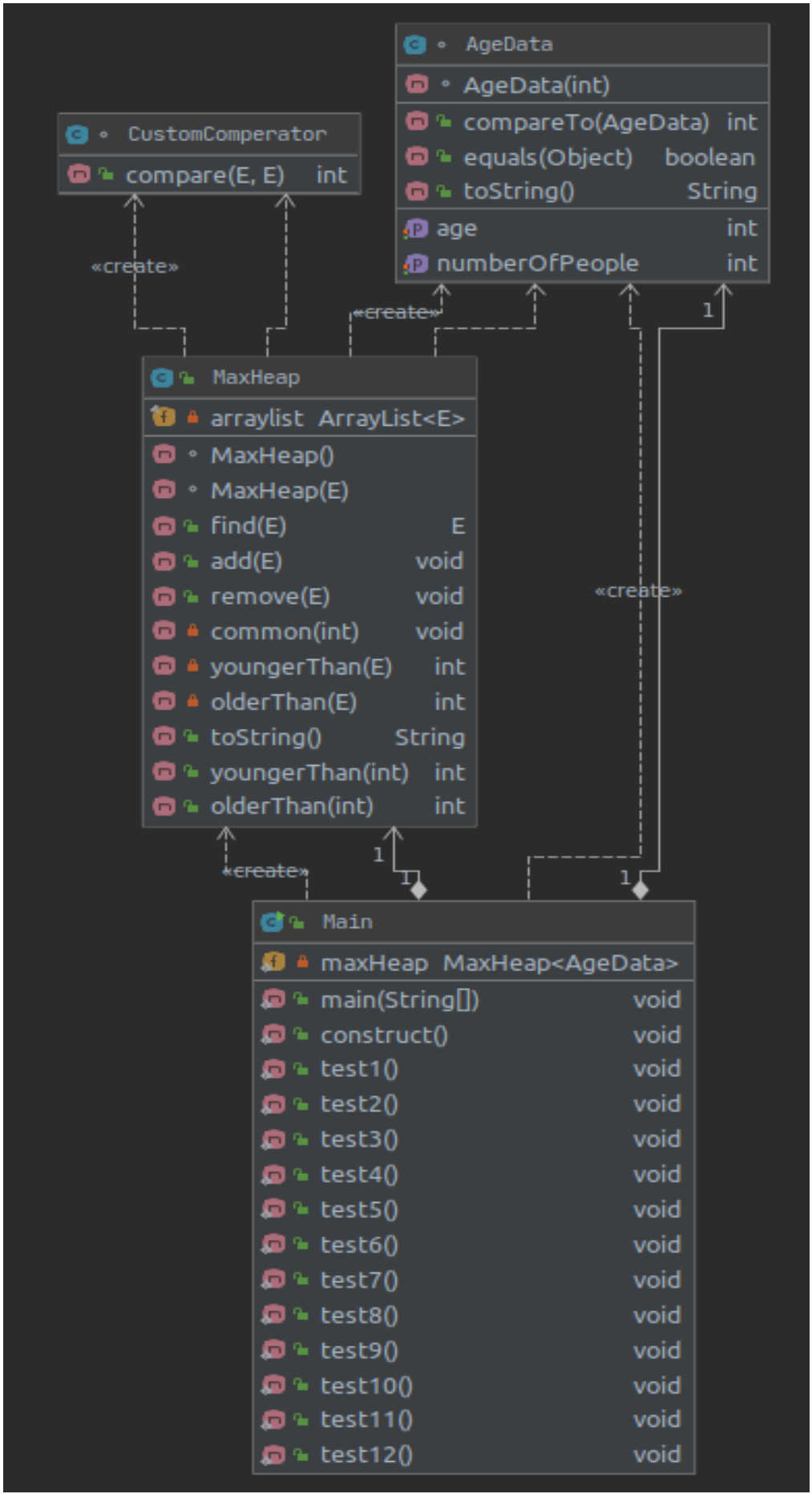
GIT Department of Computer Engineering
CSE 222/505 - Spring 2020
Homework 5 Question 4 Report

