GTU Department of Computer Engineering

CSE 222 - Spring 2020

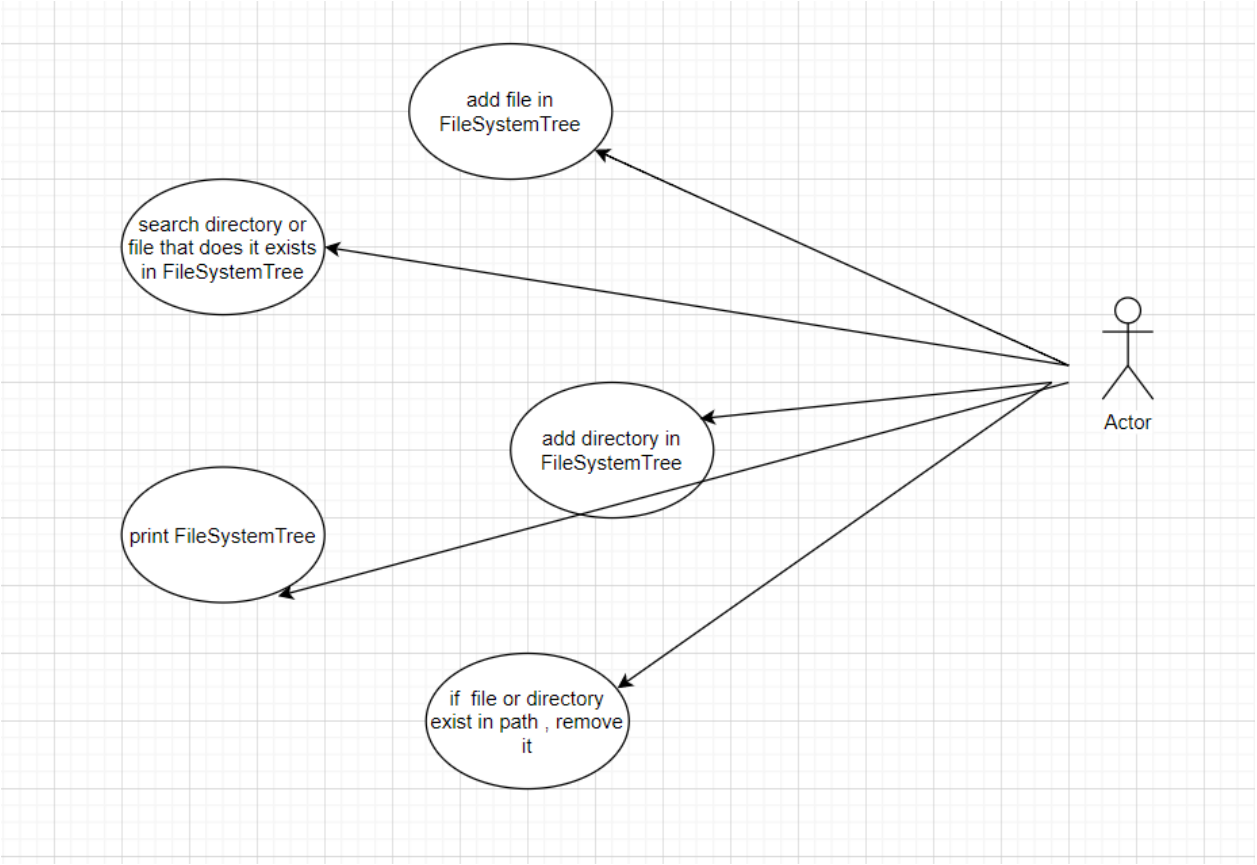
Homework 5 Report

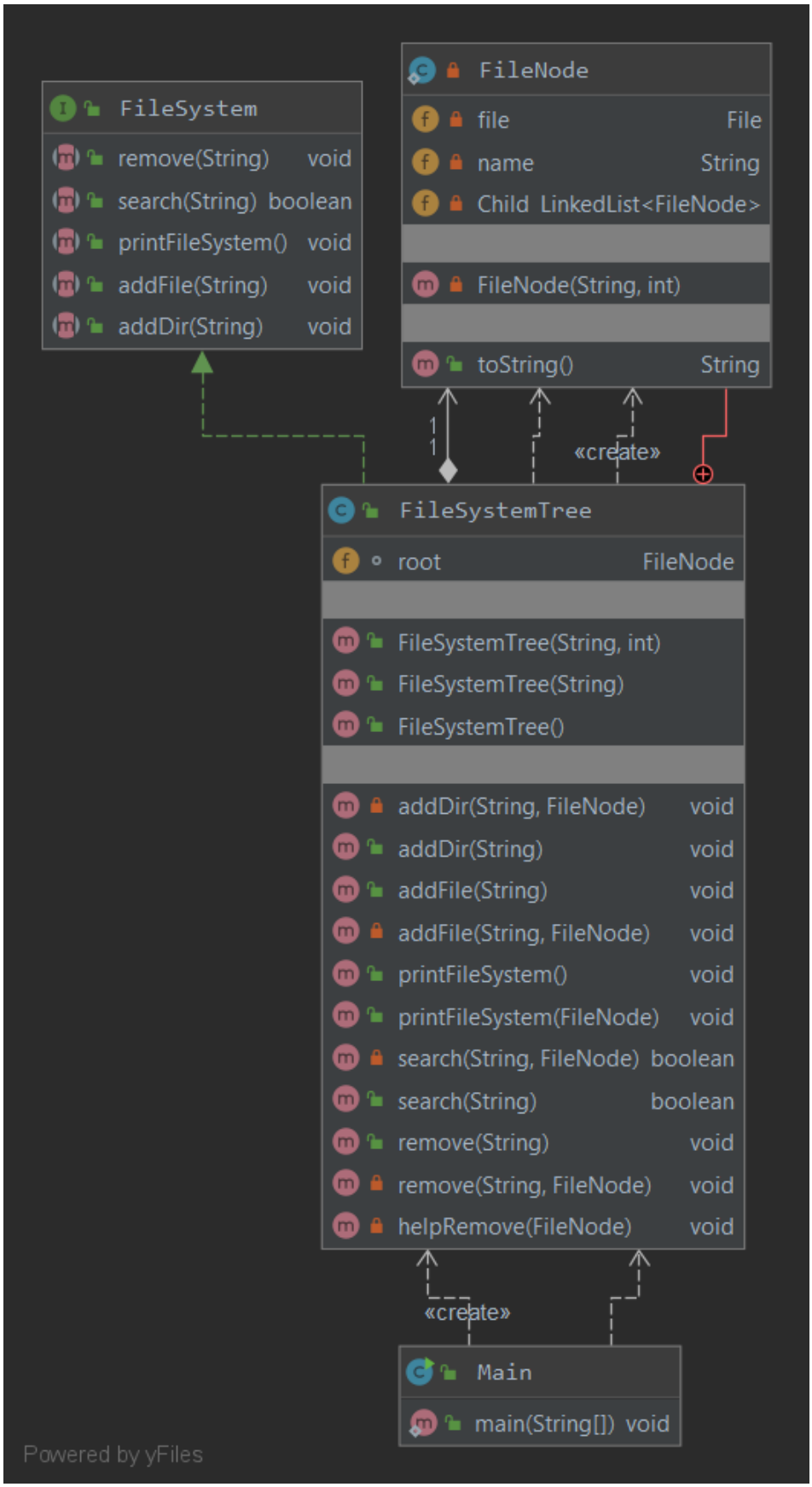
Fatih OĞUZ

151044025

PART1

USE CASE DIAGRAMS




FileSystem
remove(String) void
search(String) boolean
printFileSystem() void
addFile(String) void
addDir(String) void
FileNode
file File
name String
Child LinkedList<FileNode>
FileNode(String, int)
toString() String
1
1
«create»
FileSystemTree
root FileNode
FileSystemTree(String, int)
FileSystemTree(String)
FileSystemTree()
addDir(String, FileNode) void
addDir(String) void
addFile(String) void
addFile(String, FileNode) void
printFileSystem() void
printFileSystem(FileNode) void
search(String, FileNode) boolean
search(String) boolean
remove(String) void
remove(String, FileNode) void
helpRemove(FileNode) void
«create»
Main
main(String[]) void
Powered by yFiles

PROBLEM SOLUTION APPROACH

To install FileSystemTree, first you need to define an internal class that holds the file and name of file.

directory is required to start the system first. I distinguish between directory and file with a flag.

If flag equals 1 create directory otherwise file. Store linkedlist<FileNode> for children of directory

Because there is not limit for size of children.

If I delete other contacts or files under the directory while deleting it, I would expect the user to say yes if he is sure that he wants to show it and delete it.

If I delete the lesson iteratively, after deleting the files and directory below, I delete them in the desired directory. I had to use 2 auxiliary methods for this iterative process.

With this iterative transportation in the search method, I reach all regions and if I find it, I return it to true.

I use this search method to delete if the desired path does not exist and specify it to the user.

```
public void remove(String path){

    remove(path,root);
    if(!search(path)){
        System.out.println("the path cannot be found");
    }

}
```

```
String[] str = path.split("\\\\");
```

path bulma işleminde split metodu ile dosya adına göre yaparım.

