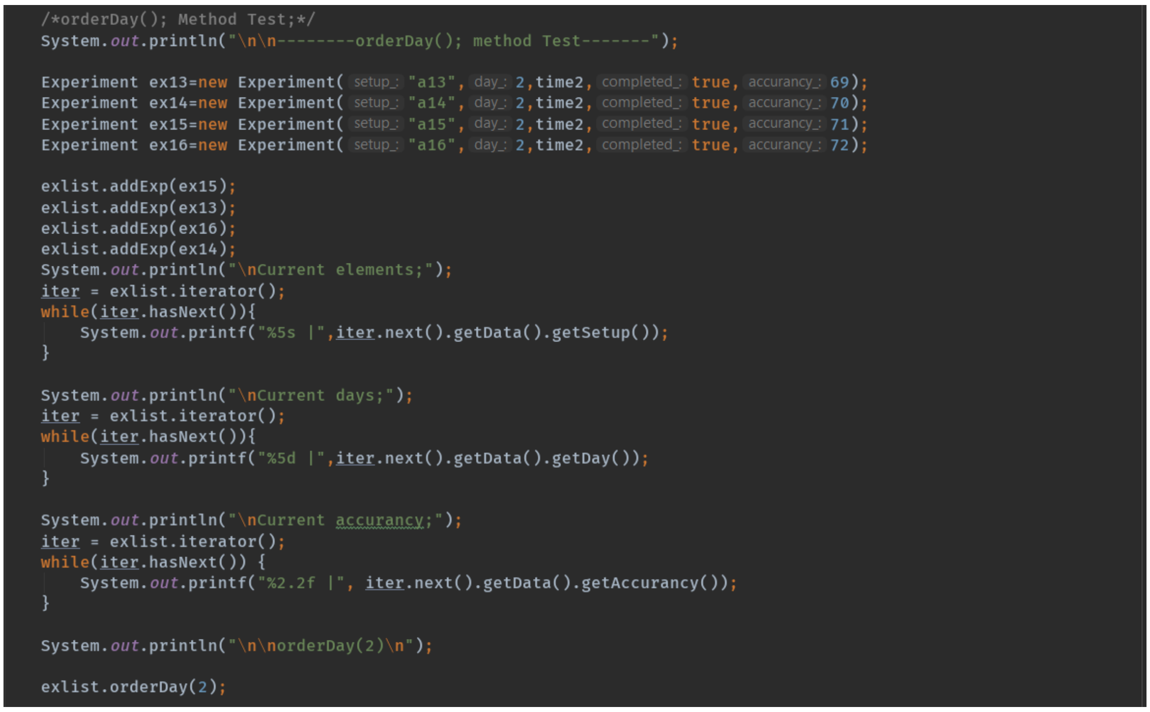


```
/*removeDay(); Method Test;*/
System.out.println("\n\n--------removeDay(); method Test-------");

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}

System.out.println("\n\nremoveDay(4);");
exlist.removeDay(4);

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}
```

