GIT Department of Computer Engineering

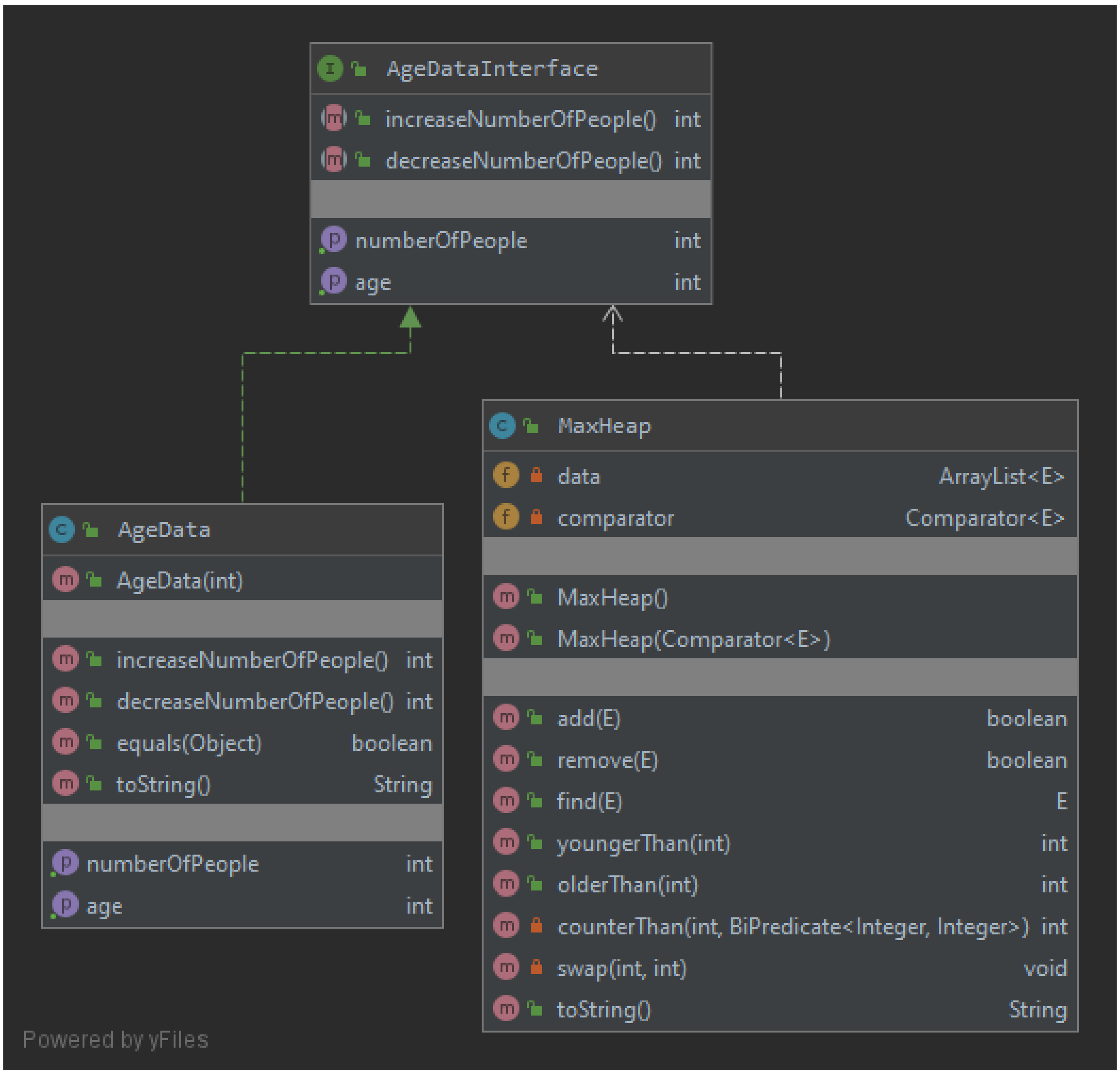
CSE 222/505 – Spring 2020

Homework #05 Part 4 Report

Abdullah ÇELİK

171044002

Class Diagram



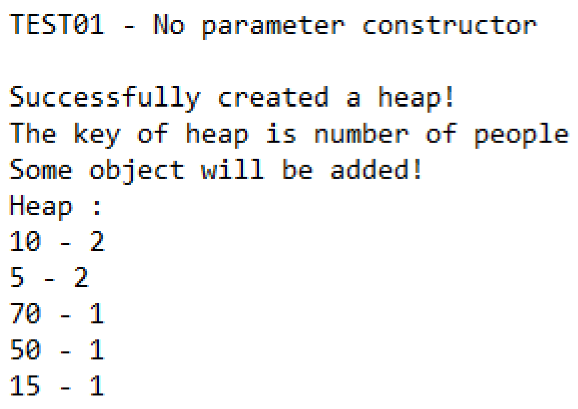
Problem Solution Approach

Firstly, since MaxHeap class is generic class, it is a problem to use methods of generic type object. There are two solutions. First solution is to cast this generic type object. This solution does not disaccord object oriented programming idea. Second solution is force to this generic type object to implement an interface i wrote. In this way, I can implement MaxHeap class using that methods offered by this interface. Secondly, generic type object is not need to be comparable. Fort this reason, I can create a Comparator object and implement it with methods that AgeDataInterface offers. In this way, I implement MaxHeapin methods that require comparison with the method of this comparator object. The remaining process was to correctly implement the required methods.