Melihcan Çilek
1801042092

1-)Class Diagram



2-)Problem Solution Approach

If I have to summarize problem for questions 1, problem for question 2 is that implement your heap by using ArrayList as described in your book. Implement MaxHeap class to handle the ArrayList heap operations. Use the AgeData class you implemented in Q3 to hold age and the number of people at that age data. The key of heap will be “number of people” this time. So, the age which the highest number of people is at, will be at the root. For this implementation, because of the key is number of people, compare to implementation at question 3, I had to change compareTo method for being able to fit to implementation so that compareTo method can either increase or decrease number of people so that nodes can be switched. According to method that is called, function of compareTo method will be changed. If method that is called yougerThan or olderThan, compareTo method compares ages but if method that is called is add, remove or find, it behaves like question 3 but considering number of people. For heap implementation, there is two type which are maximum heap and minimum heap but for this question we wanted to implement maximum heap. Max-Heap is a complete binary tree in which the value in each internal node is greater than or equal to the values in the children of that node. There are similarities between add and remove method so I created a method that behave similarity.

3-)Running Command And Results

```
//Add method / adding AgeData that is not in the heap
public static void test1(){
    construct();
    System.out.println(maxHeap);
    maxHeap.add(new AgeData(765));
    System.out.println(maxHeap);
}
```

```
//Add method / adding AgeData that is in the heap and numberOfPeople will be increased
//so we move it to top of the heap
public static void test2(){
    construct();
    System.out.println(maxHeap);
    maxHeap.add(new AgeData(10));
    System.out.println(maxHeap);
}
```

```
//Add method / adding AgeData that is in the heap and numberOfPeople will be increased
//so we move it to top of the heap
public static void test3(){
    construct();
    maxHeap.add(new AgeData(10));
    System.out.println(maxHeap);
    maxHeap.add(new AgeData(3));
    maxHeap.add(new AgeData(3));
    System.out.println(maxHeap);
}
```

```
//Find method / finding ageData that can be found
public static void test4(){
    construct();
    System.out.println(maxHeap.find(new AgeData(3)));
}
```

```
//Find method / finding ageData that cannot be found
public static void test5(){
    construct();
    System.out.println(maxHeap.find(new AgeData(45)));
}
```

```java
//Remove method / remove that has 1 number of people
public static void test6(){
    construct();
    System.out.println(maxHeap);
    maxHeap.remove(new AgeData(3));
    System.out.println(maxHeap);
}
```

```java
//Remove method / remove that has more than one number of people
public static void test7(){
    construct();
    maxHeap.add(new AgeData(10));
    System.out.println(maxHeap);
    maxHeap.remove(new AgeData(10));
    System.out.println(maxHeap);
}
```

```java
//Remove method / remove that has more than one number of people
public static void test8(){
    construct();
    maxHeap.add(new AgeData(10));
    maxHeap.add(new AgeData(3));
    System.out.println(maxHeap);
    maxHeap.remove(new AgeData(10));
    System.out.println(maxHeap);
}
```

```java
//Remove method / remove that has more than one number of people
public static void test9(){
    construct();
    maxHeap.add(new AgeData(10));
    maxHeap.add(new AgeData(3));
    maxHeap.add(new AgeData(10));
    maxHeap.add(new AgeData(3));
    System.out.println(maxHeap);
    maxHeap.remove(new AgeData(10));
    System.out.println(maxHeap);
}
```

```java
//olderThan method
public static void test10(){
    construct();
    System.out.println(maxHeap.olderThan( i: 32));
}
```

```java
//youngerThan method
public static void test11(){
    construct();
    System.out.println(maxHeap.youngerThan( i: 32));
}
```

```java
//youngerThan method
public static void test12(){
    construct();
    System.out.println(maxHeap.toString());
}
```