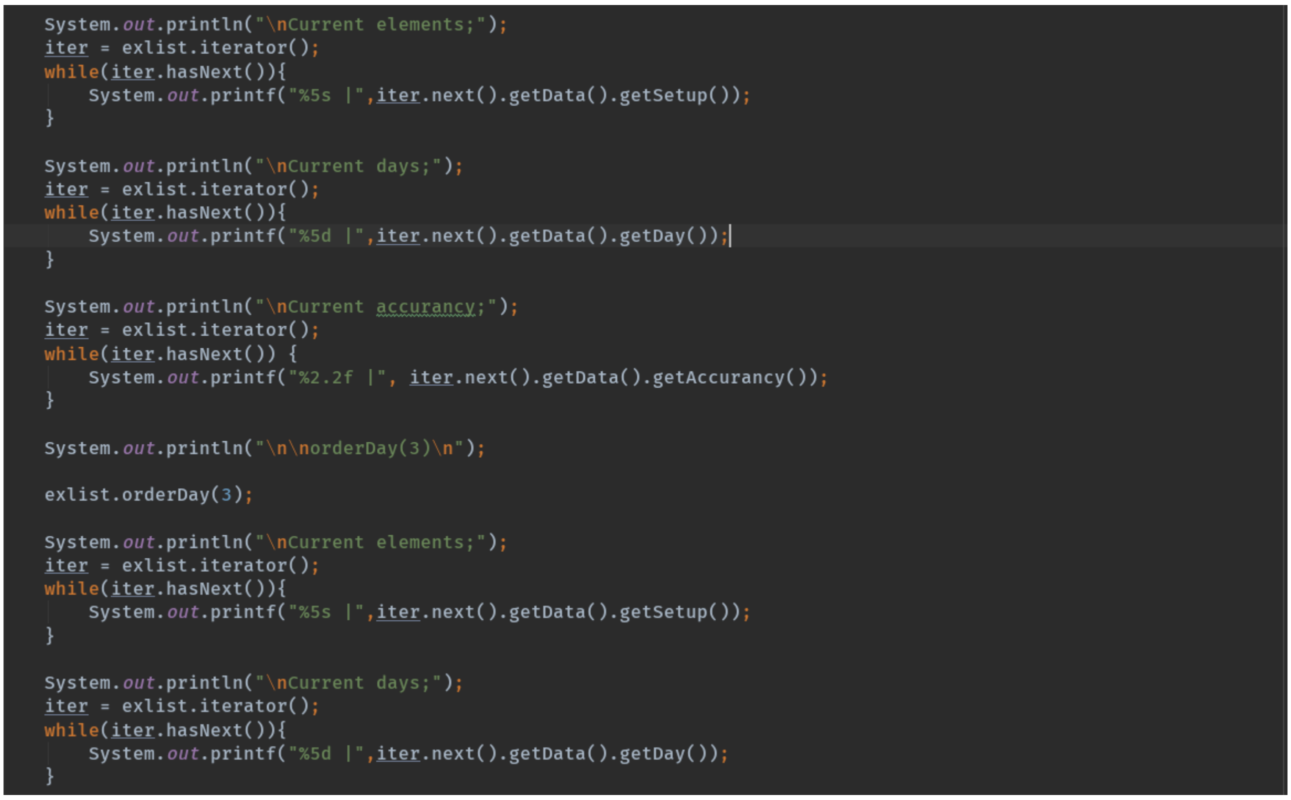


```
System.out.println("\n\nremoveDay(2);");
exlist.removeDay(2);

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3s ",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%3d ",iter.next().getData().getDay());
}
```

