Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 3 REPORT

CENGİZ TOPRAK
161044087

Course Assistant:
BURAK KOCA

1 INTRODUCTION

1.1 Problem Definition

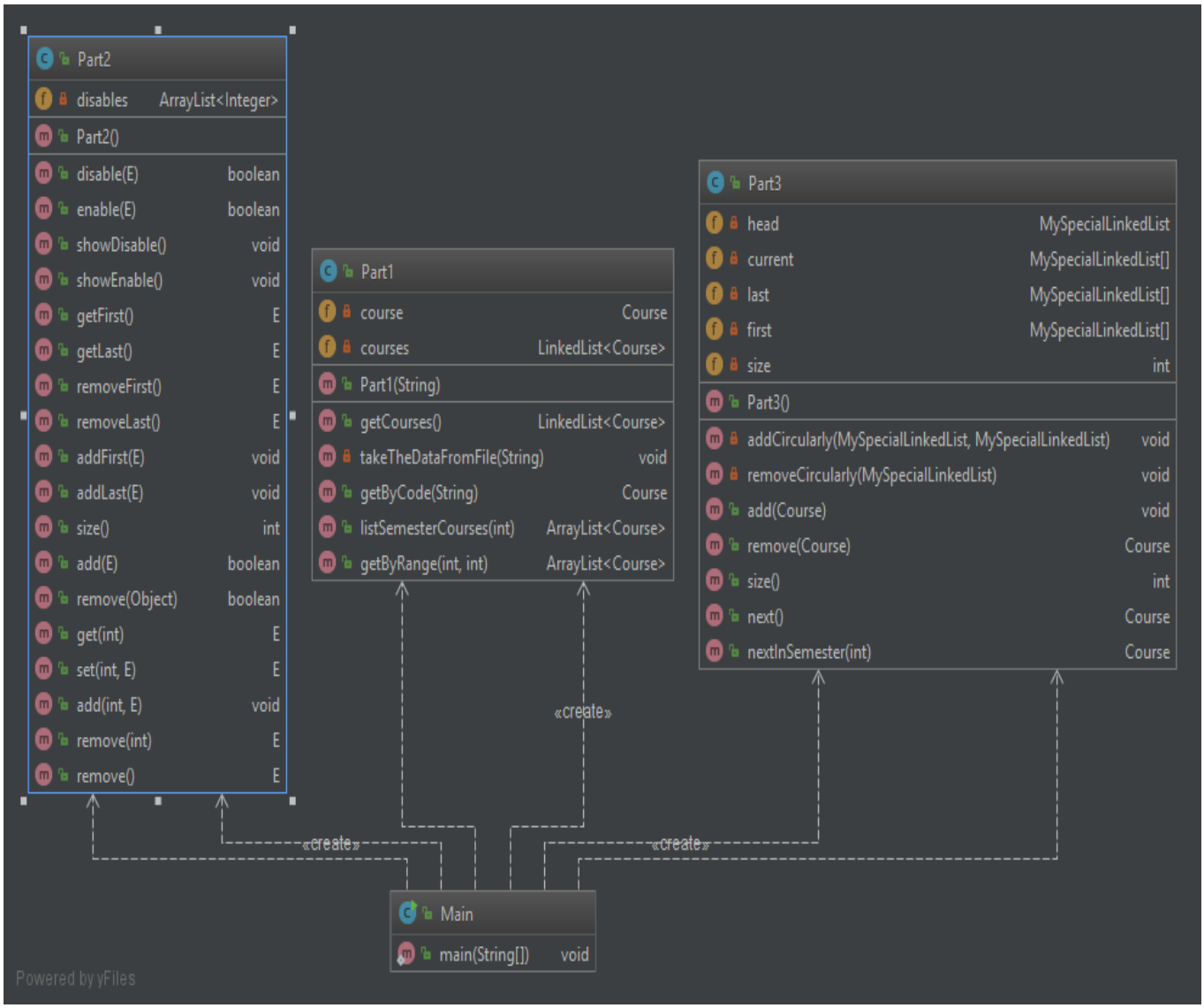
In this homework,it is expected that we understand linklist and circular linklist and implement this data structures.

1.2 System Requirements

In the part1,system need a class that keep the one instance of linkedlist and a few methods that manipulate the this linkedlist instance.
In the part2,system need a class that extends linkedlist , a few methods named disable(),enable(),showDisable() and overrided methods from super class.
In the part3,system need a class that implement a list like linkedlist and circular linkedlist and some methods that works on the its own data.

