# GTU Department of Computer Engineering

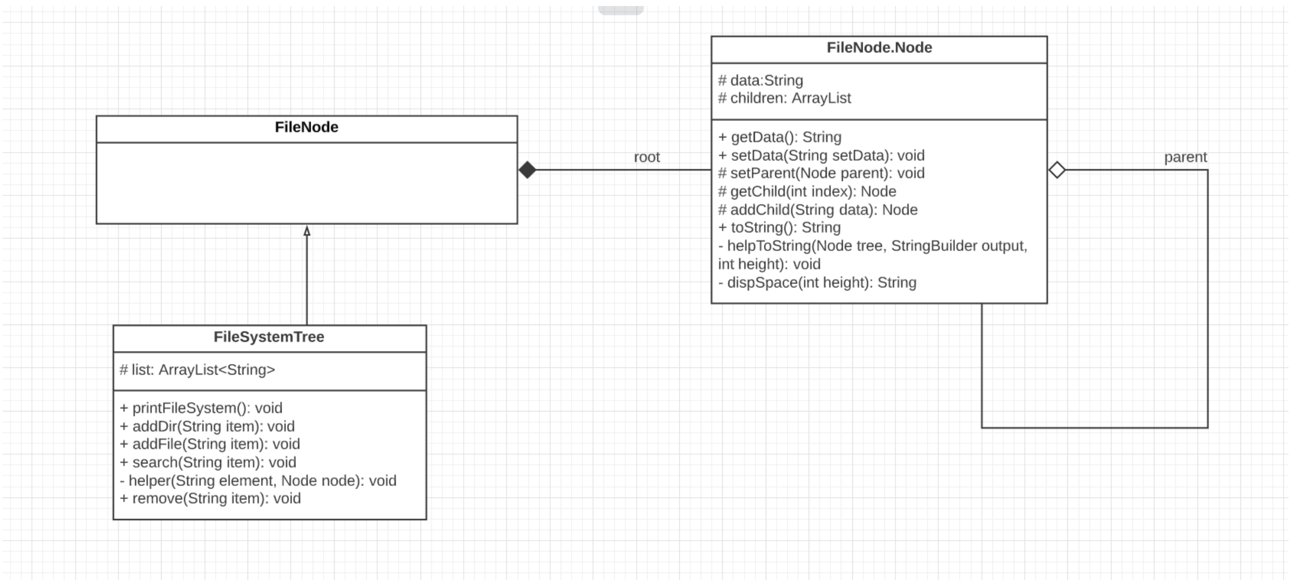
CSE 222/505 - Spring 2020

Homework 5

Mert Can BEŞİRLİ
1801042663

## #Question1#

## 1. CLASS DIAGRAMS


FileNode.Node
# data:String
# children: ArrayList
+ getData(): String
+ setData(String setData): void
# setParent(Node parent): void
# getChild(int index): Node
# addChild(String data): Node
+ toString(): String
- helpToString(Node tree, StringBuilder output, int height): void
- dispSpace(int height): String
FileNode
root
parent
FileSystemTree
# list: ArrayList<String>
+ printFileSystem(): void
+ addDir(String item): void
+ addFile(String item): void
+ search(String item): void
- helper(String element, Node node): void
+ remove(String item): void

## 2. PROBLEM SOLUTION APPROACH

In this problem, we will make the directory and file system on the tree. First, we define "root" in the constructor.
**<u>addDir method-></u>** For adding a directory, the file added must contain the <u>word "directory"</u>, otherwise it cannot be added. This method is adding the directory. The path will go as a parameter in the method. Places with "/" in the path are split and String is added to the array. If the path length is two, that is, the root will have children, the last element of the string array is added to the root children.The parents of the children also take root. If the path length is not two, since the directory we want to add the second element from the bottom of the string array is parent, we look for it in the tree.When we find it, we add it as a child and it becomes the parent of the children. If the file to be added is "file", the function ends.Directory or file can be added to the directory.
**<u>addFile method-></u>** For adding a file, the dierectory added must contain the <u>word "file"</u>, otherwise it cannot be added. This method is adding the file. The path will go as a parameter in the method. Places with "/" in the path are split and String is added to the array. If the path length is two, that is, the root will have children, the last element of the string array is added to the root children.The parents of the children also take root. If the path

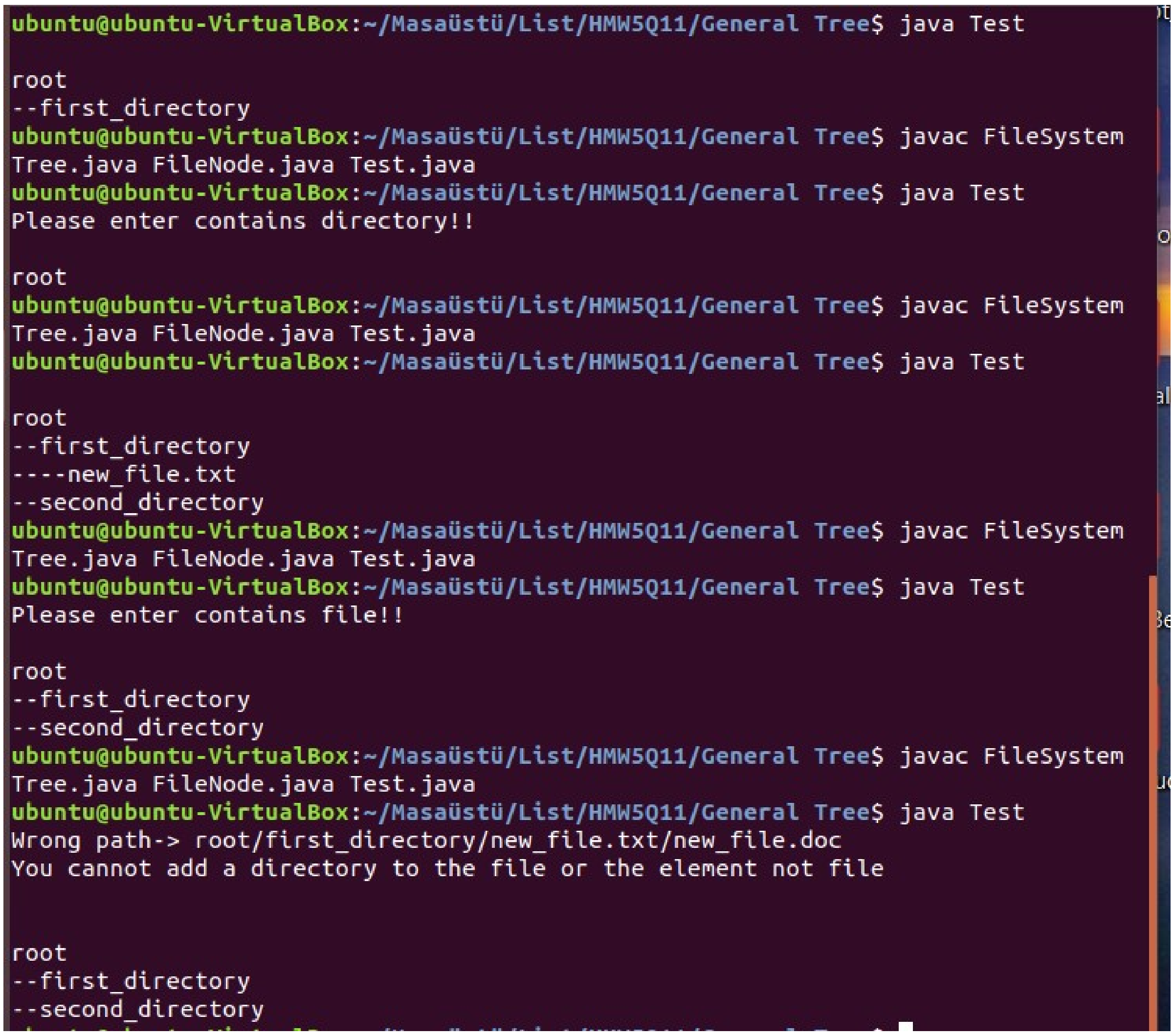
length is not two, since the directory we want to add the second element from the bottom of the string array is parent, we look for it in the tree.When we find it, we add it as a child and it becomes the parent of the children.
**search method->** We search the files or directories containing the name given in the search function. Here we use helper function and we do recursive operation. We go around the tree and if we find the right name, we add it to the list we created and hold the node. Then we add the parents to the list and then clear the list, because it will be needed in the next searches. If we cannot find it, we use iterator to advance the children by making the iterator next and then recursive.
**remove method->** This method includes path, removing element is last of path element. In there, path is going to this method and split "/" then create new array. Then, If length of path equal two or different of two, doing different process. If length is two, removing element searching to in root of children, then find it and remove it. If path is wrong, give error; path is not found.If different of two, searching path elements, respectively, then finding children and remove it.If path is wrong, give user warning.