*****orderDay() method test.*****

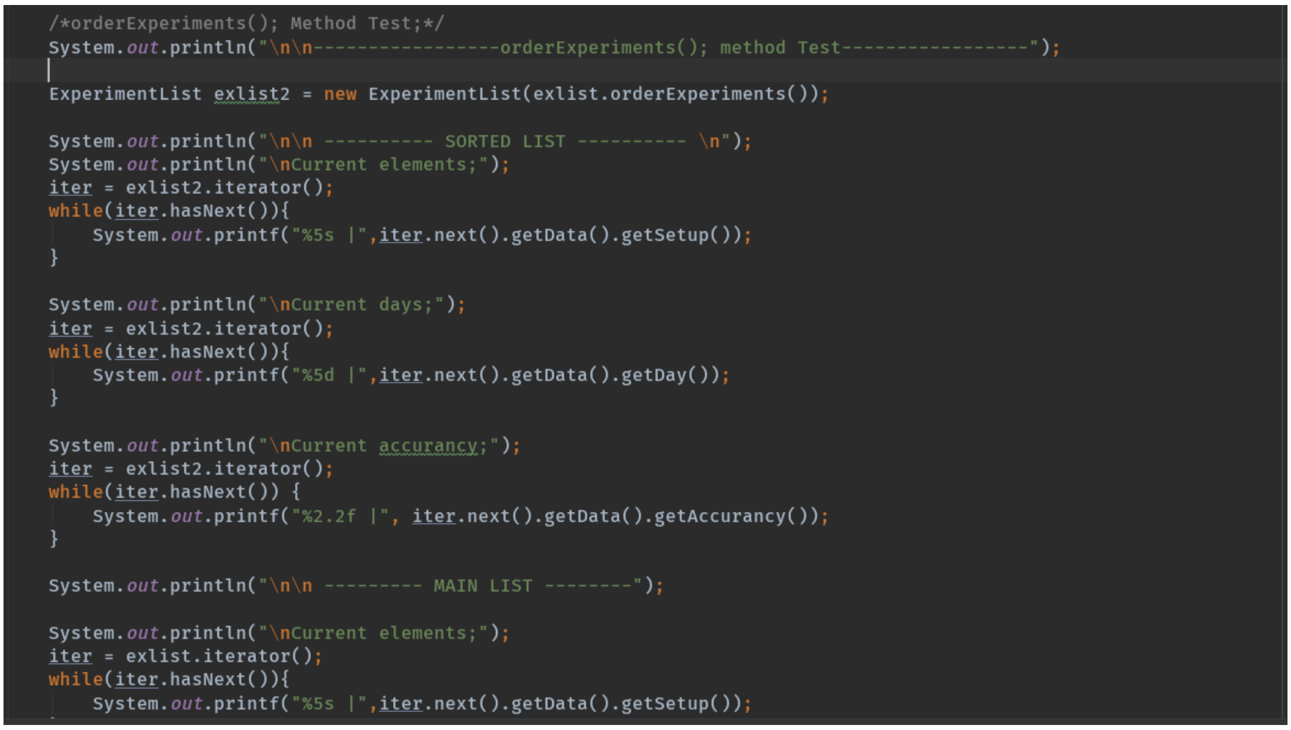


```
/*orderDay(); Method Test;*/
System.out.println("\n\n--------orderDay(); method Test-------");

Experiment ex13=new Experiment( setup: "a13", day: 2,time2, completed: true, accurancy: 69);
Experiment ex14=new Experiment( setup: "a14", day: 2,time2, completed: true, accurancy: 70);
Experiment ex15=new Experiment( setup: "a15", day: 2,time2, completed: true, accurancy: 71);
Experiment ex16=new Experiment( setup: "a16", day: 2,time2, completed: true, accurancy: 72);

exlist.addExp(ex15);
exlist.addExp(ex13);
exlist.addExp(ex16);
exlist.addExp(ex14);
System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5s |",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5d |",iter.next().getData().getDay());
}

System.out.println("\nCurrent accurancy;");
iter = exlist.iterator();
while(iter.hasNext()) {
    System.out.printf("%2.2f |", iter.next().getData().getAccurancy());
}

System.out.println("\n\norderDay(2)\n");

exlist.orderDay(2);
```



```java
System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5s |",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5d |",iter.next().getData().getDay());
}

System.out.println("\nCurrent accurancy;");
iter = exlist.iterator();
while(iter.hasNext()) {
    System.out.printf("%2.2f |", iter.next().getData().getAccurancy());
}

System.out.println("\n\norderDay(3)\n");

exlist.orderDay(3);

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5s |",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5d |",iter.next().getData().getDay());
}
```

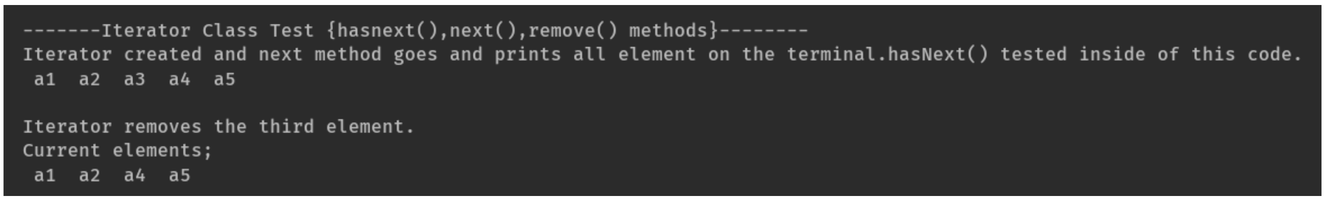
*****OrderExpriments() method test.*****



```java
/*orderExperiments(); Method Test;*/
System.out.println("\n\n-----------------orderExperiments(); method Test-----------------");

ExperimentList exlist2 = new ExperimentList(exlist.orderExperiments());

System.out.println("\n\n ---------- SORTED LIST ---------- \n");
System.out.println("\nCurrent elements;");
iter = exlist2.iterator();
while(iter.hasNext()){
    System.out.printf("%5s |",iter.next().getData().getSetup());
}

System.out.println("\nCurrent days;");
iter = exlist2.iterator();
while(iter.hasNext()){
    System.out.printf("%5d |",iter.next().getData().getDay());
}

System.out.println("\nCurrent accurancy;");
iter = exlist2.iterator();
while(iter.hasNext()) {
    System.out.printf("%2.2f |", iter.next().getData().getAccurancy());
}

System.out.println("\n\n --------- MAIN LIST --------");

System.out.println("\nCurrent elements;");
iter = exlist.iterator();
while(iter.hasNext()){
    System.out.printf("%5s |",iter.next().getData().getSetup());
```