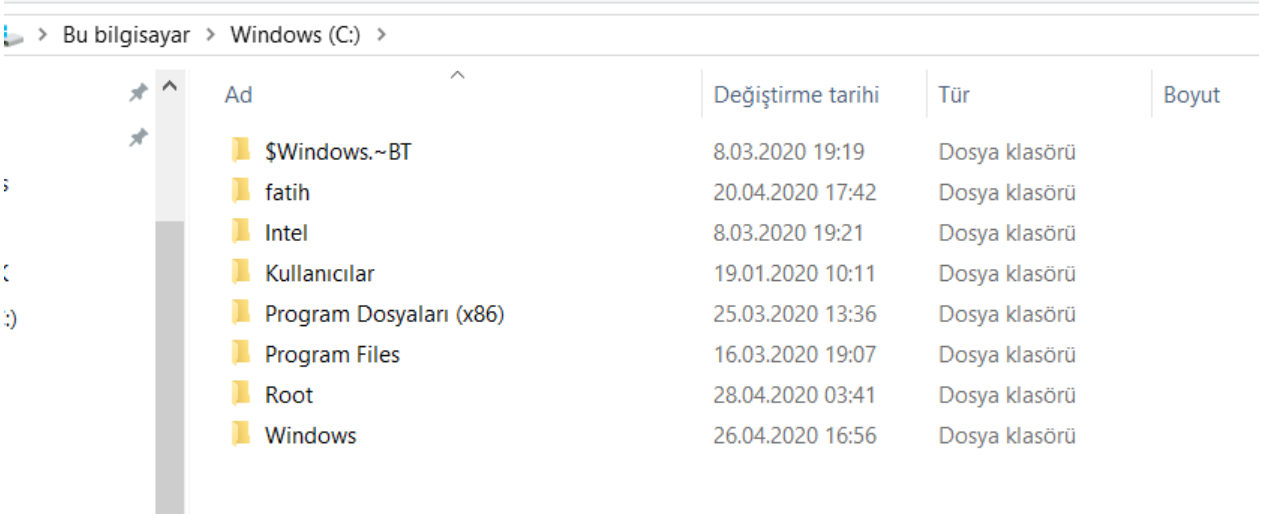
I do the recursive transportation process I wrote before

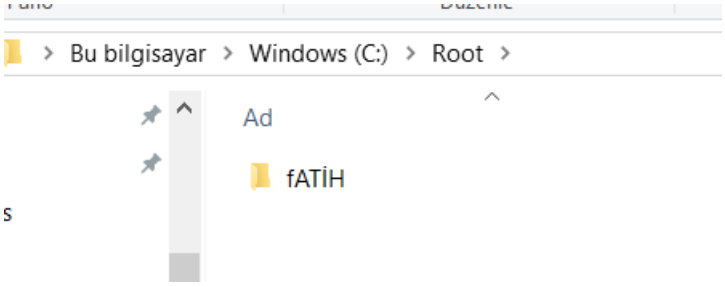
Test Case and Results

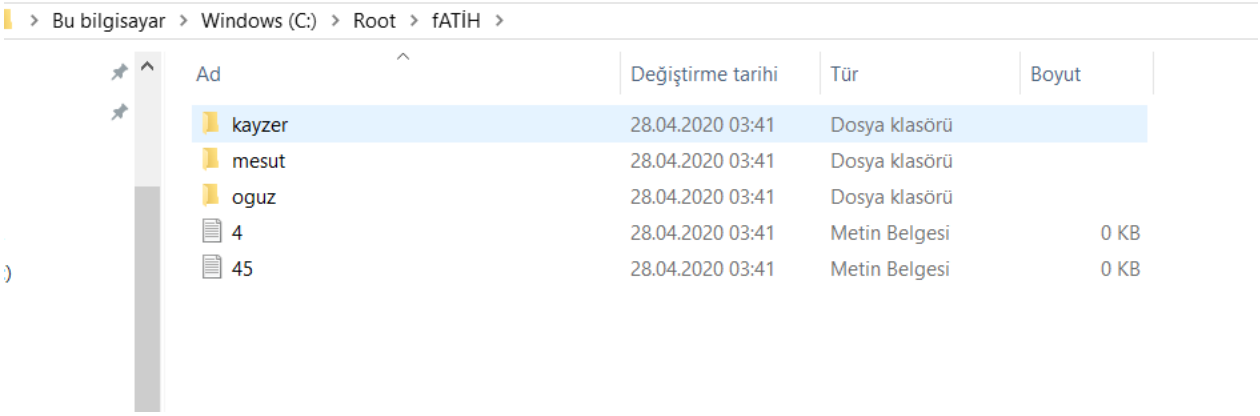
T1→ add metodu

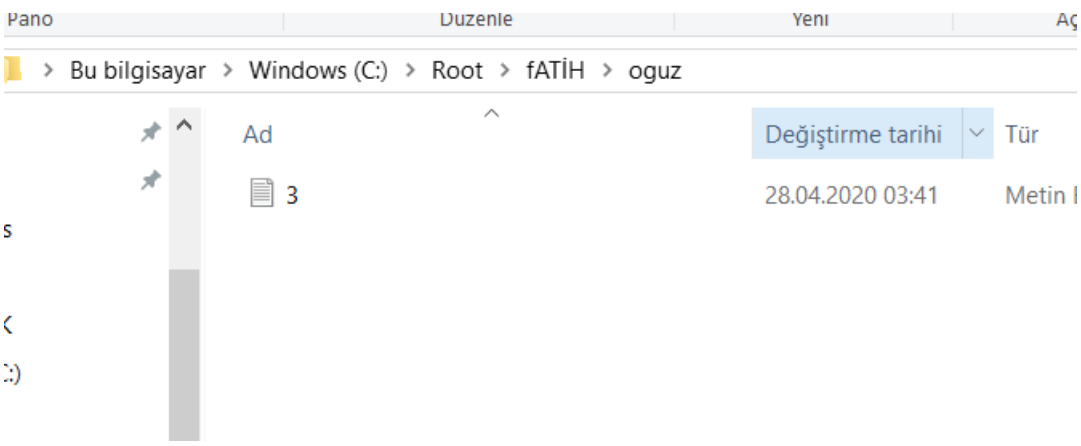
```java
fileSystemTree.addDir( path: "C:\\Root\\fATİH");
fileSystemTree.addDir( path: "C:\\Root\\fATİH\\oguz");
fileSystemTree.addDir( path: "C:\\Root\\fATİH\\mesut");
fileSystemTree.addDir( path: "C:\\Root\\fATİH\\kayzer");
fileSystemTree.addFile( path: "C:\\Root\\fATİH\\kayzer\\1.txt");
fileSystemTree.addFile( path: "C:\\Root\\fATİH\\mesut\\2.txt");
fileSystemTree.addFile( path: "C:\\Root\\fATİH\\oguz\\3.txt");
fileSystemTree.addFile( path: "C:\\Root\\fATİH\\4.txt");
fileSystemTree.addFile( path: "C:\\Root\\fATİH\\45.txt");
System.out.println("************************************");
fileSystemTree.printFileSystem();
System.out.println("************************************");
```

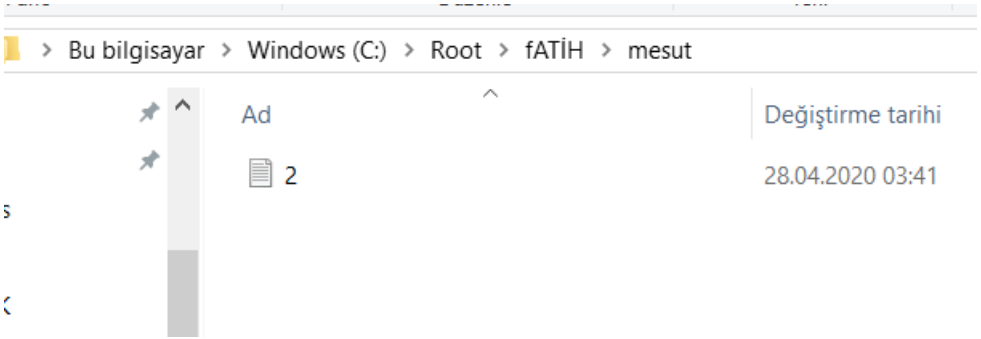
```
************************************
Root
fATİH
oguz
3.txt
mesut
2.txt
kayzer
1.txt
4.txt
45.txt
************************************
```

