Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 1 REPORT

GÖKÇE NUR ERER
171044079

Course Assistant: Ayşe Şerbetçi Turan

1 INTRODUCTION

1.1 Problem Definition

Problem is to create a data structure like a single linked list to keep some experiments. Experiments contain day, setup, time, completed status and accuracy informations. Each experiment should follow the other in the list. Also the first experiments of each day should point to each other.

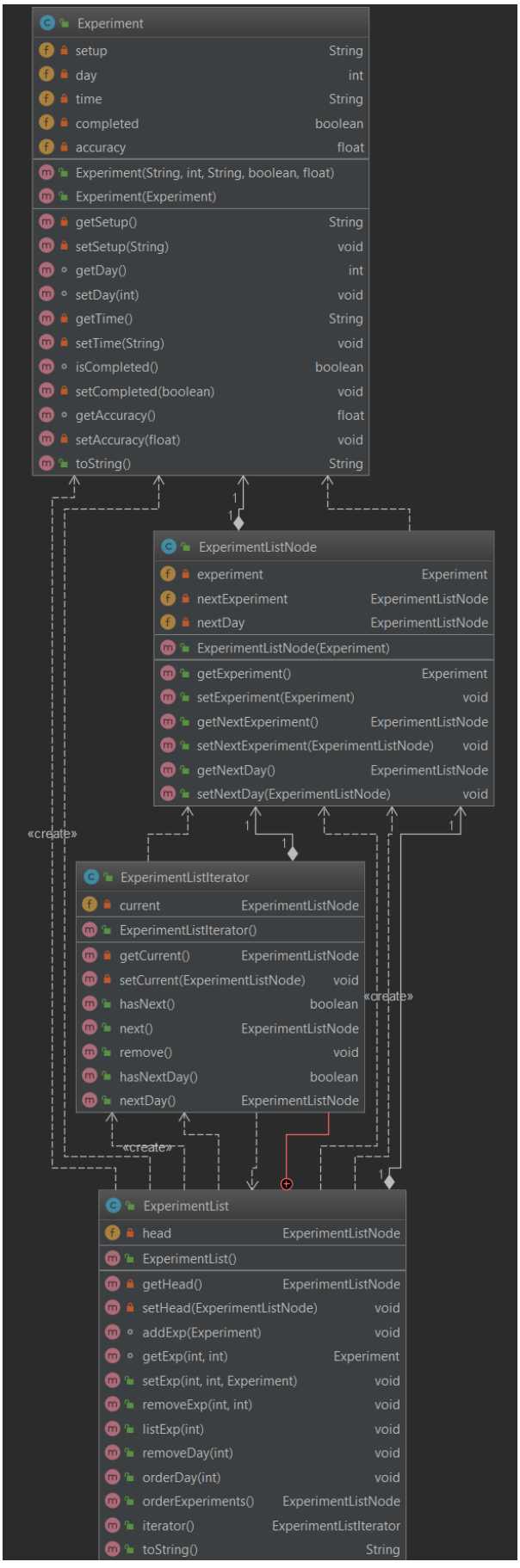
1.2 System Requirements

To solve this problem it is required to simulate some sort of single linked list structure that can hold experiment objects in them. A class that represents that kind of list structure is a requirement. Each linked is made out of nodes in theory, so there must be a class to represent that as well. All the experiments must be represented, which has to be done by another class. An iterator is required to traverse through the list, which has to be implemented as a class too.

To do operations required on the assignment, there must be a existing list. (Except adding, since it creates the list)

2 METHOD

2.1 Class Diagrams



2.2 Use Case Diagrams

Not required.

2.3 Other Diagrams (optional)

Not required.

2.4 Problem Solution Approach

To solve the problem it was required to simulate some kind of single linked list structure. 4 classes (including main) got created during the solution process. One class for creating nodes of the list, one class to represent the experiments, one class that represents the experiment list itself.

a) **Experiment Class:**

This class is used to represent experiments.
Experiments have these datas;
-Setup
-Day
-Time
-Completed status
-Accuracy
In this class, these data's getter and setter functions are implemented, so as the constructor, a copy constructor and the toString method.

b) **ExperimentListNode Class:**

This class is used to represent experiment list nodes. This class contains these variables:
-Experiment: Is the experiment that is required to hold in the node.
-NextExperiment : Is the next experiment that the current experiment is pointing.
-NextDay: Is the next day that the current experiment is pointing.
In this class, these variable's getter and setter methods are implemented so as the constructor.

c) **ExperimentList Class:**

This class is used to represent experiment list itself. This class implements the Iterable interface so it can use a iterator inside.
This class only has one variable which is the head of the list in the type of ExperimentListNode. Getter and setter methods are implemented in the class for the head as well.