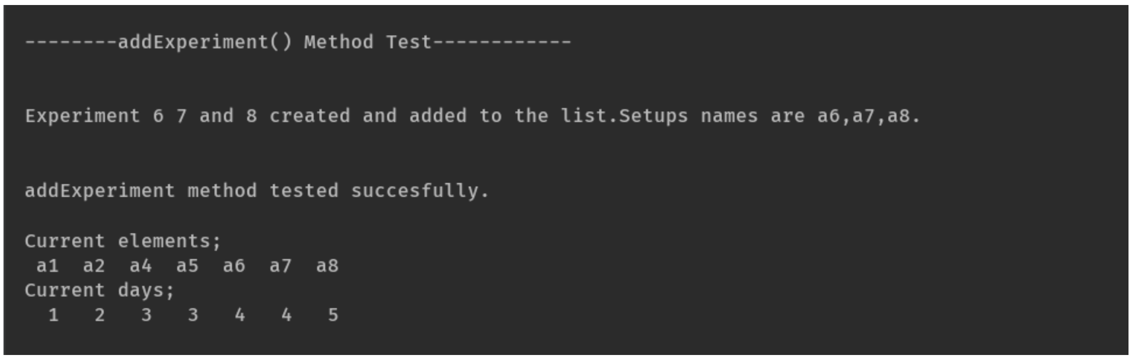
3.2 Running Results

*****ListIIterator class next(),hasnext(),remove() methods test result.*****



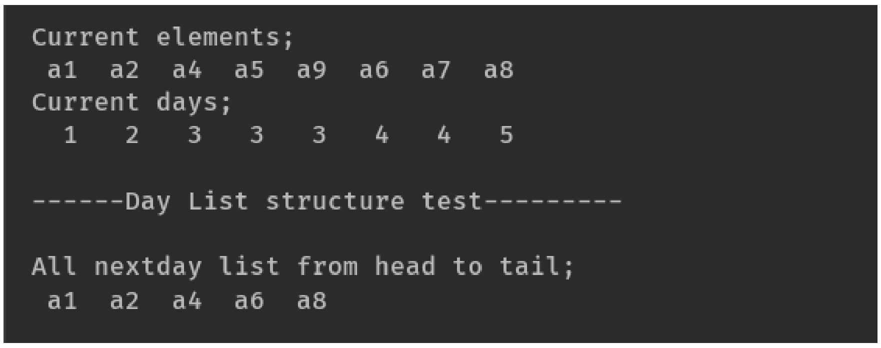
```
-------Iterator Class Test {hasnext(),next(),remove() methods}--------
Iterator created and next method goes and prints all element on the terminal.hasNext() tested inside of this code.
 a1  a2  a3  a4  a5

Iterator removes the third element.
Current elements;
 a1  a2  a4  a5
```

*****addExperiment() method test result.*****



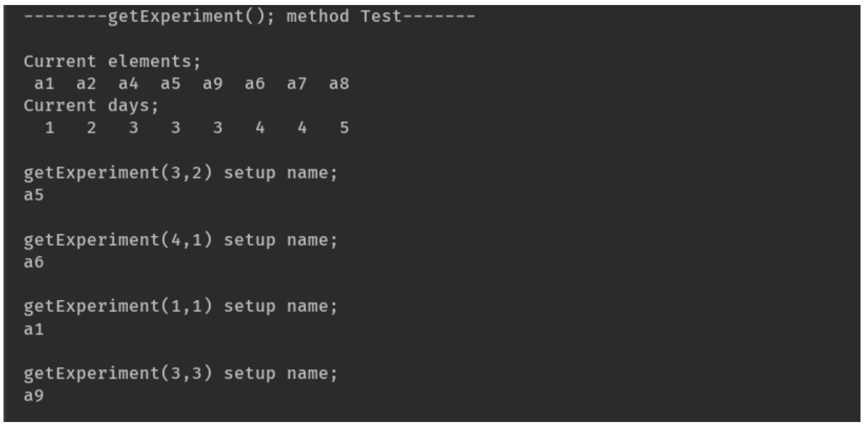
```
--------addExperiment() Method Test------------


Experiment 6 7 and 8 created and added to the list.Setups names are a6,a7,a8.


addExperiment method tested succesfully.

Current elements;
 a1  a2  a4  a5  a6  a7  a8
Current days;
  1   2   3   3   4   4   5
```