Test Cases

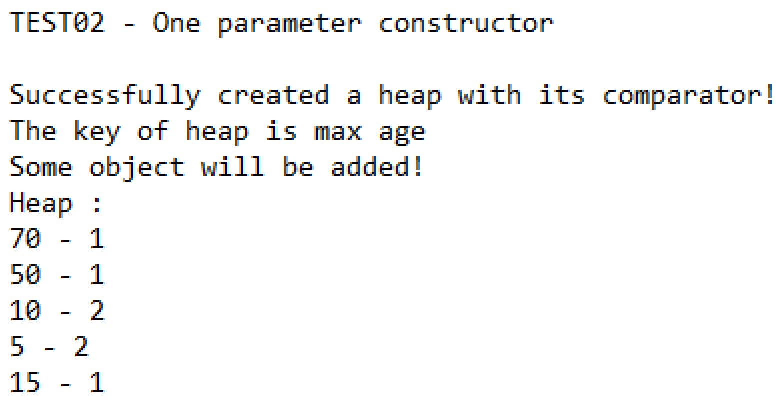
Test ID	Scenerio	Test Data	Expected Results	Actual Results	Pass/Fail
TEST01	No parameter constructor	MaxHeap	Successfully created	As expected	Pass
TEST02	One parameter constructor will be called with a comparator object	MaxHeap	Successfully created	As expected	Pass
TEST03	boolean add(E e) method called when heap is empty and has some elements	Heap Size : 0 e : AgeData(10) Heap Size : 1 e : AgeData(5) e : AgeData(15) e : AgeData(5) e : null	Successfully added except for null and set heap	As expected	Pass
TEST04	boolean remove(E e) method called when heap has some elements	Heap Size : 3 e : AgeData(5) e : AgeData(5) e : AgeData(10) e : AgeData(15) e : null	Successfully removed except for null and set heap	As expected	Pass
TEST05	E find(E e) method called when heap has some elements	Heap Size : 5 e : AgeData(10) e : AgeData(20) e : AgeData(null)	Successfully returned object if it is exist	As expected	Pass
TEST06	int youngerThan(int age) method called when heap has some elements	Heap Size : 5 age : 50	Successfully returned correct value	As expected	Pass
TEST07	int olderThan(int age) method called when heap has some elements	Heap Size : 5 age : 10	Successfully returned correct value	As expected	Pass

Running and Results



```
TEST01 - No parameter constructor

Successfully created a heap!
The key of heap is number of people
Some object will be added!
Heap :
10 - 2
5 - 2
70 - 1
50 - 1
15 - 1
```



```
TEST02 - One parameter constructor

Successfully created a heap with its comparator!
The key of heap is max age
Some object will be added!
Heap :
70 - 1
50 - 1
10 - 2
5 - 2
15 - 1
```

```
TEST03 - boolean add(E e) method
The key of heap is number of people

When heap is empty, method will be called as
        heap.add(new AgeData(10))
Before adding
Heap :

Add 10 : true
After adding
Heap :
10 - 1

When heap has some elements, method will be called respectively as
        heap.add(new AgeData(5)), heap.add(new AgeData(15)),
        heap.add(new AgeData(5)), heap.add(null)
Before adding
Heap :
10 - 1

Add 5 : true
Add 15 : true
Add 5 : true
Add null : false
After adding
Heap :
5 - 2
10 - 1
15 - 1
```

```
TEST04 - boolean remove(E e)
The key of heap is number of people

When heap has some elements, method will be called respectively as
        heap.remove(new AgeData(5)), heap.remove(new AgeData(5)),
        heap.remove(new AgeData(10)), heap.remove(new AgeData(15)),
        heap.remove(new AgeData(null))
Before removing
Heap :
5 - 2
10 - 2
15 - 1

Remove 5 : true
After removing age 5
Heap :
10 - 2
5 - 1
15 - 1

Remove 5 : true
After removing age 5
Heap :
10 - 2
15 - 1

Remove 10 : true
After removing age 10
Heap :
10 - 1
15 - 1

Remove 15 : true
After removing age 15
Heap :
10 - 1

Remove null : false
After removing null
Heap :
10 - 1
```

```
TEST05 - E find(E e)

When list has some elements, method will be called respectively
        heap.find(new AgeData(10)), heap.find(new AgeData(20)),
        heap.find(null)
Heap :
10 - 2
5 - 2
70 - 1
50 - 1
15 - 1

Is Age 10 exist? 10 - 2
Is Age 20 exist? null
Is null exist? null

TEST06 - int youngerThan(int age)

When list has some elements, method will be called
        heap.youngerThan(50)
Heap :
10 - 2
5 - 2
70 - 1
50 - 1
15 - 1

There are 5 people younger than 50!

TEST07 - int olderThan(int age)

When list has some elements, method will be called
        heap.olderThan(10)
Heap :
10 - 2
5 - 2
70 - 1
50 - 1
15 - 1

There are 3 people younger than 10!
```