## 3. TEST CASES

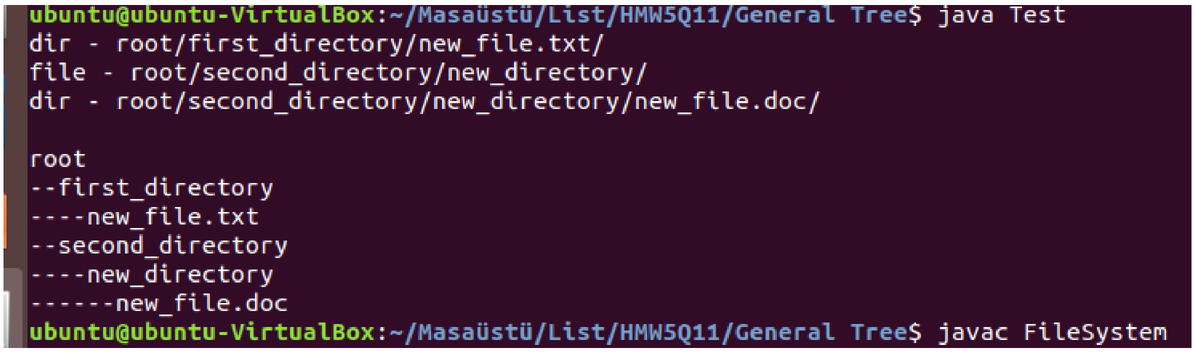
| Test Case ID | Test Scenario | Test Steps | Test Data | Excepted Results | Actual Results | Pass/Fail |
|---|---|---|---|---|---|---|
| T1 | Add Directory valid directory data | Path is root/first_directory | Path is root/first_directory | Adding first_directory successfully | As Excepted | Pass |
| T2 | Add Directory invalid directory data | Path is root/first_director | Path is root/first_director | Adding not happen because adding element shoud be "directory" word | As Excepted | Fail |
| T3 | Add File valid file data | Path is root/first_directory/new_file.txt | Path is root/first_directory/new_file.txt | Adding txt file successfully | As Excepted | Pass |
| T4 | Add File invalid file data | Path is root/first_directory/new.t | Path is root/first_directory/new.t | Adding not happen because | As Excepted | Fail |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | xt | xt | adding element should be “file” word | | |
| T5 | Add File but adding file into file | Path is root/first_directory/new_file.txt/new_file.doc | Path is root/first_directory/new_file.txt/new_file.doc | Adding not happen because not includes file into file | As Excepted | Fail |
| T6 | Searching any word | Word is “new” | Word is “new” | Searching word is finding, print these paths | As Excepted | Pass |
| T7 | Remove any file or directory | Removing path is root/first_directory/new_file.txt | Removing path is root/first_directory/new_file.txt | Removing path succesfully | As Excepted | Pass |
| T8 | Print file system, test last shape | | | Printing successfully | As Excepted | Pass |

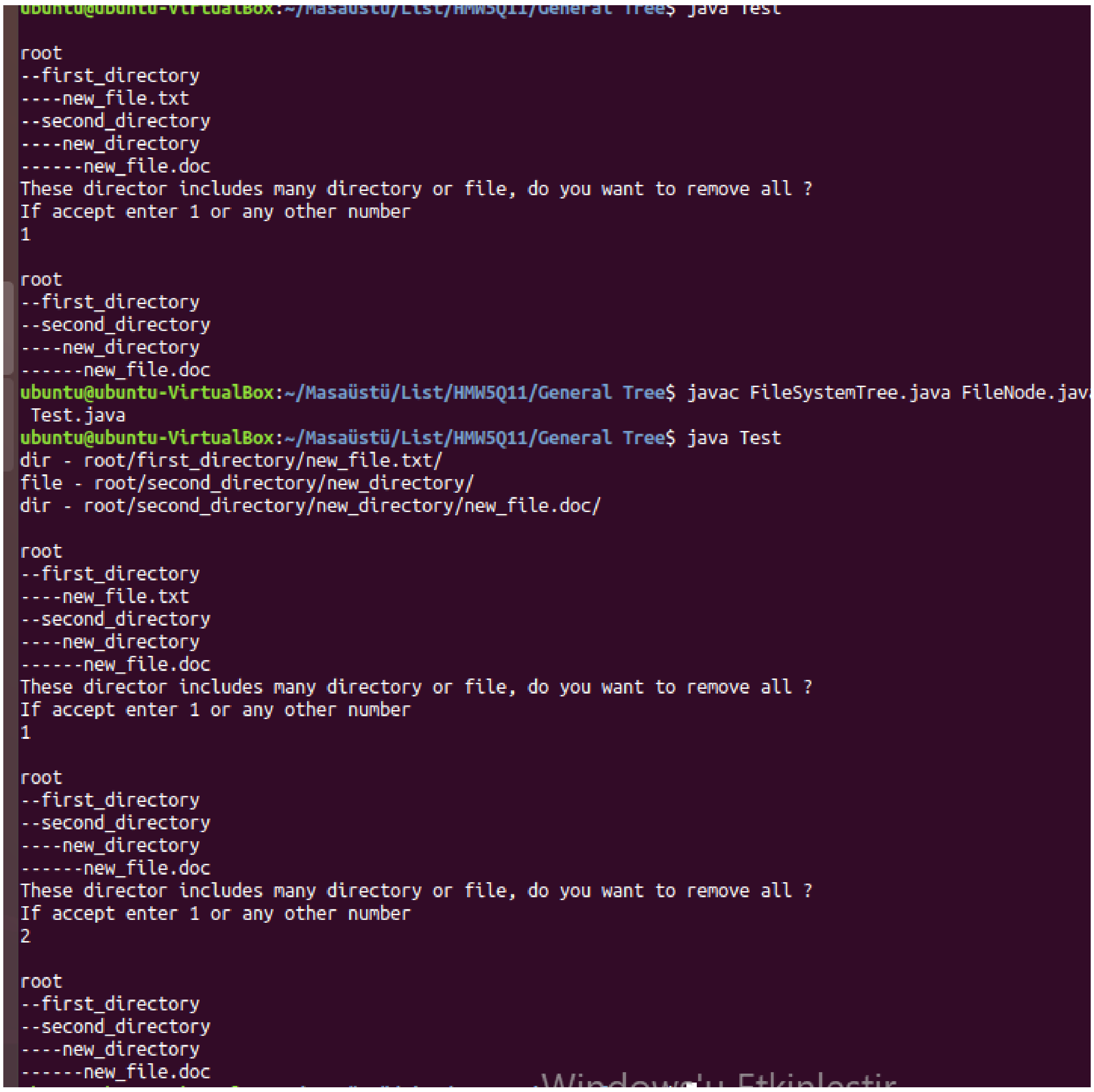
# 4. RUNNING AND RESULTS



```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test

root
--first_directory
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ javac FileSystem
Tree.java FileNode.java Test.java
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test
Please enter contains directory!!

root
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ javac FileSystem
Tree.java FileNode.java Test.java
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test

root
--first_directory
----new_file.txt
--second_directory
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ javac FileSystem
Tree.java FileNode.java Test.java
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test
Please enter contains file!!

root
--first_directory
--second_directory
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ javac FileSystem
Tree.java FileNode.java Test.java
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test
Wrong path-> root/first_directory/new_file.txt/new_file.doc
You cannot add a directory to the file or the element not file


root
--first_directory
--second_directory
```



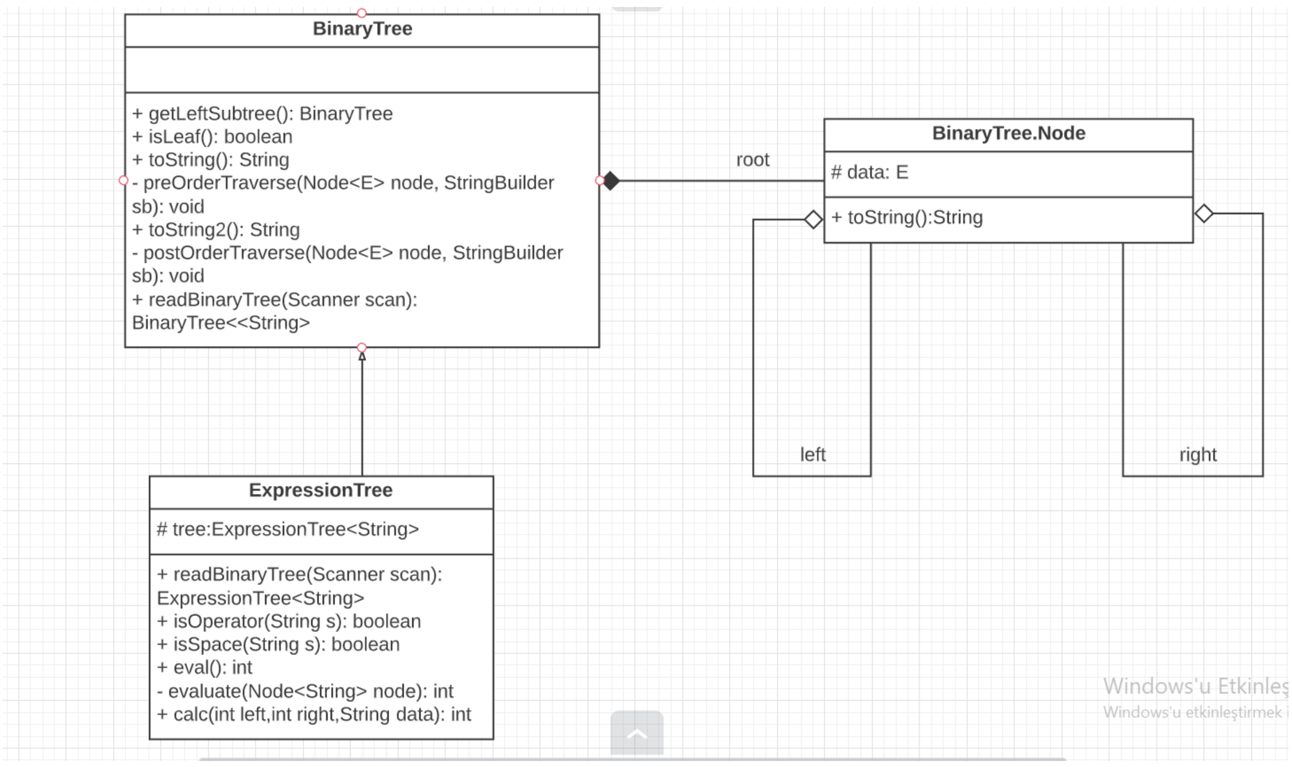
```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test
dir - root/first_directory/new_file.txt/
file - root/second_directory/new_directory/
dir - root/second_directory/new_directory/new_file.doc/

root
--first_directory
----new_file.txt
--second_directory
----new_directory
------new_file.doc
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ javac FileSystem
```



```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test

root
--first_directory
----new_file.txt
--second_directory
----new_directory
------new_file.doc
These director includes many directory or file, do you want to remove all ?
If accept enter 1 or any other number
1

root
--first_directory
--second_directory
----new_directory
------new_file.doc
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ javac FileSystemTree.java FileNode.jav
 Test.java
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q11/General Tree$ java Test
dir - root/first_directory/new_file.txt/
file - root/second_directory/new_directory/
dir - root/second_directory/new_directory/new_file.doc/

root
--first_directory
----new_file.txt
--second_directory
----new_directory
------new_file.doc
These director includes many directory or file, do you want to remove all ?
If accept enter 1 or any other number
1

root
--first_directory
--second_directory
----new_directory
------new_file.doc
These director includes many directory or file, do you want to remove all ?
If accept enter 1 or any other number
2

root
--first_directory
--second_directory
----new_directory
------new_file.doc
```