*****Day list structure test result.*****



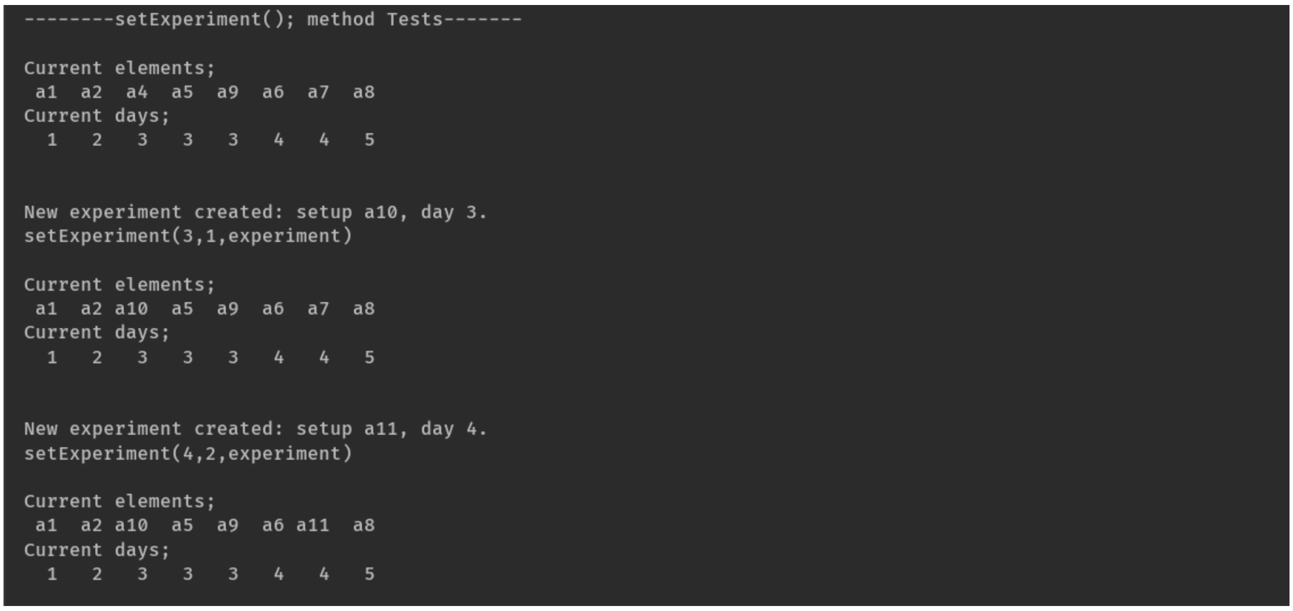
```
Current elements;
 a1  a2  a4  a5  a9  a6  a7  a8
Current days;
  1   2   3   3   3   4   4   5

------Day List structure test---------

All nextday list from head to tail;
 a1  a2  a4  a6  a8
```

*****getExp() method test result.*****



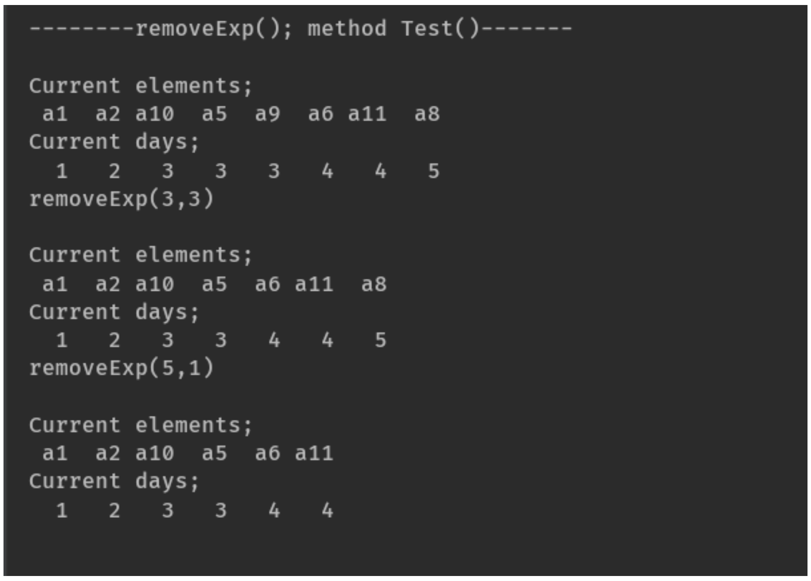
```
--------getExperiment(); method Test-------

Current elements;
 a1  a2  a4  a5  a9  a6  a7  a8
Current days;
  1   2   3   3   3   4   4   5

getExperiment(3,2) setup name;
a5

getExperiment(4,1) setup name;
a6

getExperiment(1,1) setup name;
a1

getExperiment(3,3) setup name;
a9
```

