**GTU**

**Department of Computer Engineering**

**CSE 222/505 - Spring 2020**

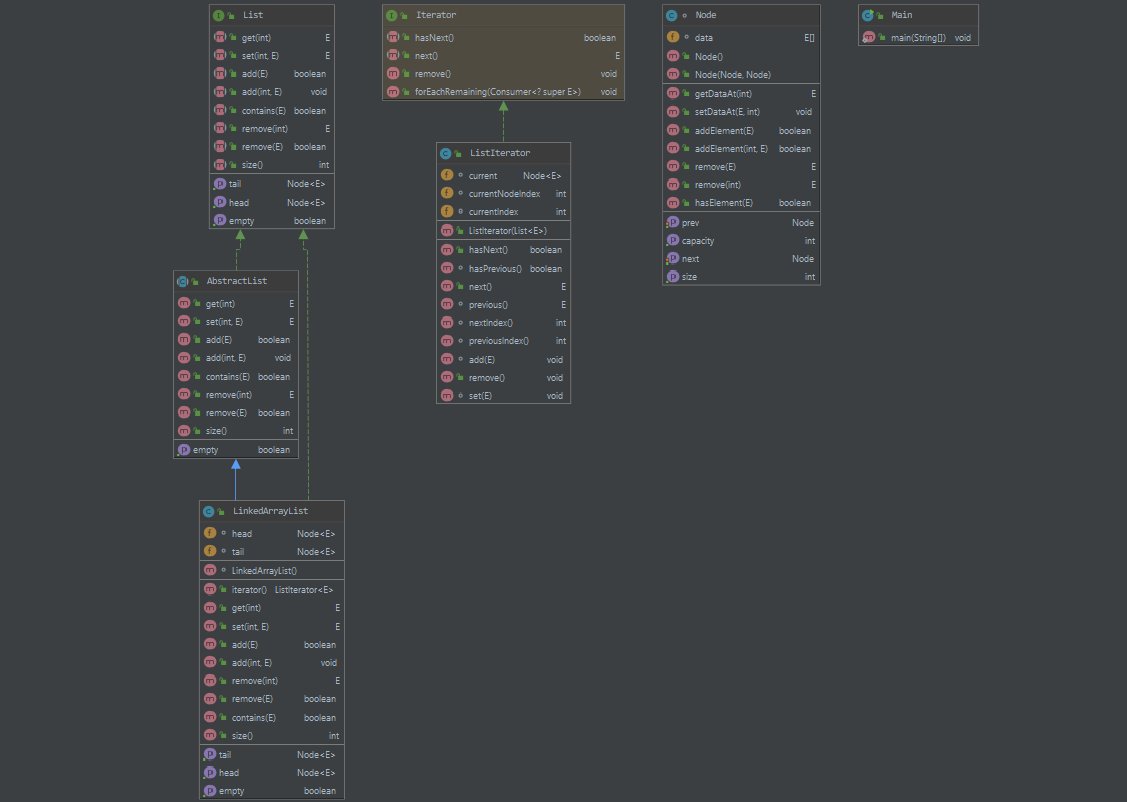
**Homework 3 Report**

**Abdulhakim Emre ARTIŞ**

**141044034**

**PART1**

**1-Class Diagrams:**



More deatailed diagram exist in Part1 directory.