## #Question2#

## 1. CLASS DIAGRAMS


BinaryTree
+ getLeftSubtree(): BinaryTree
+ isLeaf(): boolean
+ toString(): String
- preOrderTraverse(Node<E> node, StringBuilder sb): void
+ toString2(): String
- postOrderTraverse(Node<E> node, StringBuilder sb): void
+ readBinaryTree(Scanner scan): BinaryTree<<String>
root
BinaryTree.Node
# data: E
+ toString():String
left
right
ExpressionTree
# tree:ExpressionTree<String>
+ readBinaryTree(Scanner scan): ExpressionTree<String>
+ isOperator(String s): boolean
+ isSpace(String s): boolean
+ eval(): int
- evaluate(Node<String> node): int
+ calc(int left,int right,String data): int

## 2. PROBLEM SOLUTION APPROACH

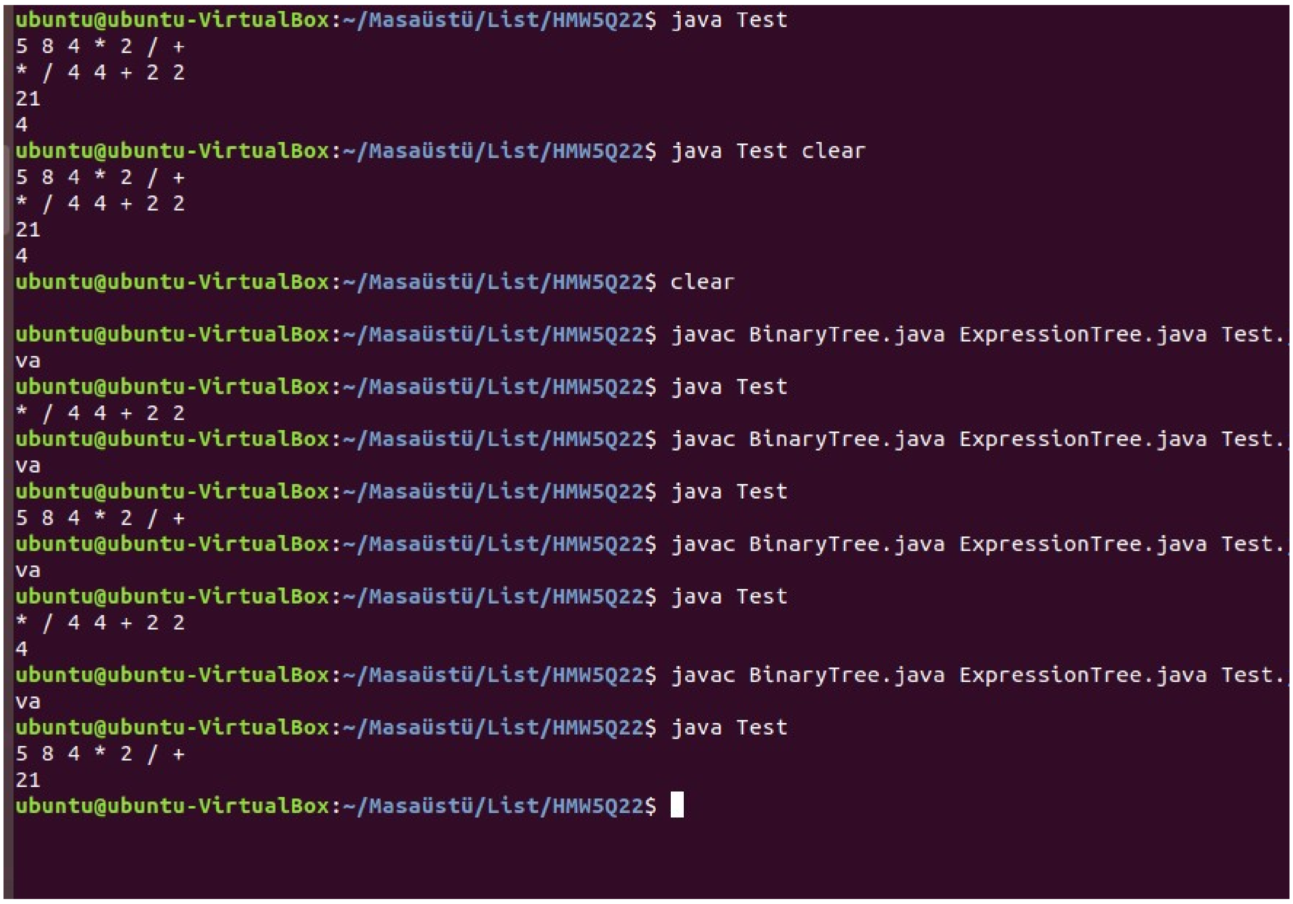
In this problem, create tree of postfix or prefix expression data. First of all, we place the elements one by one in the tree. To distinguish between prefix and postfix, we look at whether the first element is operator or operand. But there is a problem, the tree only works in single digit numbers.After the tree is formed, we print it as preordertraverse for prefix and postordertraverse for postfix.Then, using the eval method, we perform the operation using operators and operands in the tree and print the result.

## 3. TEST CASES

| Test Case ID | Test Scenario | Test Steps | Test Data | Excepted Results | Actual Results | Pass/Fail |
|---|---|---|---|---|---|---|
| T1 | Check prefix expression and then preorder traversel printing | Prefix expression have test class | * + 2 2 / 4 4 | This expression display with preorder traversel | As Excepted | Pass |
| T2 | Check postfix expression and then preorder | Postfix expression have test class | 5 8 4 * 2 / + | This expression display with postorder traversel | As Excepted | Pass |

|  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|
|  | traversel printing |  |  |  |  |  |
| T3 | Check prefix expression and then evaulation result printing | Prefix expression have test class | * + 2 2 / 4 4 | This expression evaluation is printing | As Excepted | Pass |
| T4 | Check postflix expression and then evaulation result printing | Postfix expression have test class | 5 8 4 * 2 / + | This expression evaluation is printing | As Excepted | Pass |