*****setExp() method test test.*****



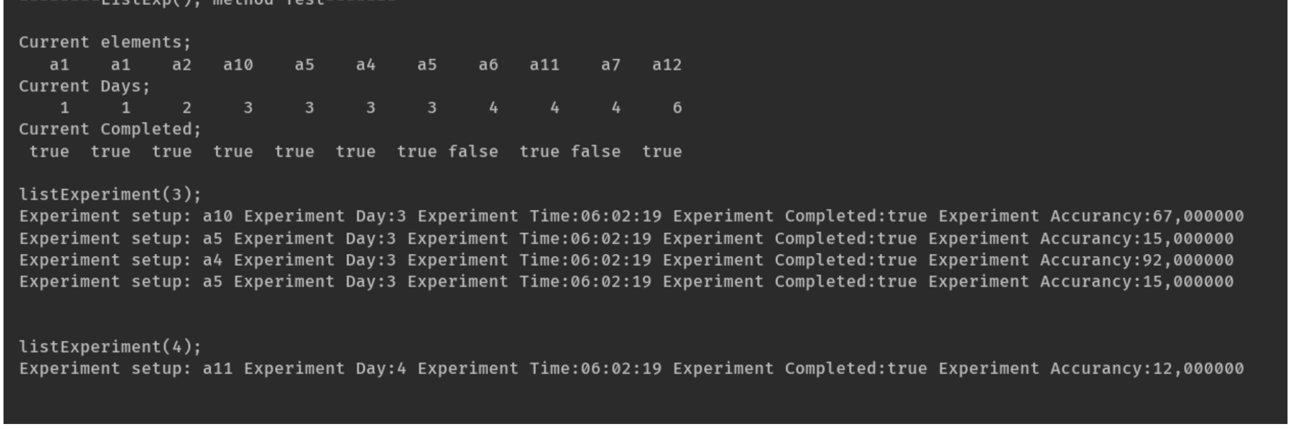
```
--------setExperiment(); method Tests-------

Current elements;
 a1  a2  a4  a5  a9  a6  a7  a8
Current days;
  1   2   3   3   3   4   4   5


New experiment created: setup a10, day 3.
setExperiment(3,1,experiment)

Current elements;
 a1  a2 a10  a5  a9  a6  a7  a8
Current days;
  1   2   3   3   3   4   4   5


New experiment created: setup a11, day 4.
setExperiment(4,2,experiment)

Current elements;
 a1  a2 a10  a5  a9  a6 a11  a8
Current days;
  1   2   3   3   3   4   4   5
```

*****removeExp() method test result.*****



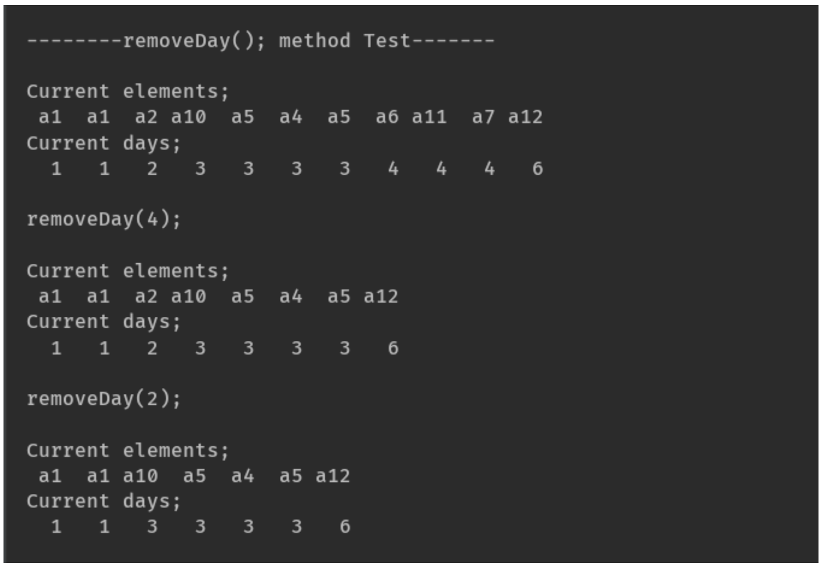
```
--------removeExp(); method Test()-------

Current elements;
 a1  a2 a10  a5  a9  a6 a11  a8
Current days;
  1   2   3   3   3   4   4   5
removeExp(3,3)

Current elements;
 a1  a2 a10  a5  a6 a11  a8
Current days;
  1   2   3   3   4   4   5
removeExp(5,1)

Current elements;
 a1  a2 a10  a5  a6 a11
Current days;
  1   2   3   3   4   4
```

