Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 3 REPORT

AHMET YASIR NACAK
161044042

Course Assistant: Mehmet Burak KOCA

1 INTRODUCTION

1.1 Problem Definition

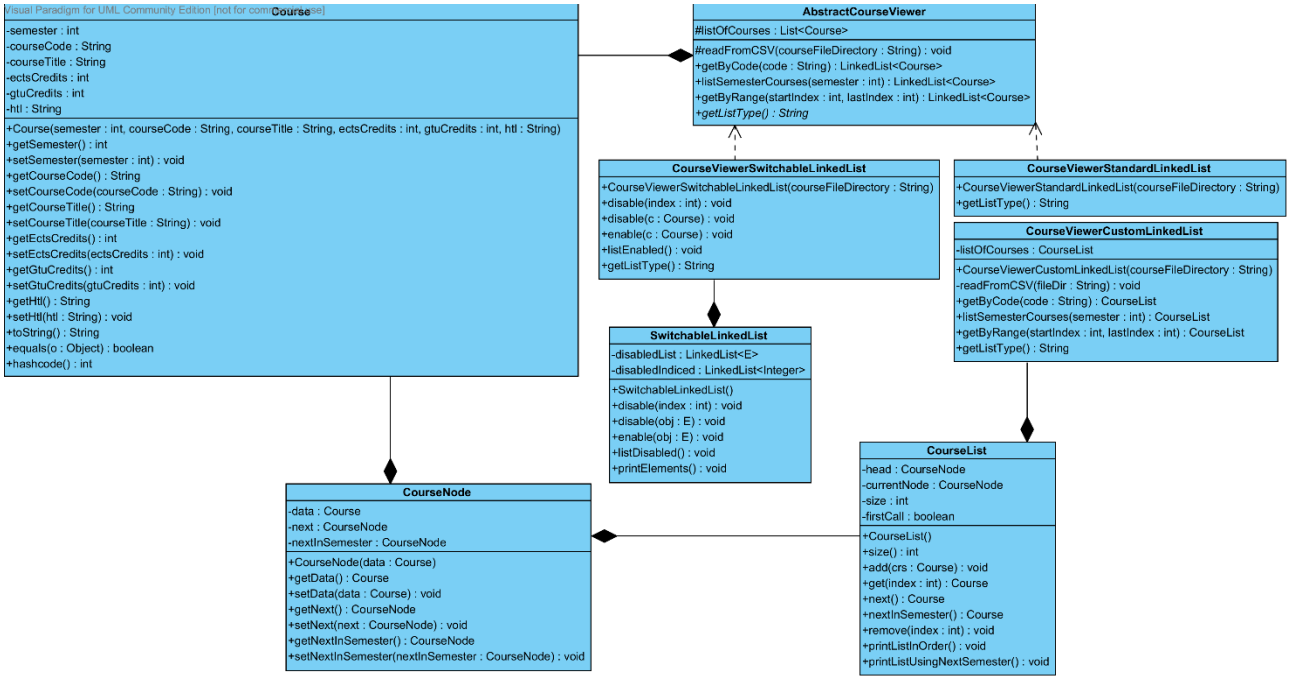
In this homework, the problem is to create three data structures that are being used to represent lists of courses and ways to show them. First of these three is the simplest of them. This data structures the simple linked list that is built-in. The second is and extension of the linked list that adds the ability to enable or disable certain elements. The final data structure is a linked list from scratch that has two links showing next elements. One is for the element right after it and the other is for the next element that is in the same semester as the current one.

1.2 System Requirements

To create the data structures mentioned in the problem definition, we need to create a simple data structure that can represent a single course. In this homework, each course has six attributes. Those attributes are the semester of the course, code and the title of the course, credits of the course in ECTS standards and GTU standards and lastly, the weekly hour distribution of the course. After creating a structure that can represent a class, we need to create container type data structures that are able to hold more than one course. For the first data structure requirement, we don't need to create a new data structure because the linked list structure provides our needs with it. For the second data structure we need to extend the basic linked list and add enabling and disabling abilities and a way to see the disabled items. For the last data structure, we need to create linked list with custom abilities from scratch. That means that we need to create a node structure that holds a course and two links for the element after it. This node structure also needs operations such as advancing in forward direction based on indices or semesters. After creating the node structure, we need to create a head node in the custom linked list structure. This head node allows us to keep an element with ourselves and this element can show the other element after that and so on. For the operations of the custom linked list structure, we need to add the simple operations that are adding an element, removing an element, accessing an element and advancing in the list. After creating the container structures in the homework, we need to create classes that can utilize these containers in terms of feeding data to them and letting the user view the courses of their choices. For this, creating 3 structures that has one of the required data structures is a good solution but since all three course viewing structures will utilize the same operations, creating a parent for them is a better solution. With this approach, we can guarantee that there won't be unnecessary duplication of operations.