**2-Problem Solution Approach**

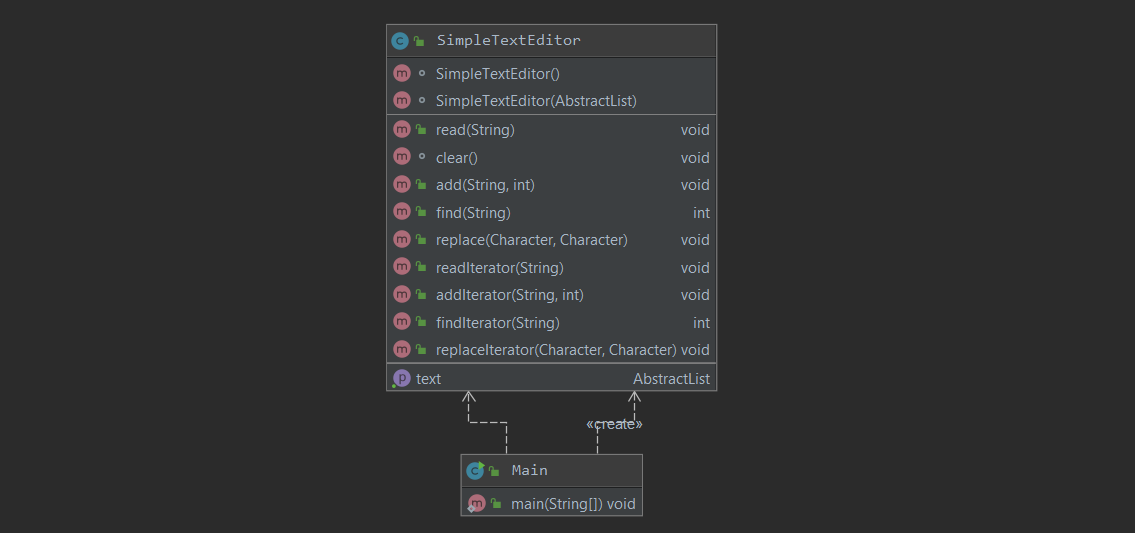
In this part i created a LinkedArrayList, which is a linked list with containing array with capacity 5. It extends AbstractList and implements list.It has 2 constructor, first one has no parameters next and previous shows null, second one gets next and previous node and creates bound with them.This class overrides necessary methods of this interface and abstract class. user can add element with 2 diffrent way, first with just giving element as parameter, it adds first empty node in this situation, second with giving element and index, it adds the element given index and swaps some elements to other node if it necessary. Remove functions also works like add functions .this class has an iterator function which returns iterator of it. That iterator class is implements iterator and methods aims same thing with Javas ListIterator , but it customized for LinkedArrayList class.

**3-Test Cases, Running and Results**

All methods tested, console outputs are avaliable in Part1/HW3P1ConsoleOutput.pdf

**PART2**

**1-Class Diagrams:**



**2-Solution Approach:**

For this part i created a class which wanted from us, to keep LinkedList and ArrayList together i created a classmember as AbstractList.This class has 2 constructors, if there is no parameter it creates objects data field as an ArrayList, with other constructor it creates objects data field with given Type of List.In this way the class can work with both of them.

**3-Test Cases, Running and Results**

All methods tested, console outputs are avaliable in Part2/ConsoleOutputP2.pdf

Also example log file exist in Part2/log.txt

**THEORICAL COMPARISON**

Comparison for read() method

ArrayList with iterator: It will run O(n) complexity

ArrayList without iterator: It will run O(n) complexity

LinkedList with Iterator: It will run O(n) complexity

LinkedList without Iterator: It will run O(n) complexity

Comparison for add() method

ArrayList with iterator: It will run O(nm) complexity

ArrayList without iterator: It will run O(nm) complexity

LinkedList with Iterator: It will run O(nm) complexity

LinkedList without Iterator: It will run O(nm) complexity

Comparison for find() method

ArrayList with iterator: It will run O(nm) complexity

ArrayList without iterator: It will run O(nm) complexity

LinkedList with Iterator: It will run O(nm) complexity

LinkedList without Iterator: It will run O(nm) complexity

Comparison for replace() method

ArrayList with iterator: It will run O(n) complexity

ArrayList without iterator: It will run O(n) complexity

LinkedList with Iterator: It will run O(n) complexity

LinkedList without Iterator: It will run O(n) complexity

EXPRIMENTAL COMPARISON

As we can see in the log.txt file the performances of method could be different with different operations. Some of them makes add faster some of it searches faster. But usually using iterators makeing the computetion faster.