2 METHOD

2.1 Class Diagrams



Class Part2 extends Linkedlist class and override many methods of super class so that this overrided method do not access disable items in the list.

2.2 Problem Solution Approach

- PART1

It is used has relation by keeping an instanse of linked list and implemented three methods named getByCode(String code),getByRange(int first_index,int last_index) and listSemesterCources(int semester).

  - getByCode(String code) return a course if it exist course named code,if not,throws exception.

  - getByRange(int first_index,int last_index) return courses between the first_index and last_index if both indexes is valid ,if not,throws exception

  - listSemesterCources(int semester) return courses in the given semester if semester is valid,if not ,throws exception.

- PART 2

It is created a generic class by using is a relation and extending linkedlist class. This class implemented four methods named disable(E item),enable(E item) ,showDisable() and showEnable() ,and keep the Arraylist in order to location of disable items.

  - disable(E item) : return true if item is disabled and add the location of disabled item to arraylist,if not,return false.

  - enable(E item) : return true if item is enabled and delete location of enabled item from arraylist,if not,return false.

  - showEnable() : print the all enables items.

  - showDisable(): print the all diasbles items.

Note: if an element add to list in the location where location is smaller than disables items location,locations of disable items increase by one and if an element remove from list in the location where location is smaller than location of disables items , location of this items decrease by one.

- PART 3

It is expected that Our own linkedlist and circular linkedlist construct and implemented a few methods like size(),add(),remove(),next() ,nextInSameSemester(). It is kept a reference named head that refer to begining of data for linkedlist and three reference array named first,current and last.

- first---> refer to first element in the same semester for circular linkedlist.
- current ---> refer to element for next() and nextInSameSemester().
- last ---> refer to last element for add() method in order to add an element at constant time.

➔ add(Course item) : add item to end of linkedlist and if there is item that semeter is the same,connect item to circular linkedlist at the same semester by calling addCircularly().

➔ remove(Course item) : remove item from list and if there is item that semester is same,disconnect item from circular linkedlist by calling removeCirculary().

➔ next() : return next item from the list if next exist,if not,return null.

➔ NextInSameSemester(int semester): return next element from list in the same semester if there is any element,if there is no element in the given semester ,return null.

3 RESULT

3.1 Test Cases

Screenshots abour test cases is available in Test folder in the homework folder.

3.2 Running Results

Running results is available in homework folder as output.txt

3.3 Time Analysis

- PART 1
  - ➔ getByCode(Course item) --> function works linear time because it searches element from starting of list to ending of list until it find item or not.So ,T(n) is in Θ(n) .
  - ➔ getByRange(int start_index ,int last_index) -- > The best case of working function is Θ(1) if start_index = last_index. The worst case of function working is Θ(n) if last_index is index of last element in the list.
  - ➔ listSemesterCourses(int semester)-> T(n) is in Θ(n) because it scans the list from begin to end for finding elements in same semester

- PART 2
    - enable(E item)--> The best case of function is to item is first element of disable elements.So,$T_b(n)$ is in $\Theta(1)$.The worst case of function working is to item is last element of disable elements.So, $T_w(n)$ is in $\Theta(n)$
    - disable(E item)-->if there is no disable items,$T_1(n) = O(n)$ .If there are disable items,both function searches item into linkedlist and searches disable items.So $T_2(n) = O(n*m)$ where n is length of linkedlist and m is length of array of disable items.As a result, $T(n)= max(O(n),O(n*n)) = O(n^2)$
- PART 3

1-) remove(Cource item)--> The best case of function working is to item is first element of linkedlist and first element of circular linkedlist in same semester or last element of circular linkedlist,or last element of linkedlist and last element of circular linkedlist or first element of circular linkedlist in same semester.Therefore,$T_b(n)$ is in $O(1)$

The worst case of function working is to item is last or not first item of linkedlist and circular linkedlist.$T_w(n)$ is in $O(n*m)$ where n is length of linkedlist and m is circular linkedlist.So, $T_w(n)$ is in $O(n^2)$.

2-) add(Course item) ---> T(n) is in O(1) because there is reference that refer to last element of linked list and there is reference that is refer to last element of circular linkedlist in same semester.So,T(n) is in O(1).

3-)next()---> T(n) is in $\Theta(1)$

4-)nextInSameSemester(int semester)--> T(n) is in $\Theta(1)$

5-) size()---> T(n) is in $\Theta(1)$