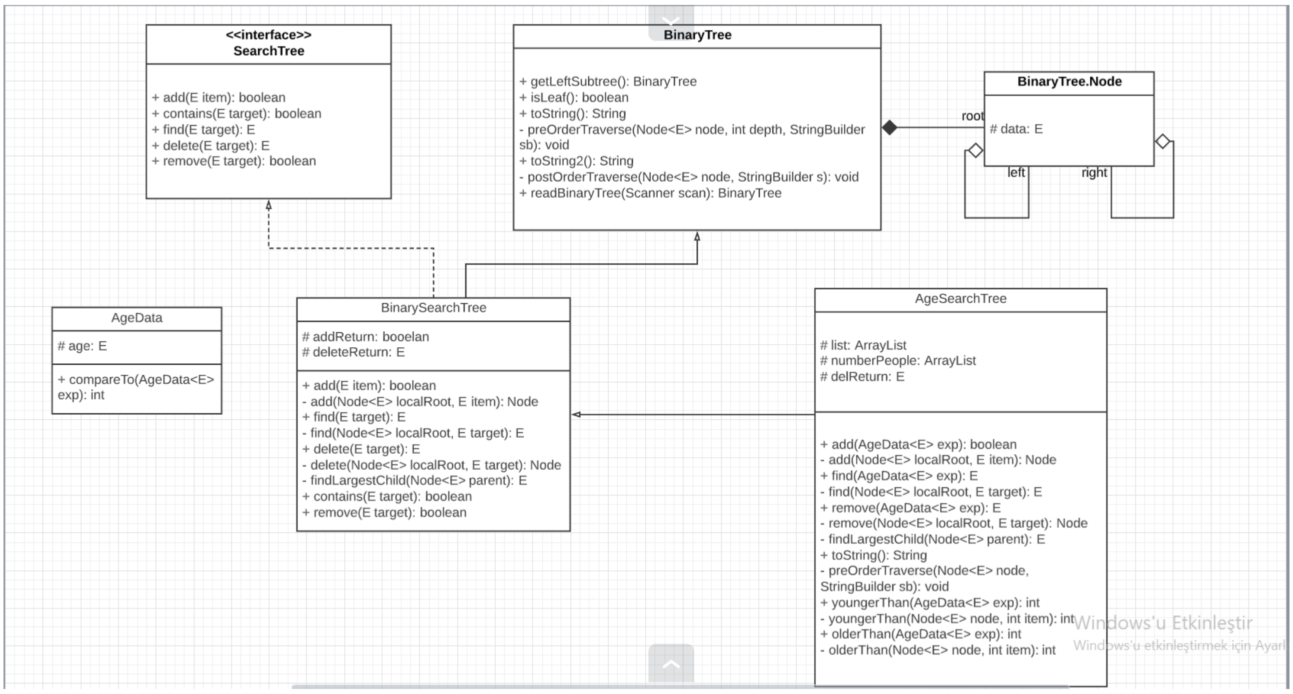
# 4. RUNNING AND RESULTS



```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ java Test
5 8 4 * 2 / +
* / 4 4 + 2 2
21
4
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ java Test clear
5 8 4 * 2 / +
* / 4 4 + 2 2
21
4
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ clear

ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ javac BinaryTree.java ExpressionTree.java Test.
va
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ java Test
* / 4 4 + 2 2
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ javac BinaryTree.java ExpressionTree.java Test.
va
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ java Test
5 8 4 * 2 / +
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ javac BinaryTree.java ExpressionTree.java Test.
va
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ java Test
* / 4 4 + 2 2
4
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ javac BinaryTree.java ExpressionTree.java Test.
va
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$ java Test
5 8 4 * 2 / +
21
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMW5Q22$
```

**#Question 3#**

## 1. CLASS DIAGRAMS



<<interface>> SearchTree
+ add(E item): boolean
+ contains(E target): boolean
+ find(E target): E
+ delete(E target): E
+ remove(E target): boolean
BinaryTree
+ getLeftSubtree(): BinaryTree
+ isLeaf(): boolean
+ toString(): String
- preOrderTraverse(Node<E> node, int depth, StringBuilder sb): void
+ toString2(): String
- postOrderTraverse(Node<E> node, StringBuilder s): void
+ readBinaryTree(Scanner scan): BinaryTree
BinaryTree.Node
# data: E
root
left
right
AgeData
# age: E
+ compareTo(AgeData<E> exp): int
BinarySearchTree
# addReturn: booelan
# deleteReturn: E
+ add(E item): boolean
- add(Node<E> localRoot, E item): Node
+ find(E target): E
- find(Node<E> localRoot, E target): E
+ delete(E target): E
- delete(Node<E> localRoot, E target): Node
- findLargestChild(Node<E> parent): E
+ contains(E target): boolean
+ remove(E target): boolean
AgeSearchTree
# list: ArrayList
# numberPeople: ArrayList
# delReturn: E
+ add(AgeData<E> exp): boolean
- add(Node<E> localRoot, E item): Node
+ find(AgeData<E> exp): E
- find(Node<E> localRoot, E target): E
+ remove(AgeData<E> exp): E
- remove(Node<E> localRoot, E target): Node
- findLargestChild(Node<E> parent): E
+ toString(): String
- preOrderTraverse(Node<E> node, StringBuilder sb): void
+ youngerThan(AgeData<E> exp): int
- youngerThan(Node<E> node, int item): int
+ olderThan(AgeData<E> exp): int
- olderThan(Node<E> node, int item): int

## 2. PROBLEM SOLUTION APPROACH

In this problem, record the number of people in each age population with binary search tree.
**add method->** In the addition process, we first check whether that element has been added before. If added, we do not add again and number of people increase one and the method finishes the process. If the element is not added, we add it to the tree and the number of people is one
**remove method->** First, we check if the element to be deleted exists in the tree, or it returns without doing anything. If there is more than one person at that age, we reduce it. If it is a person, we delete the element from the tree.
**find method->** We are looking for the element that needs to be found in the tree and, if found, return it but pdf as specified find(new AgeData(10)).toString() I was not able to do it.Because the toString method prints tree elements and I was not able to use it that way.
**youngerThan method->** In this method, we find the number of numbers smaller than the age given. Every time we find the numbers smaller than

the element in the tree, it increases according to the number of people.But we cannot send it as ageTree.youngerThan (10) as stated in pdf, because AgeSearchTree<AgeData> implementing. I made a submission this way-> ageTree.youngerThan(new AgeData(10)).
**olderThan method->** In this method, we find the number of numbers higher than the age given. Every time we find the numbers higher than the element in the tree, it increases according to the number of people.But we cannot send it as ageTree.olderThan (10) as stated in pdf, because AgeSearchTree<AgeData> implementing. I made a submission this way-> ageTree.olderThan(new AgeData(10)).

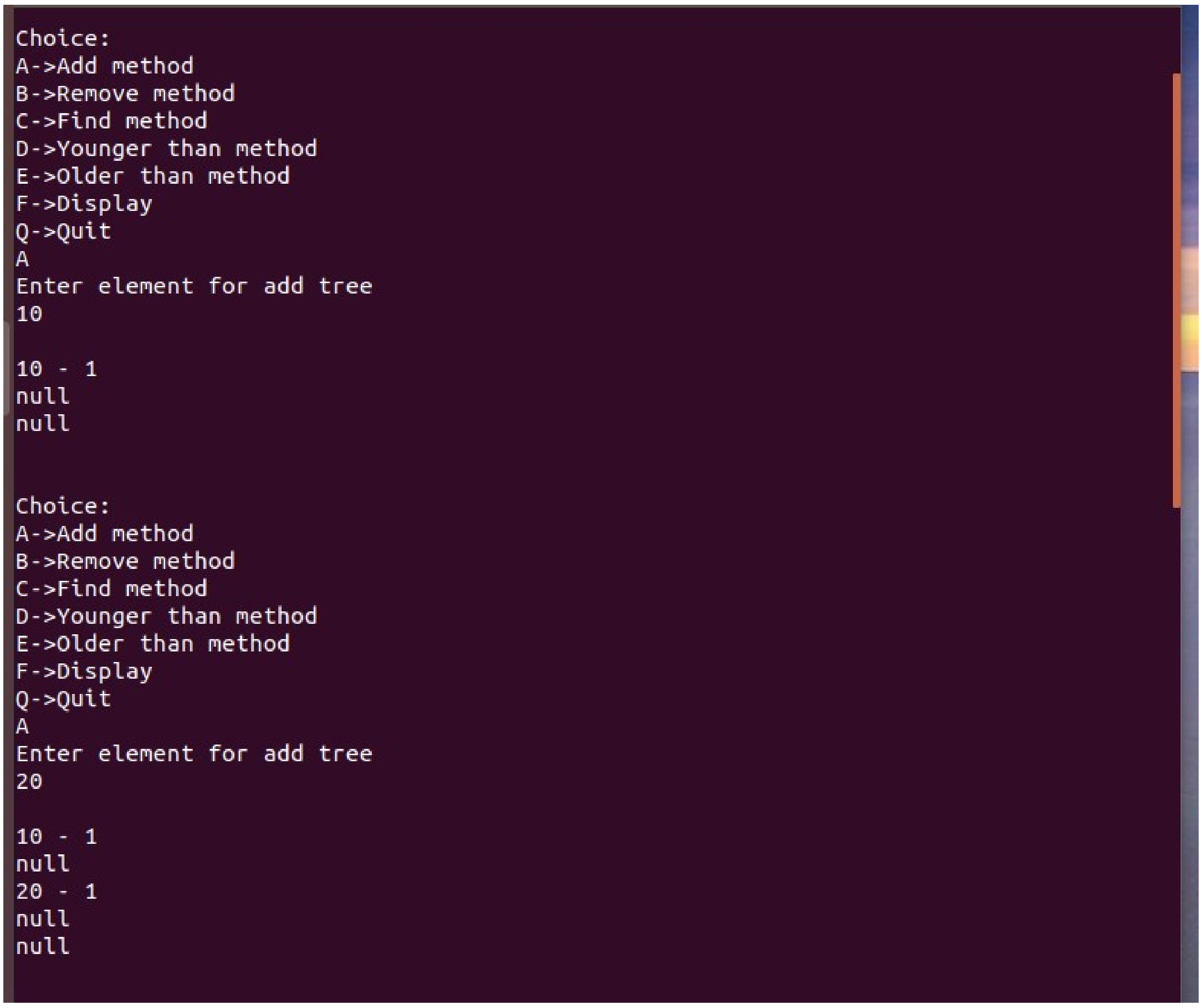
## 3. TEST CASES

| Test Case ID | Test Scenario | Test Steps | Test Data | Excepted Results | Actual Results | Pass/Fail |
|---|---|---|---|---|---|---|
| T1 | Check add age with add method | Choice: A->Add method B->Remove method C->Find method D->Younger than method E->Older than method F->Display Q->Quit | Enter element for add tree: 10 | Adding element of tree | As expected | Pass |
| T2 | Check remove age of tree Tree is: 10 - 2 5 - 1 null null null 20 - 1 15 - 1 null null null | Choice: A->Add method B->Remove method C->Find method D->Younger than method E->Older than method F->Display Q->Quit | Enter element for remove tree 20 | Removing element of tree | As Expected | Pass |
| T3 | Check remove number of people higher than one age | Choice: A->Add method B->Remove method C->Find | Enter element for remove tree 10 | Decreasing one, number of people to 10 | As Expected | Pass |