2 METHOD

2.1 Class Diagram



2.2 Problem Solution Approach

2.2.1 The Course Class

To create a class that holds courses, we need to create 6 data fields. First of them represents the semester of the class and this fields type is integer. After that we need to add two strings. One of them is the code of the course (CSE 101 for example) and the title of the class (Introduction to Computer Engineering for example). Once these are done, we need to create two integers for the credits of the course. Since we are writing this system for the GTU standards and GTU uses two types of credits for the courses, we are using two integers. One is for the GTU credits and the other for ECTS credits of the class. After all this, we need to add a string field that represents the hours distribution of the course. Since we are creating a course viewing system out of this, we do not need to separate the hours as integers. If we were creating a system that is doing calculations based on the hours distribution, we needed to separate the string to 3 integers. After the data fields, we need to create getters and setters for this class and we are done with it. The course class is independent of the other classes in this system, so it does not take inheritance from them.

2.2.2 The Switchable Linked List Class

The requirement of the second data structure is to have an extension of the standard linked list class that also can enable and disable items in it. To create this data structure, we need to take inheritance from the built-in linked list class that is in Java. After inheriting the methods in that class, we need to write new methods that is going to help us enable/disable elements in the list. For that, I used another linked list that holds the disabled courses and a helper list of that holds the indices of the disabled items. This way, once a course gets disabled from the system, it gets put in the disabled items list and its index before the removal gets put in the list of disabled indices. I also added two ways of disabling an item. Once is using the index value of the element and the other is using a course, searching it in the list and disabling it. For the enable method, we need to put the item back in its disabled place, so it is a procedure of adding the element back to the list using its previous position information and removing that item from the list of disabled indices. We also needed to create a method that can print out the disabled elements. For that, I just iterated through the list of disabled items and printed each of them.

2.2.3 The Course List and Course Node Classes

For the last data structure, we need to create a linked list from scratch without using any predefined method or inheriting another class. To create a linked list, we first need to create a node, a class that holds the course information, the element after it in the index order and the element after it based on the semester in this case. This class needs all the getters and setters because the list class will utilize these methods to create links between nodes. After creating the node class, we need to create the actual linked list data structure. This data structure must hold the starting node and since we are going to iterate through it, this class needs to have the information of the current node its in. And most importantly, we need to hold the size of the list because size information encapsulates the way we iterate through the list and provides the user a simpler method to go through elements. After creating these fields, we need to add the methods. First of these methods is the add method. This method takes an object and puts it at the end of the list. To do that, this method iterates through the list and finds the first element that has not got an element for its next node then converts the given object to a node and puts it after the element that hasn't got a next node before. After this connection, add method needs a way to find the previous and after elements of this node that has the same semester information as this one. To achieve that, the list starts scanning from the first element and stops when it finds an element that has the same semester as newly added element. Then connects given elements next in semester element to the found one. Then the method continues to iterate and marks every element that has the same semester. Once the iteration is finished, the last element that market is the one that is at the previous of the added element and it has the same semester. Because of that, the

method connects two nodes, and this concludes the adding method. The time complexity of this method is O (n). After the adding method, we need to add a method that can remove an element. To do that, we need to iterate through the list, find the element that is 1 index before the element we are trying to remove and connect its next element to the element that is after one we are trying to remove. This removes all the links coming to the element we are trying to remove and makes it no longer available to our list. To change the next in semester component, we need to do something similar. We need to iterate through the list, find the element that has the element we want to remove as its next in semester element and connects its next in semester node to the next in semester node of the element we want to remove. This method also runs in O (n). After that, we need to create a get method that allows the user to put an index and get the element that is the given index. For this we just need to iterate through the list once and need to stop when the given index amount of iterations is done and return current element we are on. This method also runs in O (n). And finally, in order utilize the iterator like node of our class, we need to add a next and next in semester methods. These methods are simple because they just call the corresponding methods of the node class. These methods run in O (1).