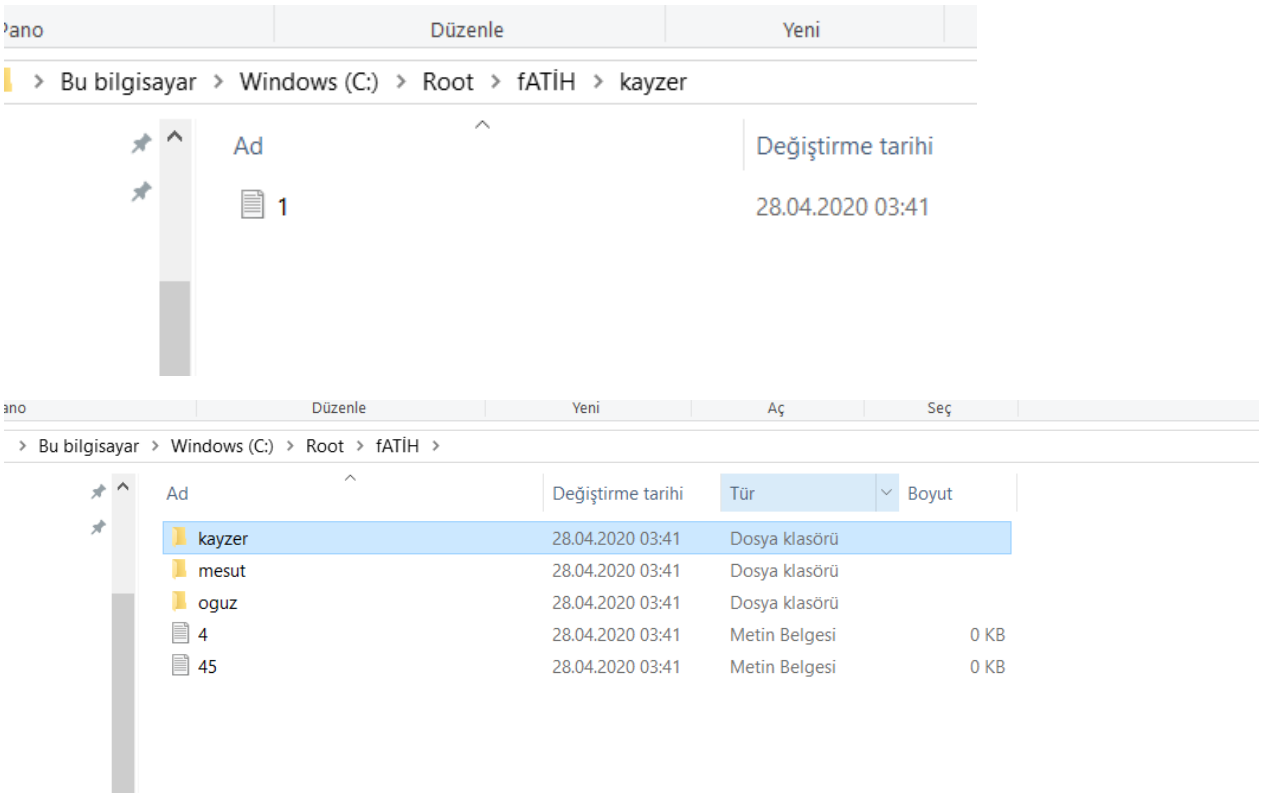



Bu bilgisayar › Windows (C:) › Root ›
Ad
fATİH


Bu bilgisayar › Windows (C:) › Root › fATİH ›
Ad
Değiştirme tarihi
Tür
Boyut
kayzer 28.04.2020 03:41 Dosya klasörü
mesut 28.04.2020 03:41 Dosya klasörü
oguz 28.04.2020 03:41 Dosya klasörü
4 28.04.2020 03:41 Metin Belgesi 0 KB
45 28.04.2020 03:41 Metin Belgesi 0 KB


Pano
Düzenle
Yeni
Aç
Bu bilgisayar › Windows (C:) › Root › fATİH › oguz
Ad
Değiştirme tarihi
Tür
3
28.04.2020 03:41


Bu bilgisayar › Windows (C:) › Root › fATİH › mesut
Ad
Değiştirme tarihi
2
28.04.2020 03:41



T2→remove

remove method after these additions

T2.1 if there is no file or directory in the given path

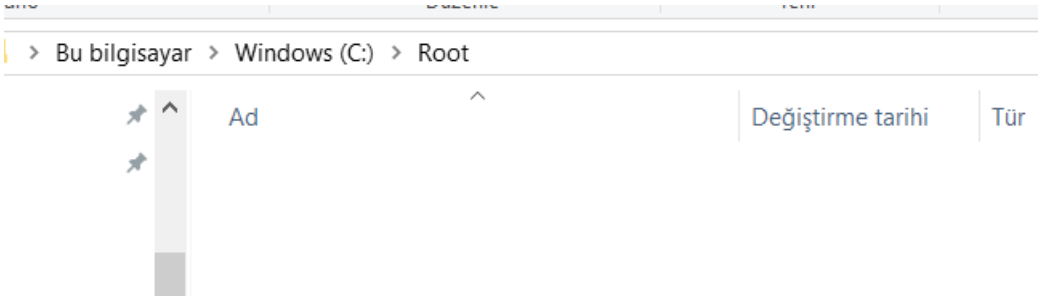
```
fileSystemTree.remove( path: "C:\\Root\\fATİH\\kayzer78");
```

```
Root
fATİH
oguz
3.txt
mesut
2.txt
kayzer
1.txt
4.txt
45.txt
**************************************
the path cannot be found
```

T2.2→If there is a file or directory under the directory

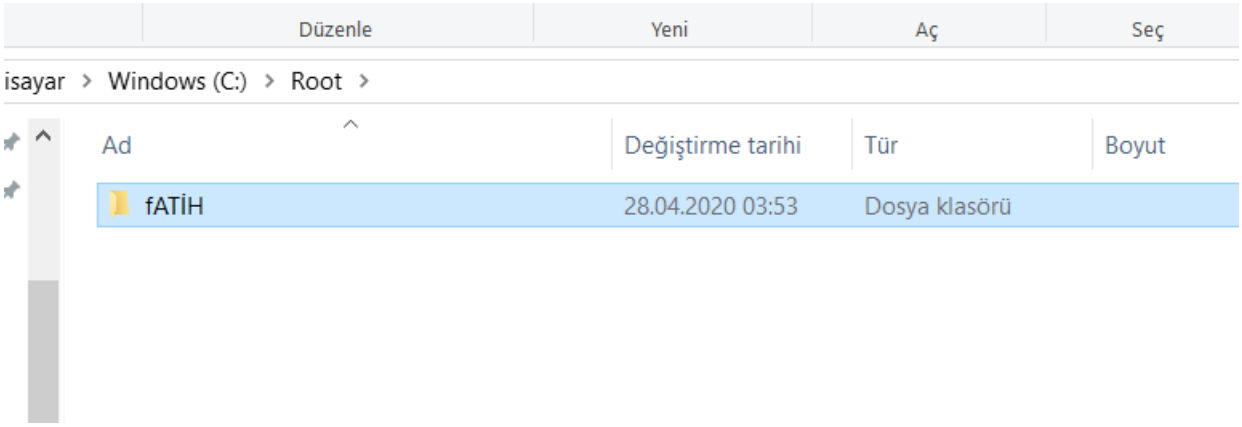
```
fileSystemTree.remove( path: "C:\\Root\\fATİH");
```

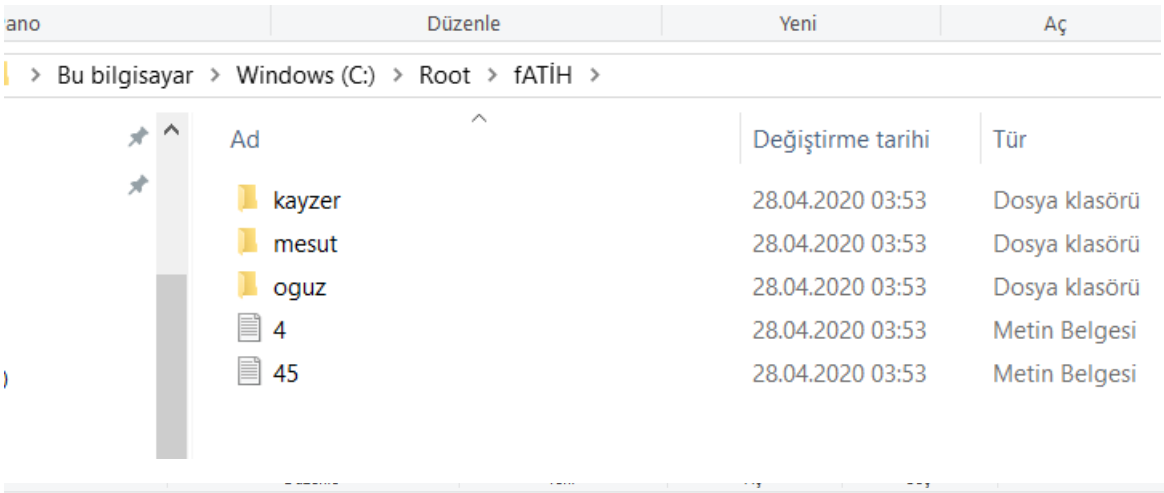
```
directory includes some other directories (or files)
fATİH
oguz
3.txt
mesut
2.txt
kayzer
1.txt
4.txt
45.txt
if you want to remove--> write yes otherwise write no
yes
delete
```

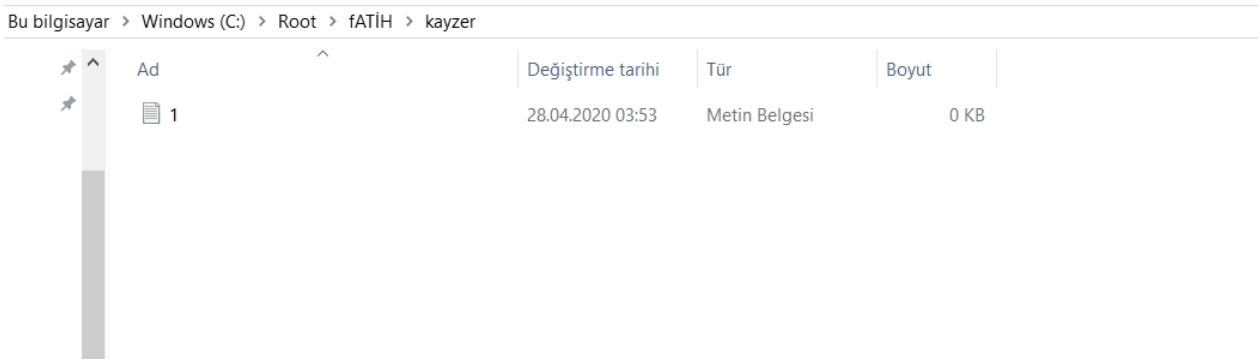


İf user say no

```
the path cannot be found
directory includes some other directories (or files)
fATİH
oguz
3.txt
mesut
2.txt
kayzer
1.txt
4.txt
45.txt
if you want to remove--> write yes otherwise write no
no
```