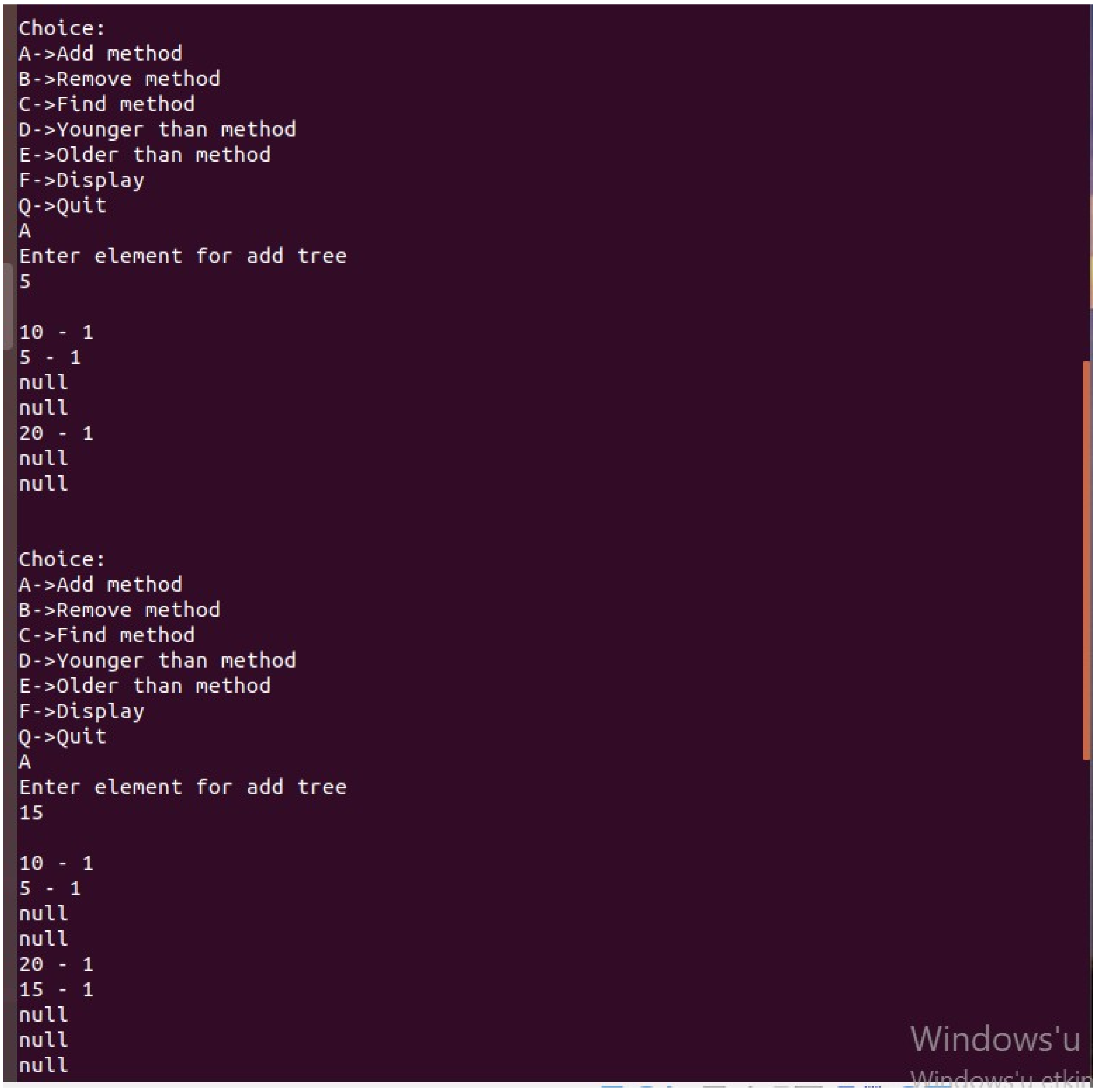
| | | | | | | |
|---|---|---|---|---|---|---|
| | | method D->Younger than method E->Older than method F->Display Q->Quit | | | | |
| T4 | Check find age of tree | Choice: A->Add method B->Remove method C->Find method D->Younger than method E->Older than method F->Display Q->Quit | Enter element for find element to tree 20 | Find element 20 int tree | As Expected | Pass |
| T5 | Check youngerThan any age number of people | Choice: A->Add method B->Remove method C->Find method D->Younger than method E->Older than method F->Display Q->Quit | Enter element for younger than element in tree 13 | Younger than number elements 2 | As Expected | Pass |
| T6 | Check olderThan any age number of people | Choice: A->Add method B->Remove method C->Find method D->Younger than method E->Older than method F->Display Q->Quit | Enter element for older than element in tree 13 | Older than number elements 2 | As Expected | Pass |
| T7 | Check add age with add method invalid data | Choice: A->Add method B->Remove | Enter element for add tree ddd | Give error handling | As Expected | Fail |

|  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|
|  |  | method C->Find method D->Younger than method E->Older than method F->Display Q->Quit |  |  |  |  |