-ExperimentListIterator Inner Class:

ExperimentList class has an inner class ExperimentListIterator which represents the iterator of the list. This class has one variable which is the current value of the iterator, in the type of ExperimentListNode.
This class has getter and setter methods for the "current" variable, and also a constructor which initalises the iterator to point to the head. This class implements the iterator interface so it overrides hasNext(), next() and remove() methods. As an addition to those overrides to make traversing through days easier, hasNextDay() and nextDay() methods are implemented.

ExperimentList class has the following methods:

-**addExp(Experiment e):** Adds the given experiment to the list. Does the checks for if the list is empty, or the experiment is on an existing day or not. Then creates a new node with the given experiment and adds the experiment to the list. The addition is done to the end of the day that the experiment provide. If the day doesn't exist, it just becomes the first experiment of that day.

-getExp(int day , int index): Returns the experiment at the given day and index. Method first finds the starting experiment of the given day then traverses the list until it founds the given index of the day. If there is not enough element it throws and exception. If the day doesnt exist in the list, it throws an exception too.

-setExp(int day, int index, Experiment e): Sets the given experiment to the given day and index. It does day and index checks and also empty list checks and throws exceptions if necessary.

-removeExp(int day , int index): Removes the experiment at the given day and index. It does the day and index checks so as the empty list check and throws exceptions if necessary. This method uses ExperimentListIterator's remove method to remove the elements.

-listExp(int day): Lists the completed experiments of the given day. It does the day and empty list checks and throws necessary exceptions. First iterator goes to the first experiment of the given day then traverses and prints the experiments one by one until the day changes.

-removeDay(int day): Removes the experiments of a given day. First it finds the given day in the list then removes the experiments one by one using the ExperimentListIterator's remove method. It also does the day check and empty list check and throws exceptions if necessary.

-orderDay(int day): Orders the given day's experiments according to their accuracy. Two iterators traverse until they reach the end, then one of them is incremented by one so it can traverse ahead to check if there are any smaller elements.

-orderExperiments(): Orders all the experiments according to their accuracy. First to not change the list, method copies all the current list to another list. Then two iterators are sent to do the search for the sorting. Returns the head of the newly created list.

-iterator(): Returns a new ExperimentListIterator object. Overridden method for the implementation of Iterable

-toString(): Overriding of the toString method to print a list better.

3 RESULT

3.1 Test Cases

Except normal usage of the methods, test cases include wrong day and index entries. And they are all handled.

3.2 Running Results

The outputs are from the current working program.

```
--------------------CONSTRUCTING THE LIST--------------------
Experiment Setup:E1
Day:1
Time:10:02:24
Completed:false
Accuracy:1.0

Experiment Setup:E2
Day:2
Time:10:02:24
Completed:true
Accuracy:61.45

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E4
Day:3
Time:10:02:24
Completed:true
Accuracy:5.45

Experiment Setup:E8
Day:3
Time:10:02:24
Completed:true
Accuracy:16.54

Experiment Setup:E5
Day:4
Time:10:02:24
Completed:true
Accuracy:54.45

Experiment Setup:E6
Day:5
Time:10:02:24
Completed:true
Accuracy:16.54
```

```
--------------------ADD TEST--------------------
Experiment Setup:E1
Day:1
Time:10:02:24
Completed:false
Accuracy:1.0

Experiment Setup:E2
Day:2
Time:10:02:24
Completed:true
Accuracy:61.45

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E4
Day:3
Time:10:02:24
Completed:true
Accuracy:5.45

Experiment Setup:E8
Day:3
Time:10:02:24
Completed:true
Accuracy:16.54

Experiment Setup:E5
Day:4
Time:10:02:24
Completed:true
Accuracy:54.45

Experiment Setup:E7
Day:4
Time:10:02:24
Completed:true
Accuracy:26.54

Experiment Setup:E6
Day:5
Time:10:02:24
Completed:true
Accuracy:16.54

Experiment Setup:E9
Day:6
Time:10:02:24
Completed:true
Accuracy:1.54