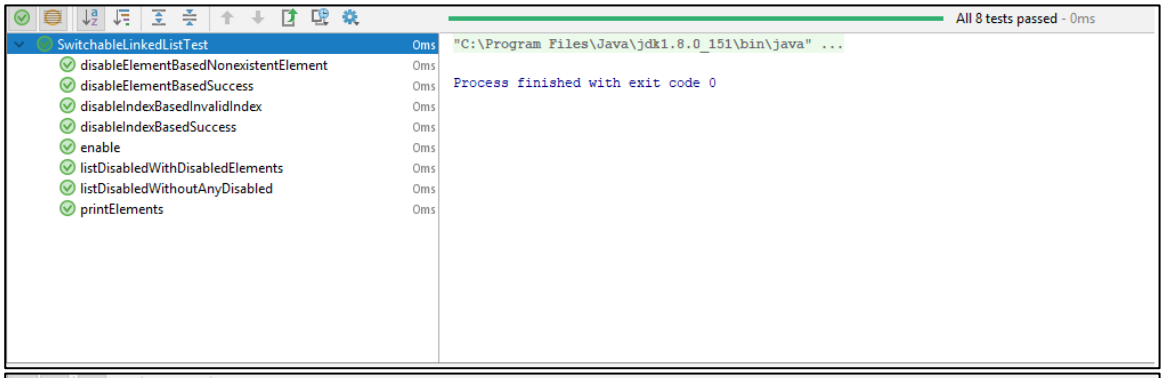
2.2.4 The Course Viewer Class Hierarchy

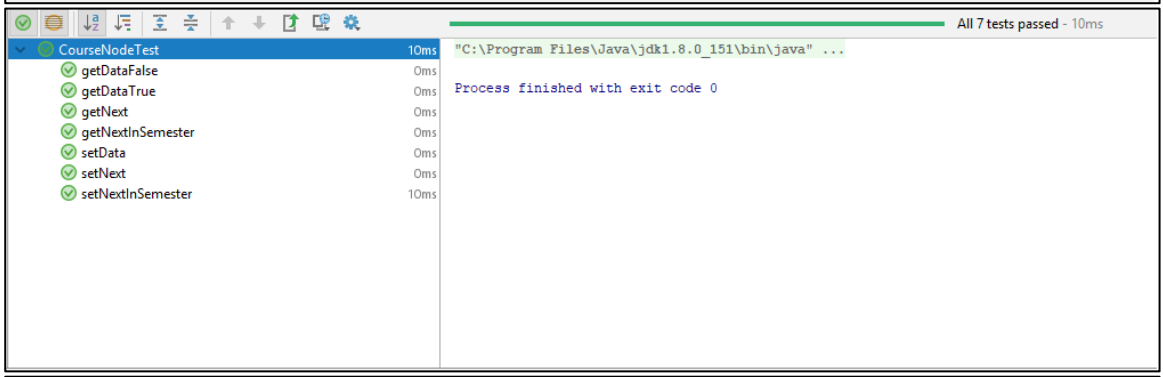
The course viewer class hierarch contains an abstract class called AbstractCourseViewer. This class is the parent of the three types of course viewers. Each of those course viewers utilize one type of the data structures we created before. The standard linked list, the switchable linked list and the custom linked list that holds courses and next in semester information of those courses. This class needs to have methods that read a CSV file and convert it to a list. After that this class needs to have a method that filters out certain courses and puts them in lists and returns them. The three types of filtering are the one that creates a list with all the courses that have a certain code, the second is the one that creates a list based on the semesters of the courses and the third is one that creates a list starting by copying all the elements from a certain and index and ends at another in the big list.