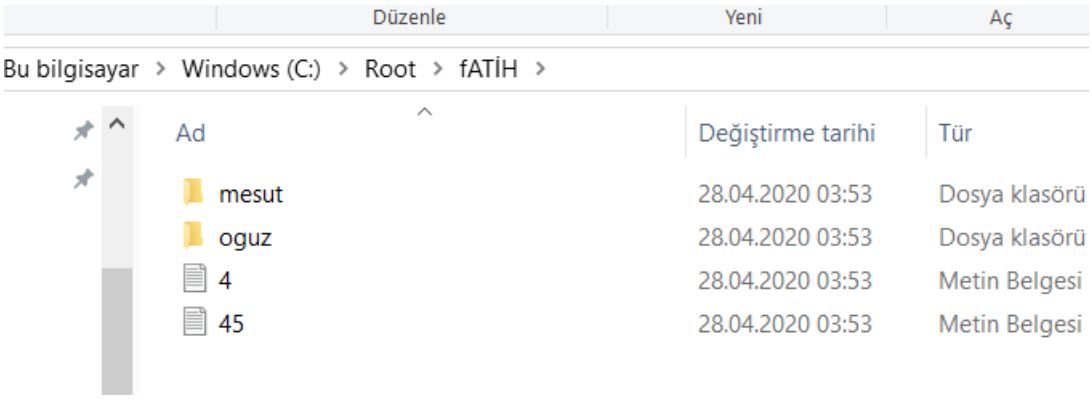




Example2

```
fileSystemTree.remove( path: "C:\\Root\\fATİH\\kayzer");
```

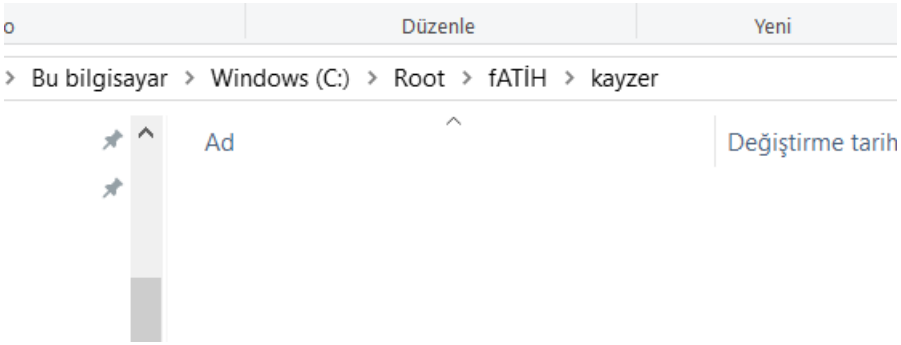
```
directory includes some other directories (or files)
kayzer
1.txt
if you want to remove--> write yes otherwise write no
yes
delete
```



T2.3→ If there is not a file or directory under the directory

```
fileSystemTree.remove( path: "C:\\Root\\fATİH\\kayzer\\1.txt");
```

```
delete
```



T3→ search metod

```
System.out.println(fileSystemTree.search( path: "C:\\Root\\fATİH\\kayzer"));
System.out.println(fileSystemTree.search( path: "C:\\Root\\fATİH"));
System.out.println(fileSystemTree.search( path: "C:\\Root\\fATİH\\kayzer46"));
```

```
true
true
false
```

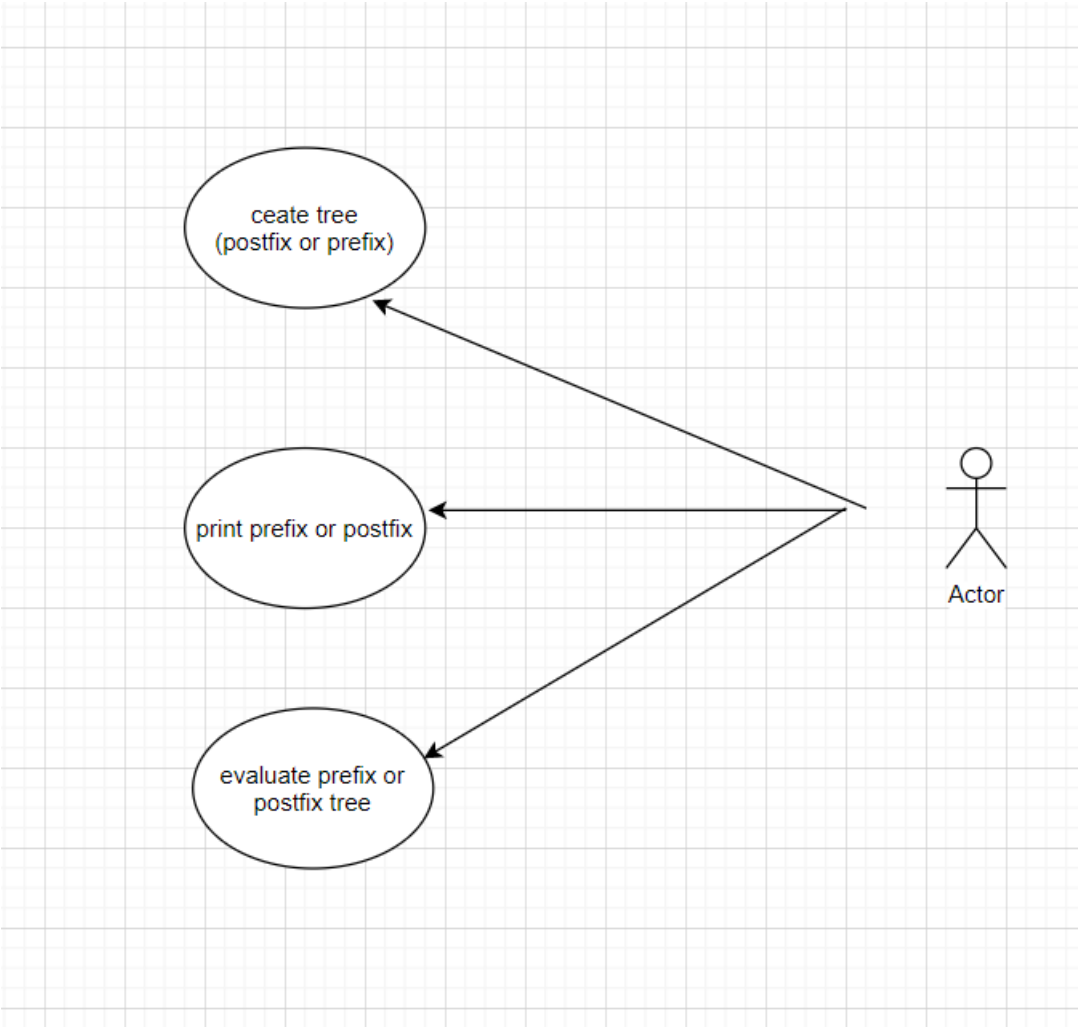
T4→printFileSystem metod

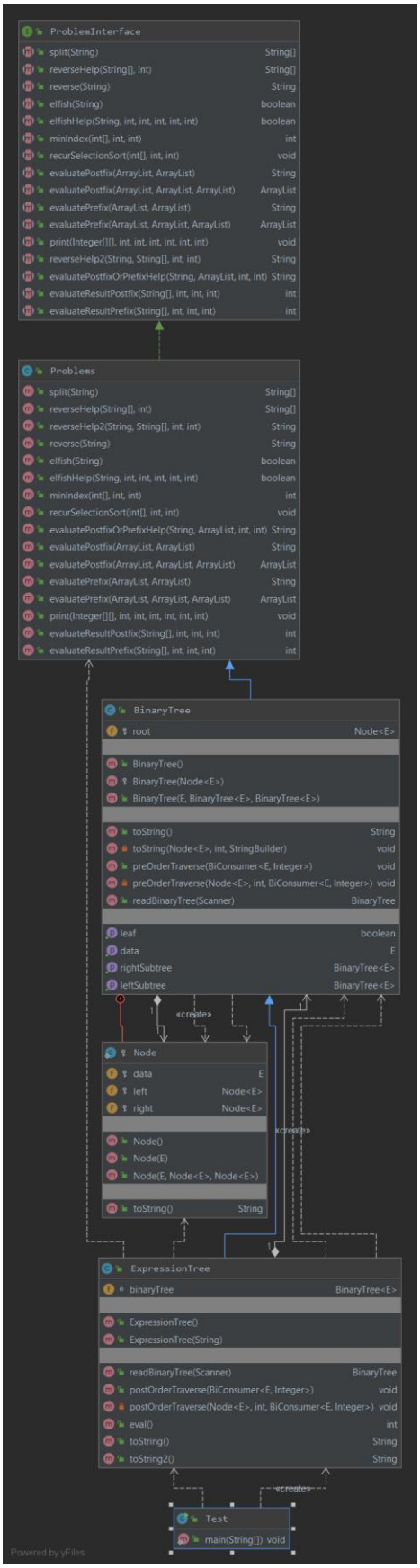
```
System.out.println("************************************");
fileSystemTree.printFileSystem();
System.out.println("************************************");
```

```
************************************
Root
fATİH
oguz
3.txt
mesut
2.txt
kayzer
1.txt
4.txt
45.txt
************************************
```