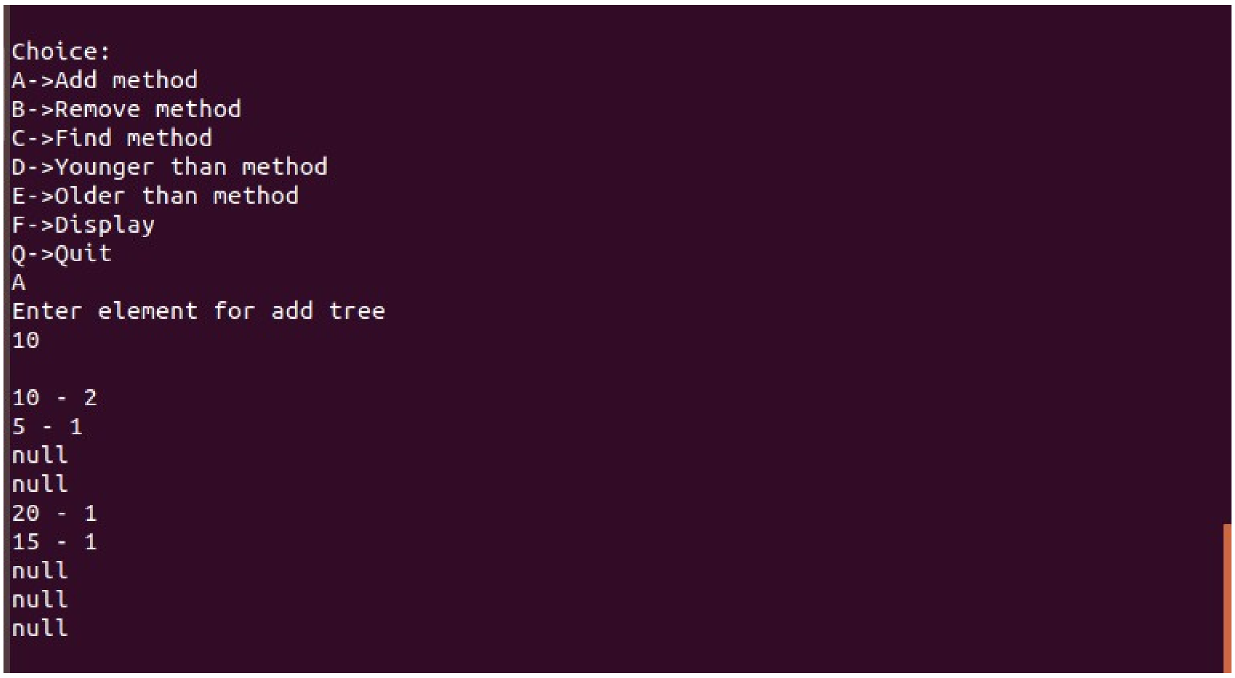
## 4. RUNNING AND RESULTS



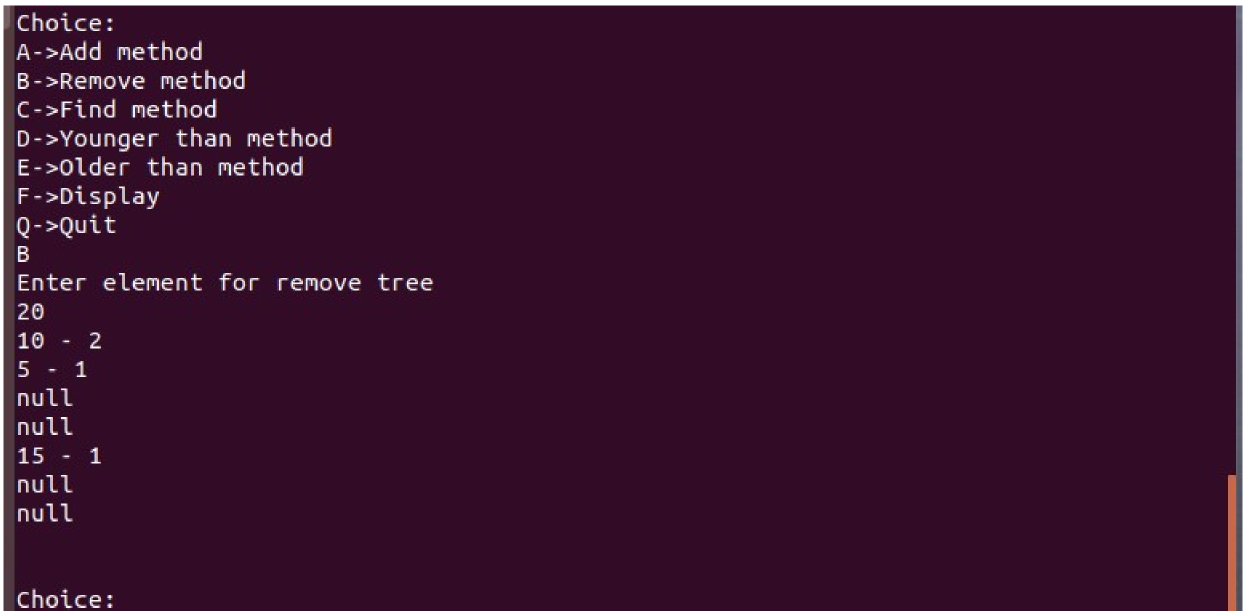
```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
A
Enter element for add tree
10

10 - 1
null
null


Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
A
Enter element for add tree
20

10 - 1
null
20 - 1
null
null
```



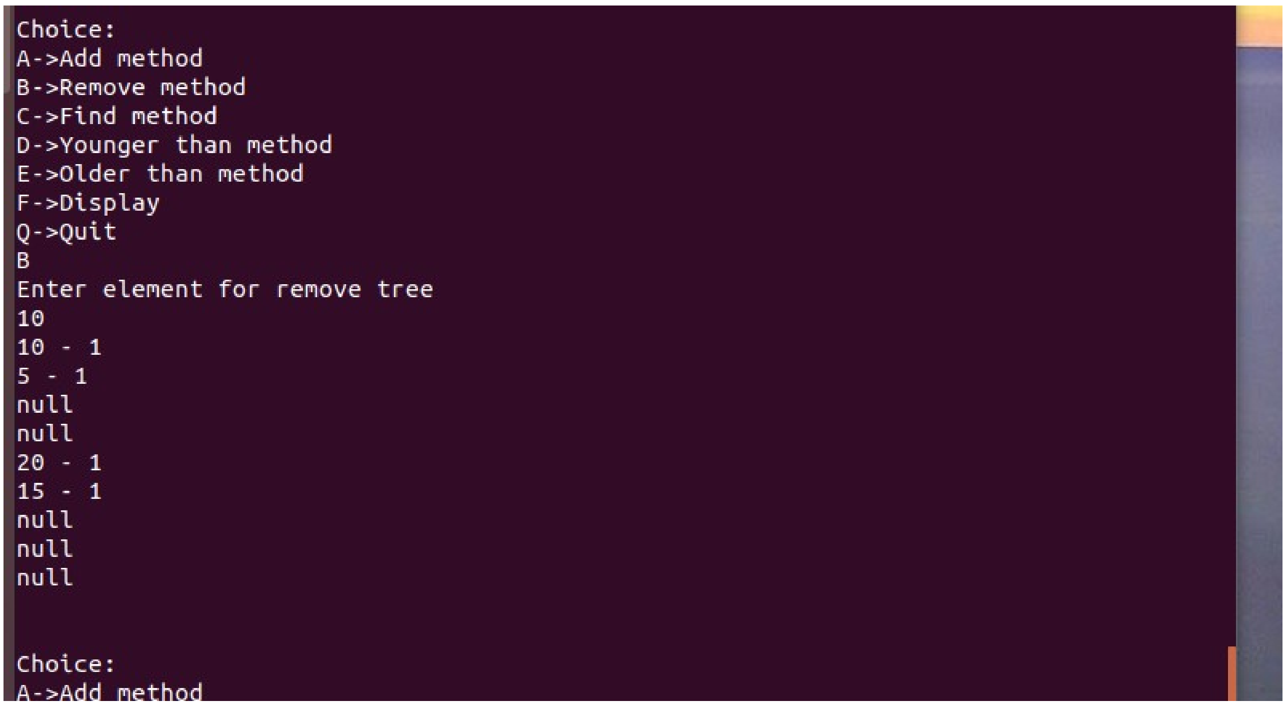
```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
A
Enter element for add tree
5

10 - 1
5 - 1
null
null
20 - 1
null
null


Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
A
Enter element for add tree
15

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



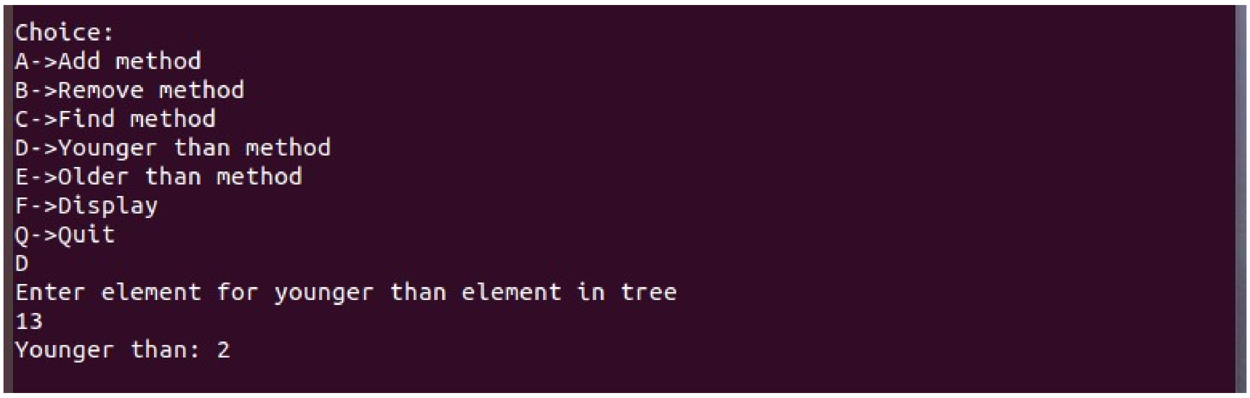
```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
A
Enter element for add tree
10

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



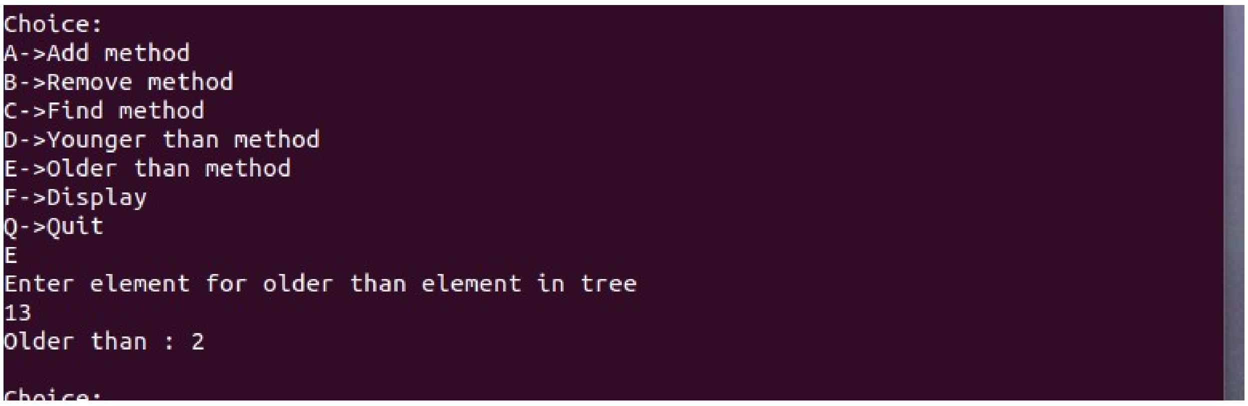
```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
B
Enter element for remove tree
20
10 - 2
5 - 1
null
null
15 - 1
null
null


Choice:
```