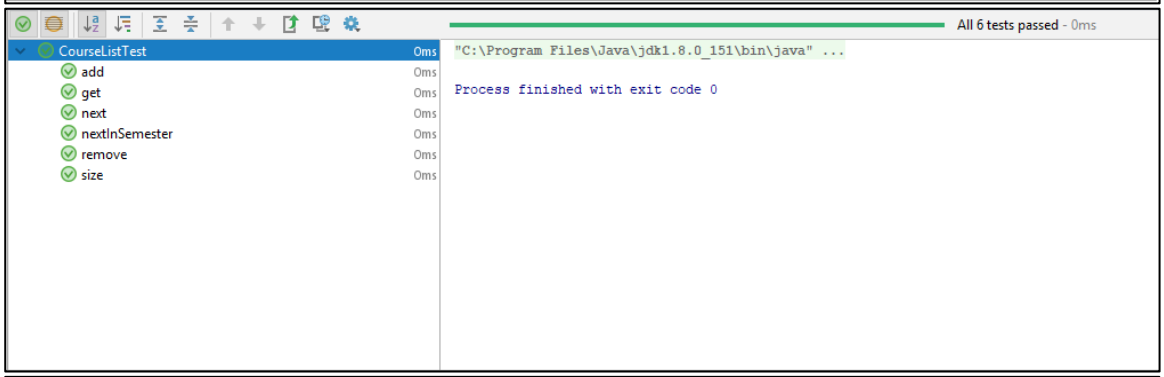
3 RESULT

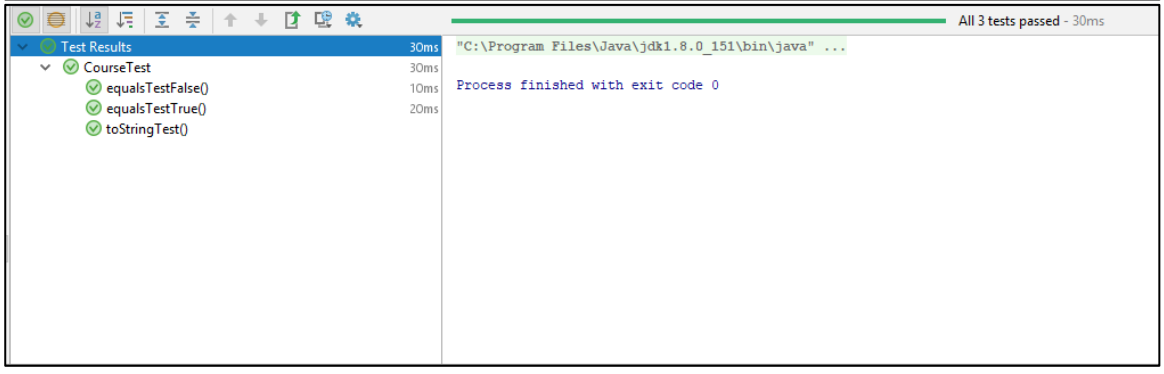
3.1 Test Cases

Here are the results of the test codes. Each of them is described in the Javadoc.









3.2 Running Results

These are the sample test codes that tests a simple and different scenario using all three data structures we've created in this homework.

Standard Linked List Tests:

CODE:

```
CourseViewerStandardLinkedList courseViewLinkedList = new CourseViewerStandardLinkedList( courseFileDirectory: "res/Courses.csv");
int semesterControl = 1;
int startIndexControl = 13, endIndexControl = 23;
String codeControl = "XXX XXX";

System.out.printf("\nPrinting courses in the semester %d:\n", semesterControl);
LinkedList<Course> testSemesterCourses = courseViewLinkedList.listSemesterCourses(semesterControl);
for (Course testSemesterCourse : testSemesterCourses) {
    System.out.println(testSemesterCourse);
}

System.out.printf("\nPrinting courses with %s code:\n", codeControl);
LinkedList<Course> testCoursesWithCode = courseViewLinkedList.getByCode(codeControl);
for (Course aTestCoursesWithCode : testCoursesWithCode) {
    System.out.println(aTestCoursesWithCode);
}

System.out.printf("\nPrinting courses in range of %d-%d:\n", startIndexControl, endIndexControl);
LinkedList<Course> testCoursesInRange = courseViewLinkedList.getByRange(startIndexControl, endIndexControl);
for (Course aTestCoursesInRange : testCoursesInRange) {
    System.out.println(aTestCoursesInRange);
}
```