PART2

USE CASE DIAGRAMS




ProblemInterface
split(String) String[]
reverseHelp(String[], int) String[]
reverse(String) String
elfish(String) boolean
elfishHelp(String, int, int, int, int, int) boolean
minIndex(int[], int, int) int
recurSelectionSort(int[], int, int) void
evaluatePostfix(ArrayList, ArrayList) String
evaluatePostfix(ArrayList, ArrayList, ArrayList) ArrayList
evaluatePrefix(ArrayList, ArrayList) String
evaluatePrefix(ArrayList, ArrayList, ArrayList) ArrayList
print(Integer[][], int, int, int, int, int, int) void
reverseHelp2(String, String[], int, int) String
evaluatePostfixOrPrefixHelp(String, ArrayList, int, int) String
evaluateResultPostfix(String[], int, int, int) int
evaluateResultPrefix(String[], int, int, int) int
Problems
split(String) String[]
reverseHelp(String[], int) String[]
reverseHelp2(String, String[], int, int) String
reverse(String) String
elfish(String) boolean
elfishHelp(String, int, int, int, int, int) boolean
minIndex(int[], int, int) int
recurSelectionSort(int[], int, int) void
evaluatePostfixOrPrefixHelp(String, ArrayList, int, int) String
evaluatePostfix(ArrayList, ArrayList) String
evaluatePostfix(ArrayList, ArrayList, ArrayList) ArrayList
evaluatePrefix(ArrayList, ArrayList) String
evaluatePrefix(ArrayList, ArrayList, ArrayList) ArrayList
print(Integer[][], int, int, int, int, int, int) void
evaluateResultPostfix(String[], int, int, int) int
evaluateResultPrefix(String[], int, int, int) int
BinaryTree
root Node<E>
BinaryTree()
BinaryTree(Node<E>)
BinaryTree(E, BinaryTree<E>, BinaryTree<E>)
toString() String
toString(Node<E>, int, StringBuilder) void
preOrderTraverse(BiConsumer<E, Integer>) void
preOrderTraverse(Node<E>, int, BiConsumer<E, Integer>) void
readBinaryTree(Scanner) BinaryTree
leaf boolean
data E
rightSubtree BinaryTree<E>
leftSubtree BinaryTree<E>
1
«create»
Node
data E
left Node<E>
right Node<E>
Node()
Node(E)
Node(E, Node<E>, Node<E>)
toString() String
«create»
1
ExpressionTree
binaryTree BinaryTree<E>
ExpressionTree()
ExpressionTree(String)
readBinaryTree(Scanner) BinaryTree
postOrderTraverse(BiConsumer<E, Integer>) void
postOrderTraverse(Node<E>, int, BiConsumer<E, Integer>) void
eval() int
toString() String
toString2() String
«creates»
Test
main(String[]) void
Powered by yFiles

PROBLEM SOLUTION APPROACH

I should override readBinary method to create a tree in the constructor.

After this process, I have to test that the tree is done correctly with the postOrderTraverse and to String2 methods.

For the evaluate method, I should use the methods of problemInterface and problem class I wrote in the previous assignment.

Test Case and Results

T1 for postfix → crate tree print and evaluate

```
String str =
                " null\n" +
                 " null\n" +
                 "2\n"+
                 " null\n" +
                 " null\n" +
                 "4\n"+
                 "-\n"+
                 " null\n" +
                 " null\n" +
                 "24\n"+
                 " null\n" +
                 " null\n"+
                 "2\n"+
                 "/\n"+
                 "+\n"+
                 " null\n" +
                 " null\n" +
                 "50\n"+
                 " null\n" +
                 " null\n" +
                 "40\n"+
                 "-\n"+
                 " null\n" +
                 " null\n" +
                 "4\n"+
                 " null\n" +
                 " null\n"+
                 "2\n"+
                 "/\n"+
                 "*\n"+
                 "-\n"+
```

```
" null\n" +
" null\n" +
"5\n"+
" null\n" +
" null\n" +
"45\n"+
"+\n"+
" null\n" +
" null\n" +
"40\n"+
" null\n" +
" null\n"+
"2\n"+
"-\n"+
"+\n"+
" null\n" +
" null\n" +
"8\n"+
" null\n" +
" null\n" +
"4\n"+
"/\n"+
" null\n" +
" null\n" +
"4\n"+
" null\n" +
" null\n"+
"2\n"+
"+\n"+
"*\n"+
"-\n"+
"+\n";
```

```
ExpressionTree<String> ex = new ExpressionTree<>(str);
System.out.println(ex.toString2());
System.out.println(ex.eval());;

System.out.println("*****************************************");
```

```
null null 2 null null 4 - null null 24 null null 2 / + null null 50 null null 40 - null null 4 null null 2 / * - null null 5 null null 45 + null null 40 null null 2 - + null null 8 null null 4 /
2 - 4 = -2
24 / 2 = 12
-2 + 12 = 10
50 - 40 = 10
4 / 2 = 2
10 * 2 = 20
10 - 20 = -10
5 + 45 = 50
40 - 2 = 38
50 + 38 = 88
8 / 4 = 2
4 + 2 = 6
2 * 6 = 12
88 - 12 = 76
-10 + 76 = 66
```

```
String str1 =
        " null\n" +
        " null\n" +
        "50\n"+
        " null\n" +
        " null\n" +
        "2\n"+
        "/\n"+
        " null\n" +
        " null\n" +
        "40\n"+
        " null\n" +
        " null\n"+
        "2\n"+
        "/\n"+
        "+\n"+
    " null\n" +
    " null\n" +
    "5\n"+
    " null\n" +
    " null\n" +
    "4\n"+
    "*\n"+
    " null\n" +
    " null\n" +
    "4\n"+
    " null\n" +
    " null\n"+
    "2\n"+
    "/\n"+
    "*\n"+
    "+\n";
```

```java
ExpressionTree<String> ex1 = new ExpressionTree<>(str1);
System.out.println(ex1.toString2());
System.out.println(ex1.eval());;

System.out.println("*********************************************");
```

```
*********************************************
null null 50 null null 2 / null null 40 null null 2 / + null null 5 null null 4 * null null 4 null null 2 / * +
50 / 2 = 25
40 / 2 = 20
25 + 20 = 45
5 * 4 = 20
4 / 2 = 2
20 * 2 = 40
45 + 40 = 85
```

```java
String str2 =
        " null\n" +
        " null\n" +
        "5\n"+
        " null\n" +
        " null\n" +
        "4\n"+
        "+\n"+
        " null\n" +
        " null\n" +
        "4\n"+
        " null\n" +
        " null\n"+
        "2\n"+
        "/\n"+
        "*\n";
```

```
ExpressionTree<String> ex2 = new ExpressionTree<>(str2);
System.out.println(ex2.toString2());
System.out.println(ex2.eval());

System.out.println("*****************************************");
```

```
*****************************************
null null 5 null null 4 + null null 4 null null 2 / *
5 + 4 = 9
4 / 2 = 2
9 * 2 = 18
```

T2 for prefix → crate tree print and evaluate

```
String str3 ="*\n"+
        "+\n"+
        "5\n"+
        " null\n" +
        " null\n" +
        "4\n"+
        " null\n" +
        " null\n" +
        "/\n"+
        "10\n"+
        " null\n" +
        " null\n" +
        "2\n"+
        " null\n" +
        " null\n" ;
```

```java
ExpressionTree<String> ex3=new ExpressionTree<>(str3);
System.out.println(ex3.toString());
System.out.println(ex3.eval());


System.out.println("*********************************************");
```

```
*********************************************
* + 5 null null 4 null null / 10 null null 2 null null
10 / 2 = 5
4 + 5 = 9
9 * 5 = 45
```

```
String str4 = "+\n"+
        "*\n"+
        "+\n"+
        "5\n"+
        " null\n" +
        " null\n" +
        "4\n"+
        " null\n" +
        " null\n" +
        "/\n"+
        "16\n"+
        " null\n" +
        " null\n" +
        "2\n"+
        " null\n" +
        " null\n" +
        "/\n"+
        "+\n"+
        "24\n"+
        " null\n" +
        " null\n" +
        "40\n"+
        " null\n" +
        " null\n" +
        "/\n"+
        "80\n"+
        " null\n" +
        " null\n" +
        "20\n"+
        " null\n" +
        " null\n";
main()
```

```
    System.out.println("*********************************************");

    ExpressionTree<String> ex4 = new ExpressionTree<>(str4);
    System.out.println(ex4.toString());
    System.out.println(ex4.eval());
```

```
*********************************************
+ * + 5 null null 4 null null / 16 null null 2 null null / + 24 null null 40 null null / 80 null null 20 null null
80 / 20 = 4
40 + 24 = 64
64 / 4 = 16
16 / 2 = 8
4 + 5 = 9
9 * 8 = 72
16 + 72 = 88
```