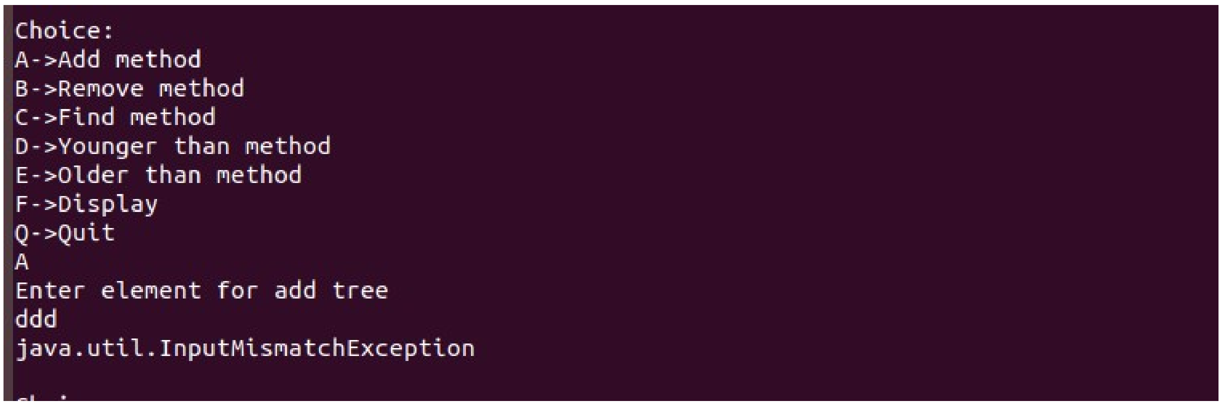
```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
B
Enter element for remove tree
10
10 - 1
5 - 1
null
null
20 - 1
15 - 1
null
null
null


Choice:
A->Add method
```



```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
D
Enter element for younger than element in tree
13
Younger than: 2
```



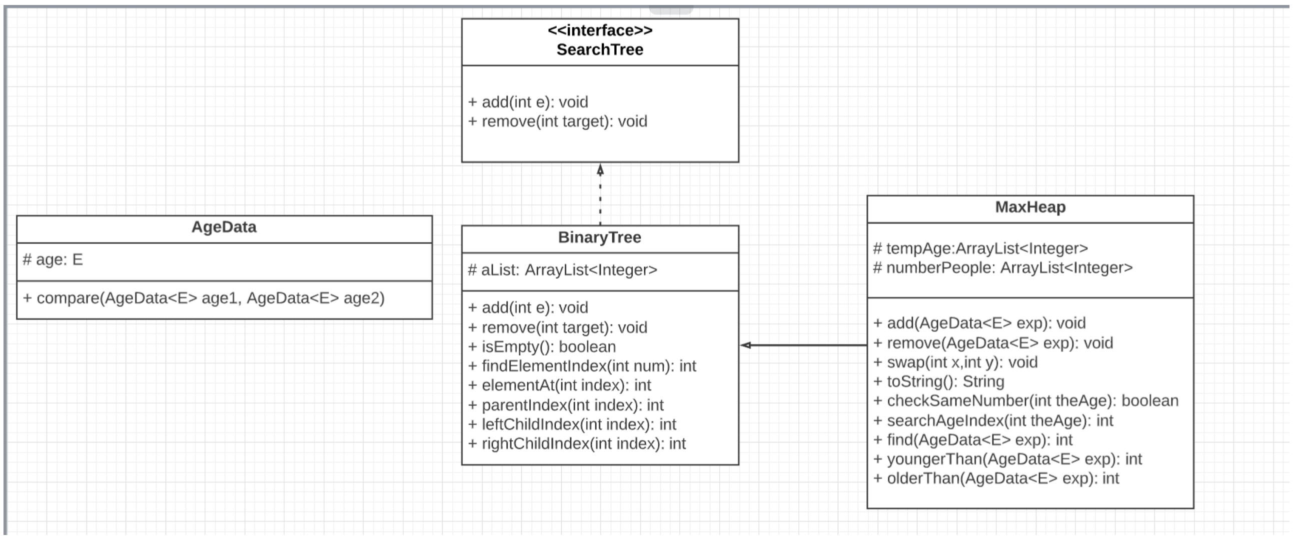
```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
E
Enter element for older than element in tree
13
Older than : 2
```



```
Choice:
A->Add method
B->Remove method
C->Find method
D->Younger than method
E->Older than method
F->Display
Q->Quit
A
Enter element for add tree
ddd
java.util.InputMismatchException
```

## <u>**#Question 4#**</u>

## **1. CLASS DIAGRAMS**


<<interface>>
SearchTree
+ add(int e): void
+ remove(int target): void
AgeData
# age: E
+ compare(AgeData<E> age1, AgeData<E> age2)
BinaryTree
# aList: ArrayList<Integer>
+ add(int e): void
+ remove(int target): void
+ isEmpty(): boolean
+ findElementIndex(int num): int
+ elementAt(int index): int
+ parentIndex(int index): int
+ leftChildIndex(int index): int
+ rightChildIndex(int index): int
MaxHeap
# tempAge:ArrayList<Integer>
# numberPeople: ArrayList<Integer>
+ add(AgeData<E> exp): void
+ remove(AgeData<E> exp): void
+ swap(int x,int y): void
+ toString(): String
+ checkSameNumber(int theAge): boolean
+ searchAgeIndex(int theAge): int
+ find(AgeData<E> exp): int
+ youngerThan(AgeData<E> exp): int
+ olderThan(AgeData<E> exp): int

## 2. PROBLEM SOLUTION APPROACH

In this problem, We tried to do the 3rd question with max heap. Max heap ArrayList implementation.
**add method->** In the addition process, we first check whether that element has been added before.If it is added, we are not adding again. We increase the number by 1 according to numberPeople index and method ends its transaction without any further processing. If that element is not added, we add that element to the heap arraylist. Then we add it to tempAge arraylist(tempAge necessary for number of people), add one to the numberpeople arraylist. Because now there is one of this element. Then we arrange the element according to the max heap rules.
**remove method->** First, we check if the element to be deleted exists in the heap. If there is no element, we throw exception, if there is, we find its index in the heap. If there is more than one person at this age, we are reducing the number of people at that age and the method ends. If it is a person, the element to be deleted first is replaced by the last element in the heap and the last element is deleted. After that, the heap is arranged according to the max heap order.This happens in two ways, firstly, If the element to be deleted is root, different operations are done, if not, different operations are performed.

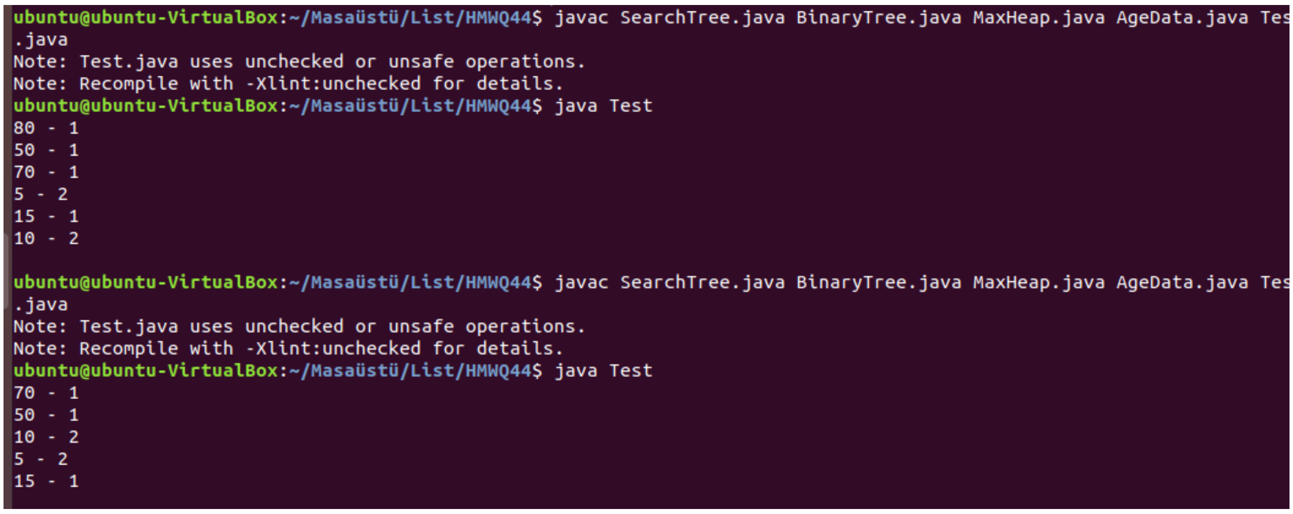
**find method->** In this method, we are looking for the age to be found in heap. If we find it, we're returning that number but pdf as specified find(new AgeData(10)).toString() I was not able to do it.Because the toString method prints heap elements and I was not able to use it that way.
**youngerThan method->** We are asked to find the number of ages younger than the age given in this method. When we find smaller numbers than the given element in Heap, the number increases according to numberOfPeople.But we cannot send it as heap.youngerThan (10) as stated in pdf, because MaxHeap<AgeData> implementing. I made a submission this way--> heap.youngerThan(new AgeData(10)).
**olderThan method->** We are asked to find the number of ages older than the age given in this method. When we find higher numbers than the given element in Heap, the number increases according to numberOfPeople.But we cannot send it as heap.olderThan (10) as stated in pdf, because MaxHeap<AgeData> implementing. I made a submission this way--> heap.olderThan(new AgeData(10)).