OUTPUT:

```
Printing courses in the semester 1:
Semester: 1. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSC). ECTS and GTU Credits: 2, 1. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: CSE 101 - Introduction To Computer Engineering. ECTS and GTU Credits: 8, 3. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: CSE 107 - Introduction To Computer Science Laboratory. ECTS and GTU Credits: 2, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: MATH 101 - Calculus I. ECTS and GTU Credits: 7, 5. H+T+L: 5+0+0.
Semester: 1. Course Code and Title: PHYS 121 - Physics I. ECTS and GTU Credits: 6, 4. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: PHYS 151 - Physics Laboratory I. ECTS and GTU Credits: 1, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: SSTR 101 - Principles Of Atatürk And The History Of Turkish Revolution I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: TUR 101 - Turkish I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.

Printing courses with XXX XXX code:
Semester: 1. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSC). ECTS and GTU Credits: 2, 1. H+T+L: 2+0+0.
Semester: 2. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSC). ECTS and GTU Credits: 2, 1. H+T+L: 2+0+0.
Semester: 3. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSB). ECTS and GTU Credits: 3, 2. H+T+L: 2+0+0.
Semester: 4. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSB). ECTS and GTU Credits: 3, 2. H+T+L: 2+0+0.
Semester: 5. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSA). ECTS and GTU Credits: 3, 2. H+T+L: 2+0+0.
Semester: 6. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSA). ECTS and GTU Credits: 3, 0. H+T+L: 2+0+0.
Semester: 7. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSB). ECTS and GTU Credits: 3, 2. H+T+L: 2+0+0.
Semester: 8. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSB). ECTS and GTU Credits: 3, 2. H+T+L: 2+0+0.
Semester: 8. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSA). ECTS and GTU Credits: 3, 2. H+T+L: 2+0+0.

Printing courses in range of 13-23:
Semester: 2. Course Code and Title: PHYS 152 - Physics Laboratory II. ECTS and GTU Credits: 1, 1. H+T+L: 0+0+2.
Semester: 2. Course Code and Title: SSTR 102 - Principles Of Atatürk And The History Of Turkish Revolution II. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 2. Course Code and Title: TUR 102 - Turkish II. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 3. Course Code and Title: CSE 241 - Object Oriented Programming. ECTS and GTU Credits: 9, 5. H+T+L: 3+2+0.
Semester: 3. Course Code and Title: CSE 211 - Discrete Mathematics. ECTS and GTU Credits: 6, 3. H+T+L: 3+0+0.
Semester: 3. Course Code and Title: CSE 231 - Circuits And Electronics. ECTS and GTU Credits: 8, 4. H+T+L: 4+0+0.
Semester: 3. Course Code and Title: CSE 233 - Circuits And Electronics Laboratory. ECTS and GTU Credits: 2, 1. H+T+L: 0+0+2.
Semester: 3. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSB). ECTS and GTU Credits: 3, 2. H+T+L: 2+0+0.
Semester: 3. Course Code and Title: EN 111 - English For Business Life. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 4. Course Code and Title: CSE 222 - Data Structures And Algorithms. ECTS and GTU Credits: 9, 5. H+T+L: 4+2+0.
```

Switchable Linked List (Enable/Disable):

CODE:

```
Disabled Item(s):
Semester: 1. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSC). ECTS and GTU Credits: 2, 1. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: PHYS 121 - Physics I. ECTS and GTU Credits: 6, 4. H+T+L: 3+0+0.

Printing courses in the semester 1:
Semester: 1. Course Code and Title: CSE 101 - Introduction To Computer Engineering. ECTS and GTU Credits: 8, 3. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: CSE 107 - Introduction To Computer Science Laboratory. ECTS and GTU Credits: 2, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: MATH 101 - Calculus I. ECTS and GTU Credits: 7, 5. H+T+L: 5+0+0.
Semester: 1. Course Code and Title: PHYS 151 - Physics Laboratory I. ECTS and GTU Credits: 1, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: SSTR 101 - Principles Of Atatürk And The History Of Turkish Revolution I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: TUR 101 - Turkish I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.

Re-enabling physics I.

Disabled Item(s):
Semester: 1. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSC). ECTS and GTU Credits: 2, 1. H+T+L: 2+0+0.

Printing courses in the semester 1:
Semester: 1. Course Code and Title: CSE 101 - Introduction To Computer Engineering. ECTS and GTU Credits: 8, 3. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: CSE 107 - Introduction To Computer Science Laboratory. ECTS and GTU Credits: 2, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: MATH 101 - Calculus I. ECTS and GTU Credits: 7, 5. H+T+L: 5+0+0.
Semester: 1. Course Code and Title: PHYS 121 - Physics I. ECTS and GTU Credits: 6, 4. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: PHYS 151 - Physics Laboratory I. ECTS and GTU Credits: 1, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: SSTR 101 - Principles Of Atatürk And The History Of Turkish Revolution I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: TUR 101 - Turkish I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
```