*****ListExp() method test result.*****



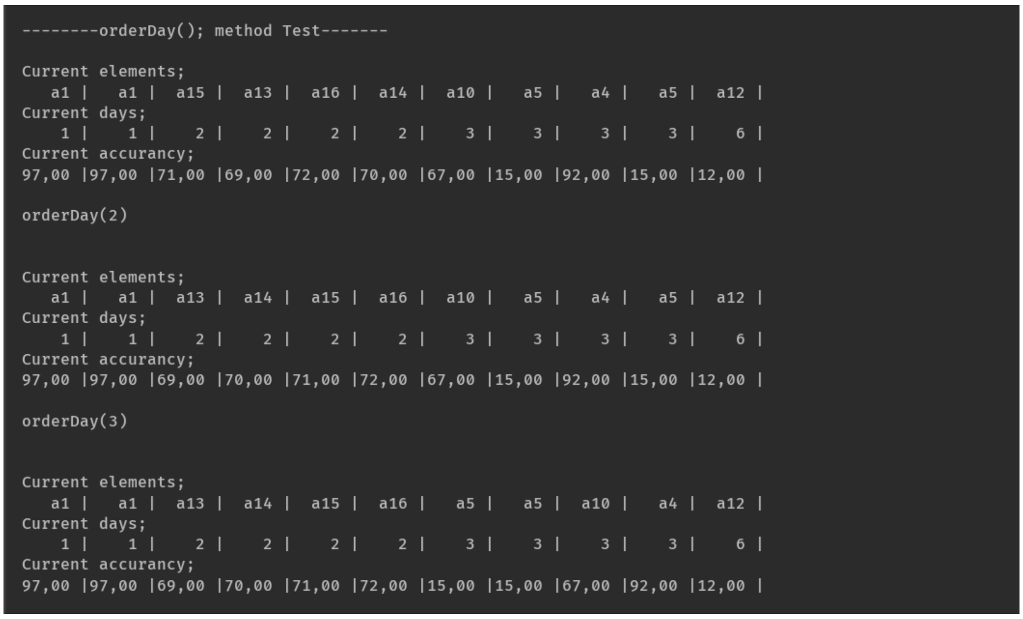
```
--------ListExp(); method Test-------

Current elements;
   a1    a1    a2   a10    a5    a4    a5    a6   a11    a7   a12
Current Days;
    1     1     2     3     3     3     3     4     4     4     6
Current Completed;
 true  true  true  true  true  true  true false  true false  true

listExperiment(3);
Experiment setup: a10 Experiment Day:3 Experiment Time:06:02:19 Experiment Completed:true Experiment Accurancy:67,000000
Experiment setup: a5 Experiment Day:3 Experiment Time:06:02:19 Experiment Completed:true Experiment Accurancy:15,000000
Experiment setup: a4 Experiment Day:3 Experiment Time:06:02:19 Experiment Completed:true Experiment Accurancy:92,000000
Experiment setup: a5 Experiment Day:3 Experiment Time:06:02:19 Experiment Completed:true Experiment Accurancy:15,000000


listExperiment(4);
Experiment setup: a11 Experiment Day:4 Experiment Time:06:02:19 Experiment Completed:true Experiment Accurancy:12,000000
```