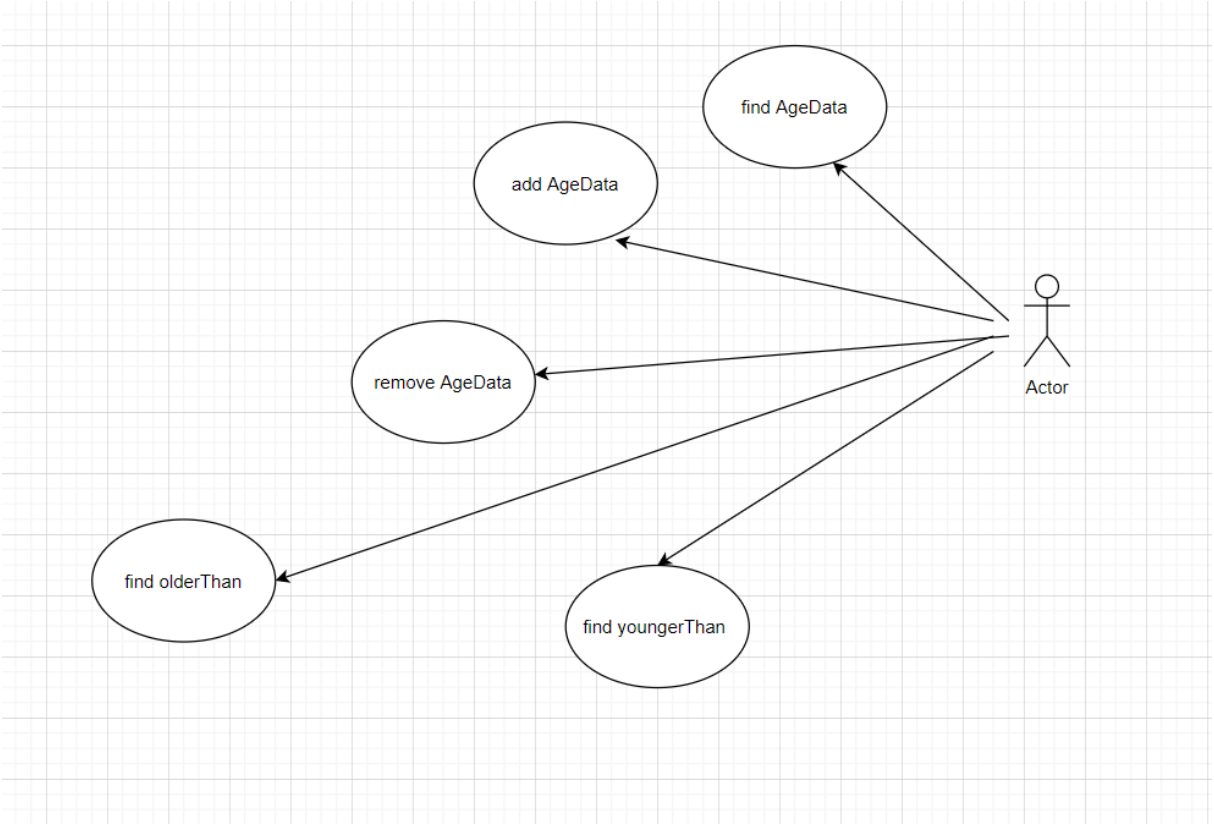
```
String str5 ="+\n"+
        "*\n"+
        "-\n"+
        "+\n"+
        "5\n"+
        " null\n" +
        " null\n" +
        "4\n"+
        " null\n" +
        " null\n" +
        "/\n"+
        "30\n"+
        " null\n" +
        " null\n" +
        "2\n"+
        " null\n" +
        " null\n"+
        "/\n"+
        "+\n"+
        "4\n"+
        " null\n" +
        " null\n" +
        "2\n"+
        " null\n" +
        " null\n" +
        "-\n"+
        "8\n"+
        " null\n" +
        " null\n" +
        "2\n"+
```

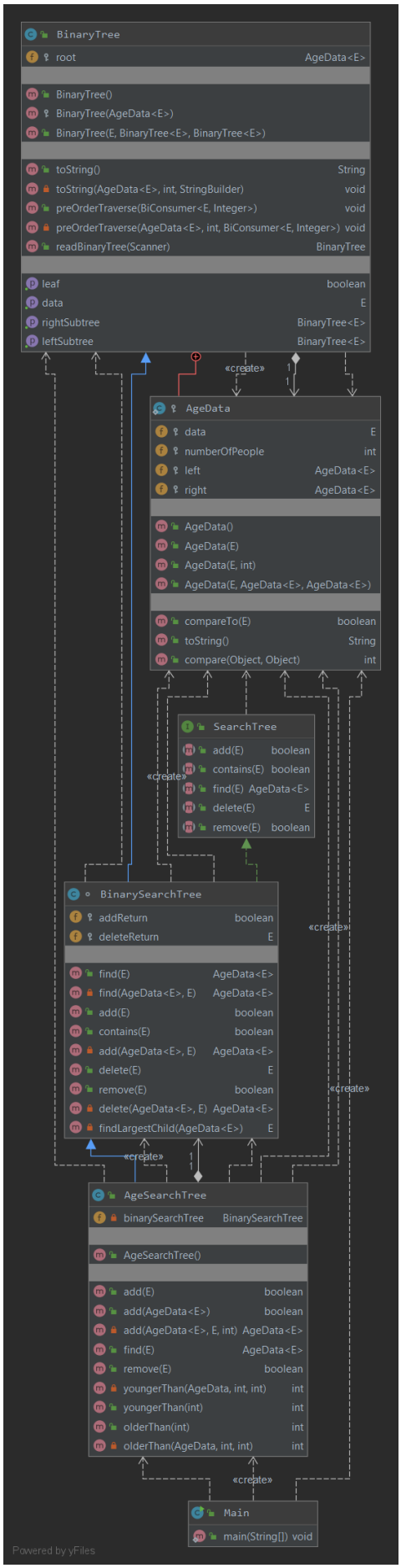
```
            " null\n"+
            "+\n"+
            "/\n"+
            "-\n"+
            "78\n"+
            " null\n" +
            " null\n" +
            "40\n"+
            " null\n" +
            " null\n" +
            "+\n"+
            "12\n"+
            " null\n" +
            " null\n" +
            "26\n"+
            " null\n" +
            " null\n"+
            "+\n"+
            "-\n"+
            "86\n"+
            " null\n" +
            " null\n" +
            "22\n"+
            " null\n" +
            " null\n" +
            "*\n"+
            "2\n"+
            " null\n" +
            " null\n" +
            "2\n"+
            " null\n" +
            " null\n";
```

```
System.out.println("*********************************************");

ExpressionTree<String> ex5 = new ExpressionTree<>(str5);
System.out.println(ex5.toString());
System.out.println(ex5.eval());
```

```
**************************************************
+ * - + 5 null null 4 null null / 30 null null 2 null null / + 4 null null 2 null null - 8 null null 2 null null + / - 78 null null 40 null null + 12 null null 26 null null + - 86 null null 22 null
2 * 2 = 4
86 - 22 = 64
4 + 64 = 68
26 + 12 = 38
78 - 40 = 38
38 / 38 = 1
68 + 1 = 69
8 - 2 = 6
2 + 4 = 6
6 / 6 = 1
30 / 2 = 15
4 + 5 = 9
9 - 15 = -6
-6 * 1 = -6
69 + -6 = 63
```

USE CASE DIAGRAMS




BinaryTree
root AgeData<E>
BinaryTree()
BinaryTree(AgeData<E>)
BinaryTree(E, BinaryTree<E>, BinaryTree<E>)
toString() String
toString(AgeData<E>, int, StringBuilder) void
preOrderTraverse(BiConsumer<E, Integer>) void
preOrderTraverse(AgeData<E>, int, BiConsumer<E, Integer>) void
readBinaryTree(Scanner) BinaryTree
leaf boolean
data E
rightSubtree BinaryTree<E>
leftSubtree BinaryTree<E>
«create»
1
1
AgeData
data E
numberOfPeople int
left AgeData<E>
right AgeData<E>
AgeData()
AgeData(E)
AgeData(E, int)
AgeData(E, AgeData<E>, AgeData<E>)
compareTo(E) boolean
toString() String
compare(Object, Object) int
SearchTree
add(E) boolean
contains(E) boolean
find(E) AgeData<E>
delete(E) E
remove(E) boolean
«create»
BinarySearchTree
addReturn boolean
deleteReturn E
find(E) AgeData<E>
find(AgeData<E>, E) AgeData<E>
add(E) boolean
contains(E) boolean
add(AgeData<E>, E) AgeData<E>
delete(E) E
remove(E) boolean
delete(AgeData<E>, E) AgeData<E>
findLargestChild(AgeData<E>) E
«create»
«create»
«create»
1
1
AgeSearchTree
binarySearchTree BinarySearchTree
AgeSearchTree()
add(E) boolean
add(AgeData<E>) boolean
add(AgeData<E>, E, int) AgeData<E>
find(E) AgeData<E>
remove(E) boolean
youngerThan(AgeData, int, int) int
youngerThan(int) int
olderThan(int) int
olderThan(AgeData, int, int) int
«create»
Main
main(String[]) void
Powered by yFiles

PROBLEM SOLUTION APPROACH

Number Of People field must be added to the AgeData inner class to keep the number of people of the same age. AgeSearchTree class should inherit BinarySearchTree class to facilitate methods.

In the add method, in case of trying to add the same age, the paramter is checked with the compare method, and the same age is increased by the numberOfPeople field as the person of the coming age. The remove method takes the opposite of the method of the add method. The whole tree is traverse for the find method. return if element is found. Similar traverse is done in youngerThan and olderThan methods and the total number of people is returned.