```

```
--------------------GET EXP TEST----------
Experiment Setup:E4
Day:3
Time:10:02:24
Completed:true
Accuracy:5.45
--Testing with wrong index--
No such index
--------------------SET EXP TEST----------
Experiment Setup:E1
Day:1
Time:10:02:24
Completed:false
Accuracy:1.0

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E4
Day:3
Time:10:02:24
Completed:true
Accuracy:5.45

Experiment Setup:E8
Day:3
Time:10:02:24
Completed:true
Accuracy:16.54

Experiment Setup:E5
Day:4
Time:10:02:24
Completed:true
Accuracy:54.45

Experiment Setup:E7
Day:4
Time:10:02:24
Completed:true
Accuracy:26.54

Experiment Setup:E6
Day:5
Time:10:02:24
Completed:true
Accuracy:16.54

Experiment Setup:E9
Day:6
Time:10:02:24
Completed:true
Accuracy:1.54


--Testing to set a experiment with a different day value to check exception handling--
Mismatch of given experiment and chosen experiment

--------------------REMOVE EXP TEST--------------------
Experiment Setup:E1
Day:1
Time:10:02:24
Completed:false
Accuracy:1.0

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E4
Day:3
Time:10:02:24
Completed:true
Accuracy:5.45

Experiment Setup:E8
Day:3
Time:10:02:24
Completed:true
Accuracy:16.54

Experiment Setup:E5
Day:4
Time:10:02:24
Completed:true
Accuracy:54.45


Experiment Setup:E7
Day:4
Time:10:02:24
Completed:true
Accuracy:26.54

Experiment Setup:E9
Day:6
Time:10:02:24
Completed:true
Accuracy:1.54


--Testing invalid day--
Index or day mismatch
```

```
--------------------LIST EXP TEST--------------------
Experiment Setup:E5
Day:4
Time:10:02:24
Completed:true
Accuracy:54.45
Experiment Setup:E7
Day:4
Time:10:02:24
Completed:true
Accuracy:26.54

--Testing invalid day--
Invalid day
--------------------REMOVE DAY TEST--------------------
Experiment Setup:E1
Day:1
Time:10:02:24
Completed:false
Accuracy:1.0

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E5
Day:4
Time:10:02:24
Completed:true
Accuracy:54.45

Experiment Setup:E7
Day:4
Time:10:02:24
Completed:true
Accuracy:26.54

Experiment Setup:E9
Day:6
Time:10:02:24
Completed:true
Accuracy:1.54


--Testing with an invalid day--
There is no such given day in the list.
--------------------ORDER DAY TEST--------------------
Experiment Setup:E1
Day:1
Time:10:02:24
Completed:false
Accuracy:1.0

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E3
Day:2
Time:10:02:24
Completed:true
Accuracy:53.45

Experiment Setup:E5
Day:4
Time:10:02:24
Completed:true
Accuracy:54.45

Experiment Setup:E7
Day:4
Time:10:02:24
Completed:true
Accuracy:26.54

Experiment Setup:E9
Day:6
Time:10:02:24
Completed:true
Accuracy:1.54


--Testing with a invalid day--
There is no such day.
--------------------ORDER EXPERIMENTS TEST
    --------------------
Experiment Setup:E1
Day:0
Time:10:02:24
Completed:false
Accuracy:1.0

Process finished with exit code 0
```

Time Complexities of The Methods:

Methods	Best Case	Average Case	Worst Case
addExp	O(1)	O(n)	O(n)
getExp	O(n)	O(n)	O(n)
setExp	O(n)	O(n)	O(n)
removeExp	O(n)	O(n)	O(n)
listExp	O(n)	O(n)	O(n)
removeDay	O(n)	O(n)	O(n)
orderDay	O(n)	$O(n^2)$	$O(n^2)$
orderExp	O(n)	$O(n^2)$	$O(n^2)$

addExp : adding the first experiment only adds to the head so it has constant complexity. Rest of the adding conditions require not nested loops which ends up with the complexity at O(n).

getExp : To get the requested experiment 2 not nested loops has to loop no matter what so from O(2n) the time complexity is O(n).

setExp : If exceptions are ignored, the method will executed 2 not nested loops sof rom O(2n) the time complexity is O(n).

removeExp : To remove the experiment either 2 not nested loops has to loop or if the experiment is at the head it will call the remove method which also has an O(n) time complexity. So it is either O(2n) -> O(n) or just O(n) from the method call.

listExp : Method will use 2 not nested loops no matter what so from O(2n) -> O(n).

removeDay : Method will use 2 not nested loops no matter what so from O(2n) -> O(n).

orderDay : Method uses 2 not nested and 2 nested loops. The worst case would be entering all the loops yet $O(n^2+2n)$ -> $O(n^2)$. The best case would be O(n) where there is one element and n would be 1.

orderExperiments : Method uses 2 not nested and 2 nested loops. The worst case would be entering all the loops yet $O(n^2+2n)$ -> $O(n^2)$. The best case would be O(n) where there is one element and n would be 1.

- Main titles -> 16pt , 2 line break
- Subtitles -> 14pt, 1.5 line break
- Paragraph -> 12pt, 1.5 line break