OUTPUT:

```
CourseViewerSwitchableLinkedList courseViewEnableDisable = new CourseViewerSwitchableLinkedList( courseFileDirectory: "res/Courses.csv");

courseViewEnableDisable.disable( index: 0);
courseViewEnableDisable.disable( index: 3);
courseViewEnableDisable.listDisabled();

System.out.printf("\nPrinting courses in the semester %d:\n", semesterControl);
testSemesterCourses = courseViewEnableDisable.listSemesterCourses(semesterControl);
for (Course testSemesterCourse : testSemesterCourses) {
    System.out.println(testSemesterCourse);
}
System.out.println("\nRe-enabling physics I.\n");

courseViewEnableDisable.enable(new Course( semester: 1, courseCode: "PHYS 121", courseTitle: "Physics I", ectsCredits: 6, gtuCredits: 4, htl: "3+0+0"));
courseViewEnableDisable.listDisabled();
System.out.printf("\nPrinting courses in the semester %d:\n", semesterControl);
testSemesterCourses = courseViewEnableDisable.listSemesterCourses(semesterControl);
for (Course testSemesterCourse : testSemesterCourses) {
    System.out.println(testSemesterCourse);
}
```

Custom Linked List (Next In Semester Capability):

CODE:

```
Printing courses that are in semester 1 using NextSemester() method:
Semester: 1. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSC). ECTS and GTU Credits: 2, 1. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: CSE 101 - Introduction To Computer Engineering. ECTS and GTU Credits: 8, 3. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: CSE 107 - Introduction To Computer Science Laboratory. ECTS and GTU Credits: 2, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: MATH 101 - Calculus I. ECTS and GTU Credits: 7, 5. H+T+L: 5+0+0.
Semester: 1. Course Code and Title: PHYS 121 - Physics I. ECTS and GTU Credits: 6, 4. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: PHYS 151 - Physics Laboratory I. ECTS and GTU Credits: 1, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: SSTR 101 - Principles Of Atatürk And The History Of Turkish Revolution I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: TUR 101 - Turkish I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
--------------------------------------------------------------------------------------------------------------------------------
Removing an element and printing the courses that are in same semester as before using NextSemester() method:
--------------------------------------------------------------------------------------------------------------------------------
Semester: 1. Course Code and Title: XXX XXX - Teknik Olmayan Seçmeli (SSC). ECTS and GTU Credits: 2, 1. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: CSE 101 - Introduction To Computer Engineering. ECTS and GTU Credits: 8, 3. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: MATH 101 - Calculus I. ECTS and GTU Credits: 7, 5. H+T+L: 5+0+0.
Semester: 1. Course Code and Title: PHYS 121 - Physics I. ECTS and GTU Credits: 6, 4. H+T+L: 3+0+0.
Semester: 1. Course Code and Title: PHYS 151 - Physics Laboratory I. ECTS and GTU Credits: 1, 1. H+T+L: 0+0+2.
Semester: 1. Course Code and Title: SSTR 101 - Principles Of Atatürk And The History Of Turkish Revolution I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
Semester: 1. Course Code and Title: TUR 101 - Turkish I. ECTS and GTU Credits: 2, 2. H+T+L: 2+0+0.
```

OUTPUT:

```
CourseViewerCustomLinkedList courseViewNextInSemester = new CourseViewerCustomLinkedList( courseFileDirectory: "res/Courses.csv");
System.out.printf("Printing courses that are in semester %d using NextSemester() method:\n", semesterControl);
courseViewNextInSemester.listOfCourses.printListUsingNextSemester(semesterControl);
courseViewNextInSemester.listOfCourses.remove( index: 2);
System.out.println("\n--------------------------------------------------------------------------------------------------------------------------------");
System.out.println("Removing an element and printing the courses that are in same semester as before using NextSemester() method:");
System.out.println("\n--------------------------------------------------------------------------------------------------------------------------------");
courseViewNextInSemester.listOfCourses.printListUsingNextSemester(semesterControl);
```