*****removeDay() method test.*****



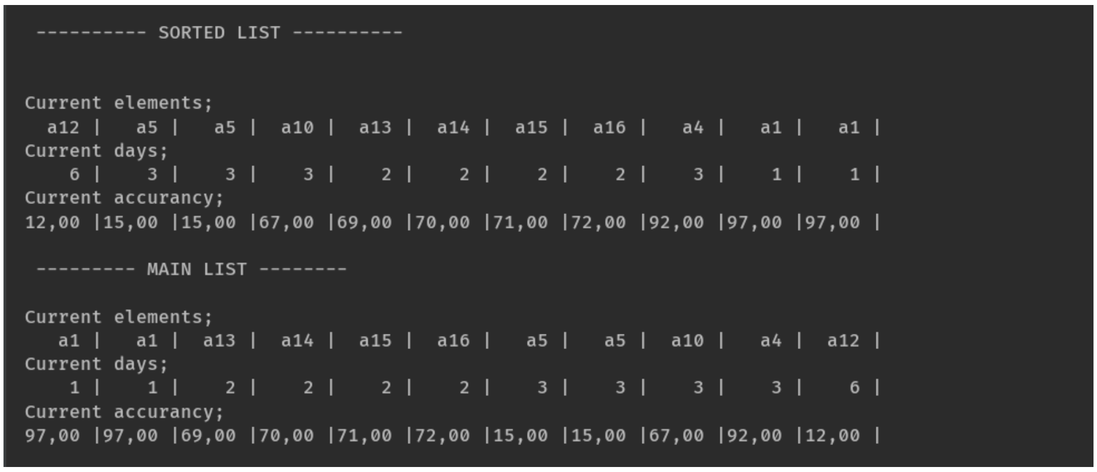
```
--------removeDay(); method Test-------

Current elements;
 a1  a1  a2 a10  a5  a4  a5  a6 a11  a7 a12
Current days;
  1   1   2   3   3   3   3   4   4   4   6

removeDay(4);

Current elements;
 a1  a1  a2 a10  a5  a4  a5 a12
Current days;
  1   1   2   3   3   3   3   6

removeDay(2);

Current elements;
 a1  a1 a10  a5  a4  a5 a12
Current days;
  1   1   3   3   3   3   6
```

*****orderDay() method test.*****



```
--------orderDay(); method Test-------

Current elements;
   a1 |   a1 |  a15 |  a13 |  a16 |  a14 |  a10 |   a5 |   a4 |   a5 |  a12 |
Current days;
    1 |    1 |    2 |    2 |    2 |    2 |    3 |    3 |    3 |    3 |    6 |
Current accurancy;
97,00 |97,00 |71,00 |69,00 |72,00 |70,00 |67,00 |15,00 |92,00 |15,00 |12,00 |

orderDay(2)


Current elements;
   a1 |   a1 |  a13 |  a14 |  a15 |  a16 |  a10 |   a5 |   a4 |   a5 |  a12 |
Current days;
    1 |    1 |    2 |    2 |    2 |    2 |    3 |    3 |    3 |    3 |    6 |
Current accurancy;
97,00 |97,00 |69,00 |70,00 |71,00 |72,00 |67,00 |15,00 |92,00 |15,00 |12,00 |

orderDay(3)


Current elements;
   a1 |   a1 |  a13 |  a14 |  a15 |  a16 |   a5 |   a5 |  a10 |   a4 |  a12 |
Current days;
    1 |    1 |    2 |    2 |    2 |    2 |    3 |    3 |    3 |    3 |    6 |
Current accurancy;
97,00 |97,00 |69,00 |70,00 |71,00 |72,00 |15,00 |15,00 |67,00 |92,00 |12,00 |
```

*****orderExperiment() method test.*****



```
 ---------- SORTED LIST ----------


Current elements;
  a12 |   a5 |   a5 |  a10 |  a13 |  a14 |  a15 |  a16 |   a4 |   a1 |   a1 |
Current days;
    6 |    3 |    3 |    3 |    2 |    2 |    2 |    2 |    3 |    1 |    1 |
Current accurancy;
12,00 |15,00 |15,00 |67,00 |69,00 |70,00 |71,00 |72,00 |92,00 |97,00 |97,00 |

 --------- MAIN LIST --------

Current elements;
   a1 |   a1 |  a13 |  a14 |  a15 |  a16 |   a5 |   a5 |  a10 |   a4 |  a12 |
Current days;
    1 |    1 |    2 |    2 |    2 |    2 |    3 |    3 |    3 |    3 |    6 |
Current accurancy;
97,00 |97,00 |69,00 |70,00 |71,00 |72,00 |15,00 |15,00 |67,00 |92,00 |12,00 |
```