## 3. TEST CASES

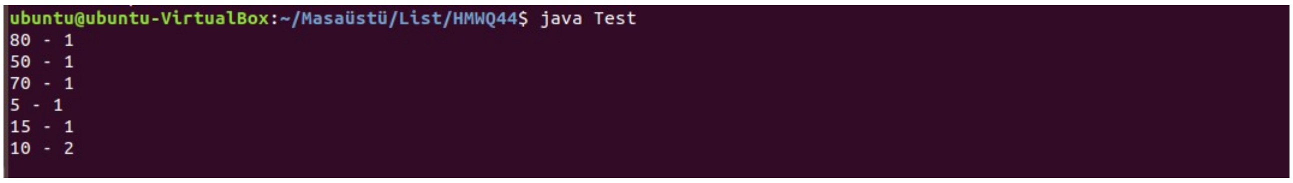
| Test Case ID | Test Scenario | Test Steps | Test Data | Excepted Results | Actual Results | Pass/Fail |
|---|---|---|---|---|---|---|
| T1 | Check heap add ages then display | The ages to be added are determined | Heap adding data 10 5 70 10 50 5 15 80 | Printing heap 80 - 1 50 - 1 70 - 1 5 - 2 15 - 1 10 - 2 | As Excepted | Pass |
| T2 | Check heap remove ages of number of people is 1 | The ages to be removed is determined | Removing data is 80 | Printing heap 70 - 1 50 - 1 10 - 2 5 - 2 15 – 1 | As Excepted | Pass |
| T3 | Check heap remove ages of number of people higher than 1 | The ages to be removed is determined | Removing data is 5 | Removed is successfull 80 - 1 50 - 1 70 - 1 5 - 1 15 - 1 10 - 2 | As Excepted | Pass |

| | | | | | | |
|---|---|---|---|---|---|---|
| T4 | Check find age in heap then find it | The ages to be finding is determined | Finding data is 70 | Find it successfully | As Excepted | Pass |
| T5 | Check find age in heap then not find it | The ages to be finding is determined | Finding data is 13 | Element not found, give error throw exception | As Excepted | Fail |
| T6 | Find younger than people any elements then display result | The ages to be younger is determined | Finding number less than 20. | Finding successfully, result is 5 | As Excepted | Pass |
| T7 | Find older than people any elements then display result | The ages to be older is determined | Finding number higher than 25 | Finding succesfully, result is 3 | As Excepted | Pass |

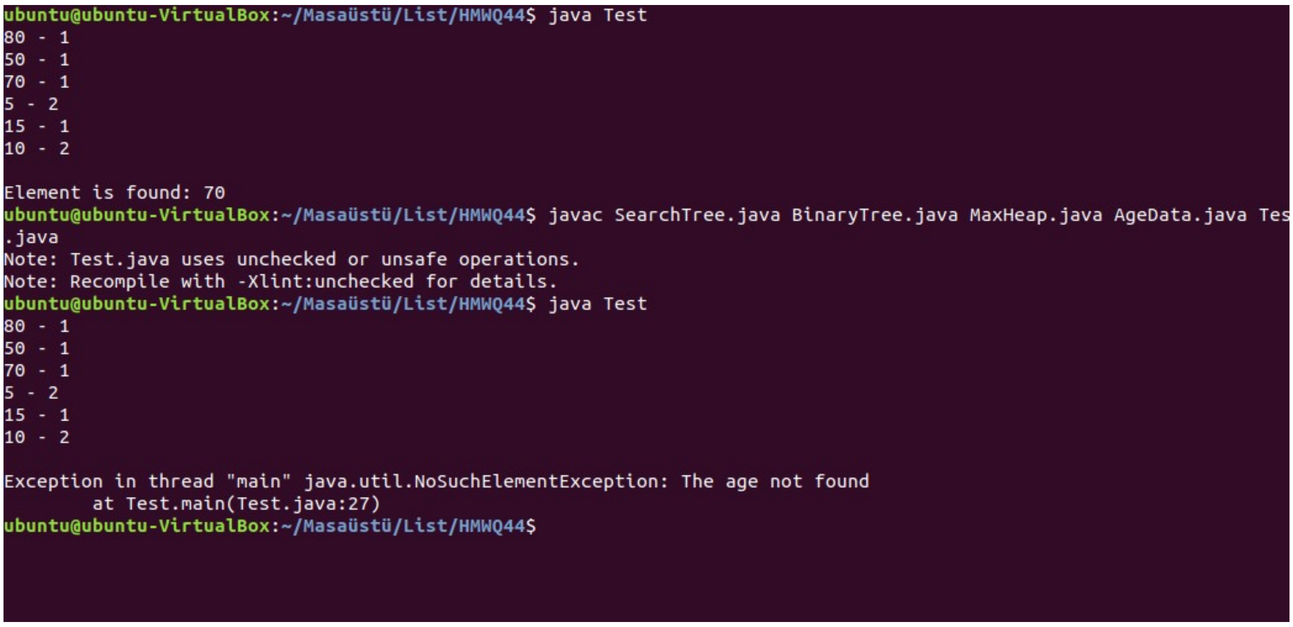
# 4. RUNNING AND RESULTS



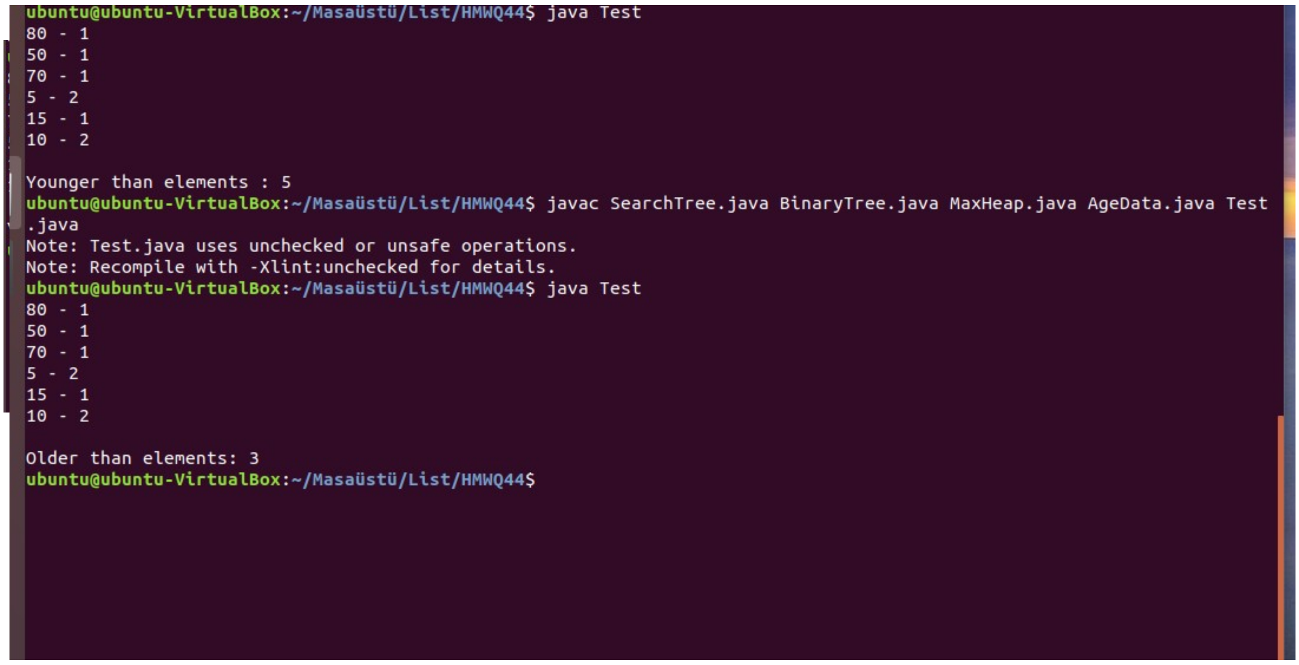
```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ javac SearchTree.java BinaryTree.java MaxHeap.java AgeData.java Tes
.java
Note: Test.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ java Test
80 - 1
50 - 1
70 - 1
5 - 2
15 - 1
10 - 2

ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ javac SearchTree.java BinaryTree.java MaxHeap.java AgeData.java Tes
.java
Note: Test.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ java Test
70 - 1
50 - 1
10 - 2
5 - 2
15 - 1
```



```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ java Test
80 - 1
50 - 1
70 - 1
5 - 1
15 - 1
10 - 2
```



```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ java Test
80 - 1
50 - 1
70 - 1
5 - 2
15 - 1
10 - 2

Element is found: 70
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ javac SearchTree.java BinaryTree.java MaxHeap.java AgeData.java Tes
.java
Note: Test.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ java Test
80 - 1
50 - 1
70 - 1
5 - 2
15 - 1
10 - 2

Exception in thread "main" java.util.NoSuchElementException: The age not found
        at Test.main(Test.java:27)
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$
```



```
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ java Test
80 - 1
50 - 1
70 - 1
5 - 2
15 - 1
10 - 2

Younger than elements : 5
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ javac SearchTree.java BinaryTree.java MaxHeap.java AgeData.java Test
.java
Note: Test.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$ java Test
80 - 1
50 - 1
70 - 1
5 - 2
15 - 1
10 - 2

Older than elements: 3
ubuntu@ubuntu-VirtualBox:~/Masaüstü/List/HMWQ44$
```