Test Case and Results

T1-) add methods

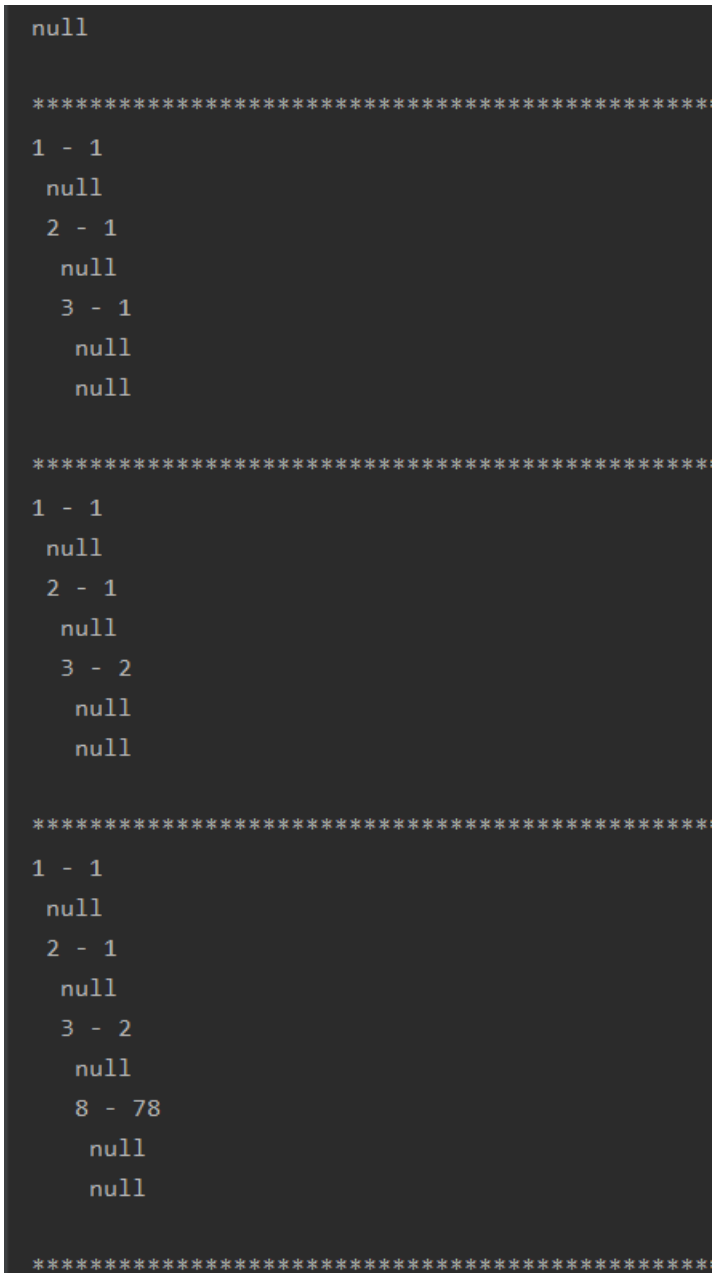
```
ageTree.add(new BinaryTree.AgeData<>(10));
ageTree.add(new BinaryTree.AgeData<>(20));
ageTree.add(new BinaryTree.AgeData<>(5));
ageTree.add(new BinaryTree.AgeData<>(15));
ageTree.add(new BinaryTree.AgeData<>(10));
System.out.println(ageTree+"\n*********************************************************************************");
```

```
10 - 2
 5 - 1
  null
  null
 20 - 1
  15 - 1
   null
   null
  null

*********************************************************************************
```

```
System.out.println(ageTree+"\n*********************************************************************************");
ageTree.add(1);
ageTree.add(2);
ageTree.add(3);
System.out.println(ageTree+"\n*********************************************************************************");
ageTree.add(3);
System.out.println(ageTree+"\n*********************************************************************************");
```

```
********************************************************************************
1 - 1
 null
 2 - 1
  null
  3 - 1
   null
   null

********************************************************************************
1 - 1
 null
 2 - 1
  null
  3 - 2
   null
   null
```

```
ageTree.add(1);
ageTree.add(2);
ageTree.add(3);
System.out.println(ageTree+"\n********************************************************************************");
ageTree.add(3);
System.out.println(ageTree+"\n********************************************************************************");
ageTree.add(new BinaryTree.AgeData<Integer>( data: 8, size: 78));
System.out.println(ageTree+"\n********************************************************************************");
ageTree.add(new BinaryTree.AgeData<>( data: 2, size: 23));
System.out.println(ageTree+"\n********************************************************************************");
```



İncrement people

```
System.out.println(ageTree+"\n**********");
ageTree.add(new BinaryTree.AgeData<>( data: 2, size: 23));
System.out.println(ageTree+"\n**********");
```

```
1 - 1
 null
 2 - 1
  null
  3 - 2
   null
   8 - 78
    null
    null

*********************************
1 - 1
 null
 2 - 24
  null
  3 - 2
   null
   8 - 78
    null
    null

*********************************
```

T2 olderThan , youngerThan and find metods

```
System.out.println(ageTree+"\n*********************************************************************************");
System.out.println("youngerThan(15) --> " +  ageTree.youngerThan( item: 15));
System.out.println("olderThan(15)   " + ageTree.olderThan( item: 15));
System.out.println("find(10) --> " + ageTree.find( target: 10));
System.out.println("*********************************************************************************\nremove all age data");
```

```
*********************************************************************************
youngerThan(15) --> 3
olderThan(15)   1
find(10) --> 10 - 2
*********************************************************************************
```

```
ageTree.add(1);
ageTree.add(2);
ageTree.add(3);
System.out.println(ageTree+"\n*************************************************************************");
ageTree.add(3);
System.out.println(ageTree+"\n*************************************************************************");
ageTree.add(new BinaryTree.AgeData<Integer>( data: 8, size: 78));
System.out.println(ageTree+"\n*************************************************************************");
ageTree.add(new BinaryTree.AgeData<>( data: 2, size: 23));
System.out.println(ageTree+"\n*************************************************************************");

System.out.println("youngerThan(3) --> "+  ageTree.youngerThan( item: 3));
System.out.println("olderThan(2) --> "+  ageTree.olderThan( item: 2));
```

```
*************************************************************************
1 - 1
 null
 2 - 24
  null
  3 - 2
   null
   8 - 78
    null
    null

*************************************************************************
youngerThan(3) --> 25
olderThan(2) --> 80
```

T3 remove metods

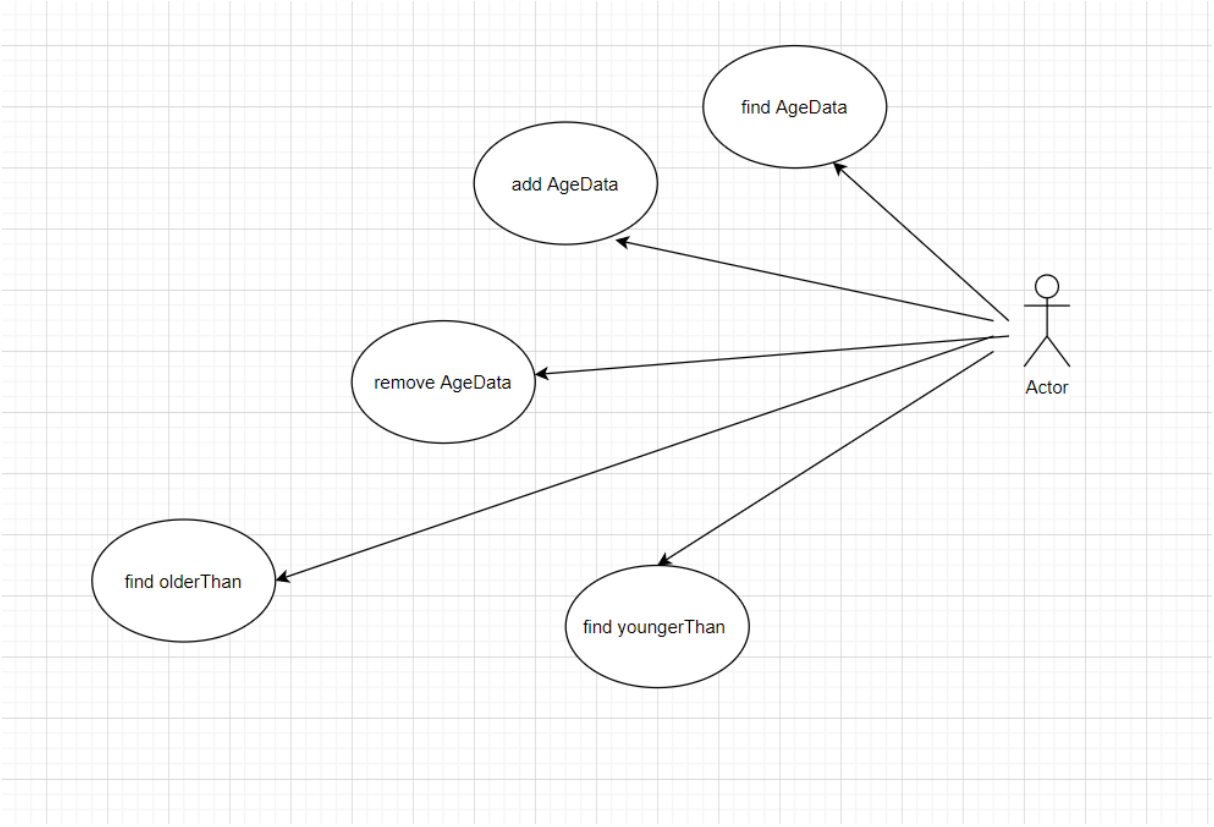
```
System.out.println(ageTree+"\n*************************************************************************");
ageTree.remove( target: 10);
ageTree.remove( target: 20);
ageTree.remove( target: 5);
System.out.println(ageTree+"\n*************************************************************************");
```

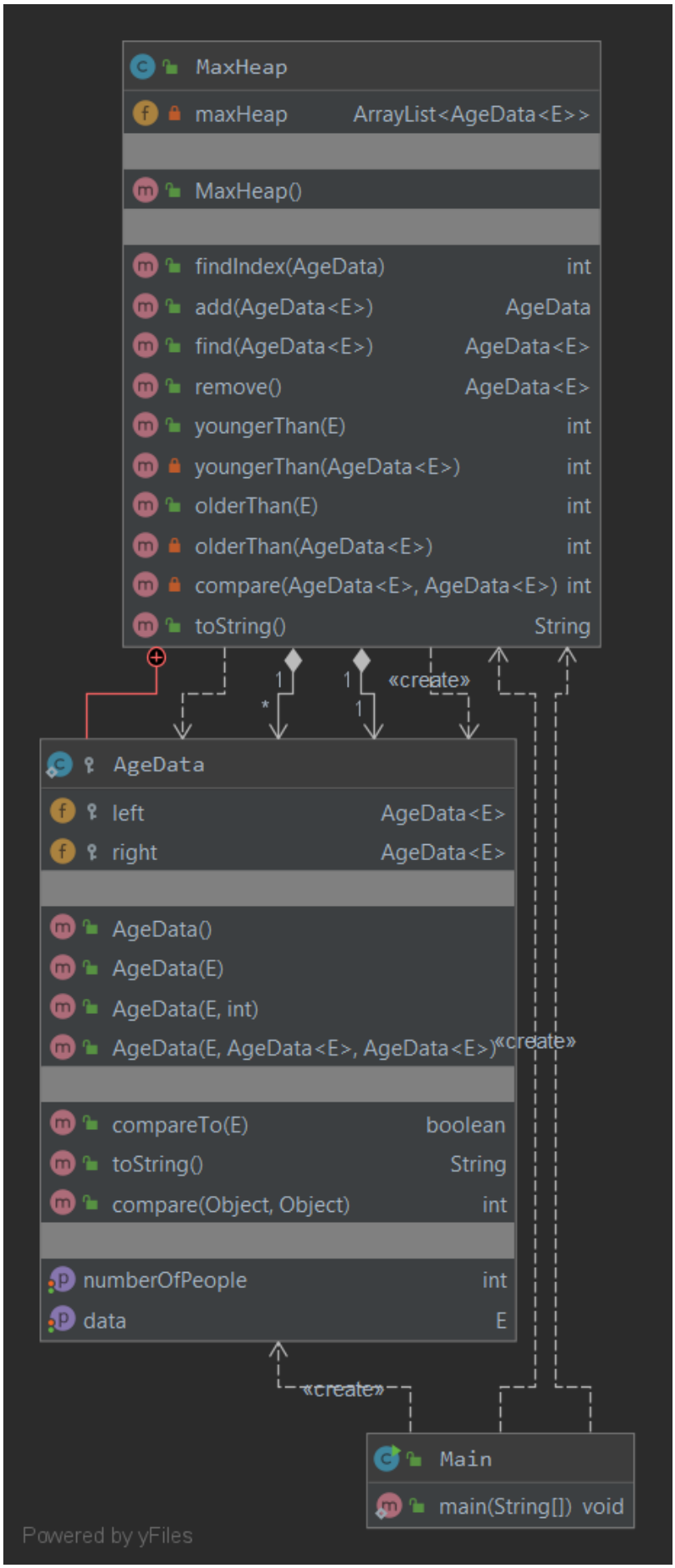
```
*************************************************************************
10 - 1
 null
 15 - 1
  null
  null
```

```
System.out.println(ageTree+"\n*************************************************************************");
ageTree.remove( target: 10);
ageTree.remove( target: 20);
ageTree.remove( target: 5);
System.out.println(ageTree+"\n*************************************************************************");
ageTree.remove( target: 15);
ageTree.remove( target: 10);
System.out.println(ageTree+"\n*************************************************************************");
```

```
**********************************************************************************
10 - 1
 null
 15 - 1
  null
  null

**********************************************************************************
null

**********************************************************************************
```

USECASE DIAGRAMS




MaxHeap
maxHeap ArrayList<AgeData<E>>
MaxHeap()
findIndex(AgeData) int
add(AgeData<E>) AgeData
find(AgeData<E>) AgeData<E>
remove() AgeData<E>
youngerThan(E) int
youngerThan(AgeData<E>) int
olderThan(E) int
olderThan(AgeData<E>) int
compare(AgeData<E>, AgeData<E>) int
toString() String
1
*
1
1
«create»
AgeData
left AgeData<E>
right AgeData<E>
AgeData()
AgeData(E)
AgeData(E, int)
AgeData(E, AgeData<E>, AgeData<E>)
«create»
compareTo(E) boolean
toString() String
compare(Object, Object) int
numberOfPeople int
data E
«create»
Main
main(String[]) void
Powered by yFiles

PROBLEM SOLUTION APPROACH

To keep heap, an arrayList must be made. And this arrayList should be of the AgeData type. AgeData should carry the age and the number of people that have this age.

When adding AgeData, if there is no member of that age, AgeData is created, if there are any, the number of people is increased. Remove operation is performed similarly.

Find youngerThan and olderThan methods can be done by looping in arrayList.

This will have an algorithm of type O (n).

Test Cases and Results

T1: the ranking should be done by taking the number of people key

```
maxHeap.add(new MaxHeap.AgeData<>(26));
maxHeap.add(new MaxHeap.AgeData<>(6));
maxHeap.add(new MaxHeap.AgeData<>(18));
maxHeap.add(new MaxHeap.AgeData<>(20));
maxHeap.add(new MaxHeap.AgeData<>(28));
maxHeap.add(new MaxHeap.AgeData<>(39));
maxHeap.add(new MaxHeap.AgeData<>(66));
System.out.println(maxHeap);
```

```
MaxHeap--->
 26 1
 6 1
 18 1
 20 1
 28 1
 39 1
 66 1
```

```
maxHeap.add(new MaxHeap.AgeData<>(26));
maxHeap.add(new MaxHeap.AgeData<>(6));
maxHeap.add(new MaxHeap.AgeData<>(18));
maxHeap.add(new MaxHeap.AgeData<>(20));
maxHeap.add(new MaxHeap.AgeData<>(28));
maxHeap.add(new MaxHeap.AgeData<>(39));
maxHeap.add(new MaxHeap.AgeData<>(66));
System.out.println(maxHeap);
maxHeap.add(new MaxHeap.AgeData<>(39));
System.out.println(maxHeap);
```

```
MaxHeap--->
 39 2
 6 1
 26 1
 20 1
 28 1
 18 1
 66 1
```

```java
maxHeap.add(new MaxHeap.AgeData<>(26));
maxHeap.add(new MaxHeap.AgeData<>(6));
maxHeap.add(new MaxHeap.AgeData<>(18));
maxHeap.add(new MaxHeap.AgeData<>(20));
maxHeap.add(new MaxHeap.AgeData<>(28));
maxHeap.add(new MaxHeap.AgeData<>(39));
maxHeap.add(new MaxHeap.AgeData<>(66));
System.out.println(maxHeap);
maxHeap.add(new MaxHeap.AgeData<>(39));
System.out.println(maxHeap);
maxHeap.add(new MaxHeap.AgeData<>(37));
maxHeap.add(new MaxHeap.AgeData<>(29));
maxHeap.add(new MaxHeap.AgeData<>(76));
maxHeap.add(new MaxHeap.AgeData<>(32));
maxHeap.add(new MaxHeap.AgeData<>(74));
maxHeap.add(new MaxHeap.AgeData<>(89));
maxHeap.add(new MaxHeap.AgeData<>(8));
maxHeap.add(new MaxHeap.AgeData<>(74));
maxHeap.add(new MaxHeap.AgeData<>(89));
maxHeap.add(new MaxHeap.AgeData<>(8));
maxHeap.add(new MaxHeap.AgeData<>( data: 0, size: 89));
System.out.println(maxHeap);
```

```
MaxHeap--->
 0 89
 6 1
 39 2
 20 1
 28 1
 89 2
 74 2
 37 1
 29 1
 76 1
 32 1
 18 1
 26 1
 66 1
 8 2
```

T2 youngerThan ,olderThan and find methods

```
MaxHeap--->
 0 89
 6 1
 39 2
 20 1
 28 1
 89 2
 74 2
 37 1
 29 1
 76 1
 32 1
 18 1
 26 1
 66 1
 8 2

youngerThan(29) --> 96
olderThan(32) --> 9
find(37) --> 37 1
remove all AgeData
```

```
System.out.println(maxHeap);
System.out.println("youngerThan(29) --> " + maxHeap.youngerThan( item: 29));
System.out.println("olderThan(32) --> " + maxHeap.olderThan( item: 32));
System.out.println("find(37) --> " + maxHeap.find(new MaxHeap.AgeData<>(37)));
```

T2.2)Does the find method give the correct number of people

```
System.out.println("find(0) --> " + maxHeap.find(new MaxHeap.AgeData<>(0)));
```

```
find(0) --> 0 89
```

T3-) Does the remove method work correctly

```
maxHeap.remove();
System.out.println(maxHeap);
```

```
MaxHeap--
 8 2
 6 1
 39 2
 20 1
 28 1
 89 2
 74 2
 37 1
 29 1
 76 1
 32 1
 18 1
 26 1
 66 1
```

```
maxHeap.remove();
System.out.println(maxHeap);
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
System.out.println(maxHeap);
```

```
MaxHeap---
 29 1
 6 1
 18 1
 20 1
 28 1
 66 1
 26 1
 37 1
```

```
maxHeap.remove();
System.out.println(maxHeap);
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
System.out.println(maxHeap);
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
maxHeap.remove();
System.out.println("remove all AgeData");
```

```
remove all AgeData
MaxHeap--->
```

T4 Mix

```
System.out.println("remove all AgeData");
System.out.println(maxHeap);
maxHeap.add(new MaxHeap.AgeData<>(10));
maxHeap.add(new MaxHeap.AgeData<>(5));
maxHeap.add(new MaxHeap.AgeData<>(70));
maxHeap.add(new MaxHeap.AgeData<>(10));
maxHeap.add(new MaxHeap.AgeData<>(50));
maxHeap.add(new MaxHeap.AgeData<>(50));
maxHeap.add(new MaxHeap.AgeData<>(50));
maxHeap.add(new MaxHeap.AgeData<>(5));
maxHeap.add(new MaxHeap.AgeData<>(15));
System.out.println(maxHeap);
System.out.println("youngerThan(70) --> " + maxHeap.youngerThan( item: 70));
System.out.println("olderThan(50) --> " + maxHeap.olderThan( item: 50) );
System.out.println(maxHeap);
System.out.println("remove");
maxHeap.remove();
System.out.println(maxHeap);
```

```
MaxHeap--->
 50 3
 10 2
 70 1
 5 2
 15 1

youngerThan(70) --> 8
olderThan(50) --> 1
MaxHeap--->
 50 3
 10 2
 70 1
 5 2
 15 1

remove
MaxHeap--->
 10 2
 5 2
 70 1
 15 1
```