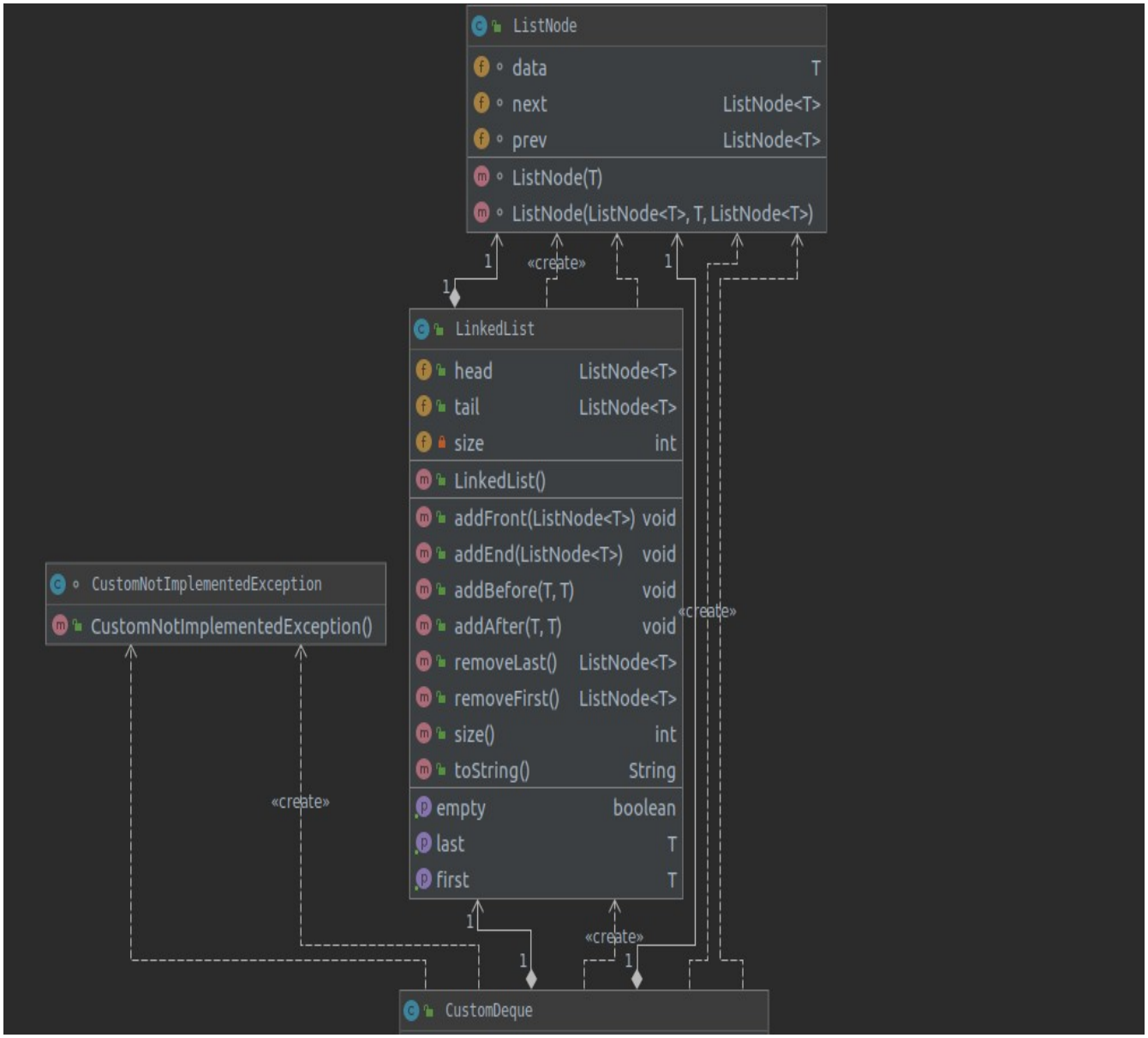
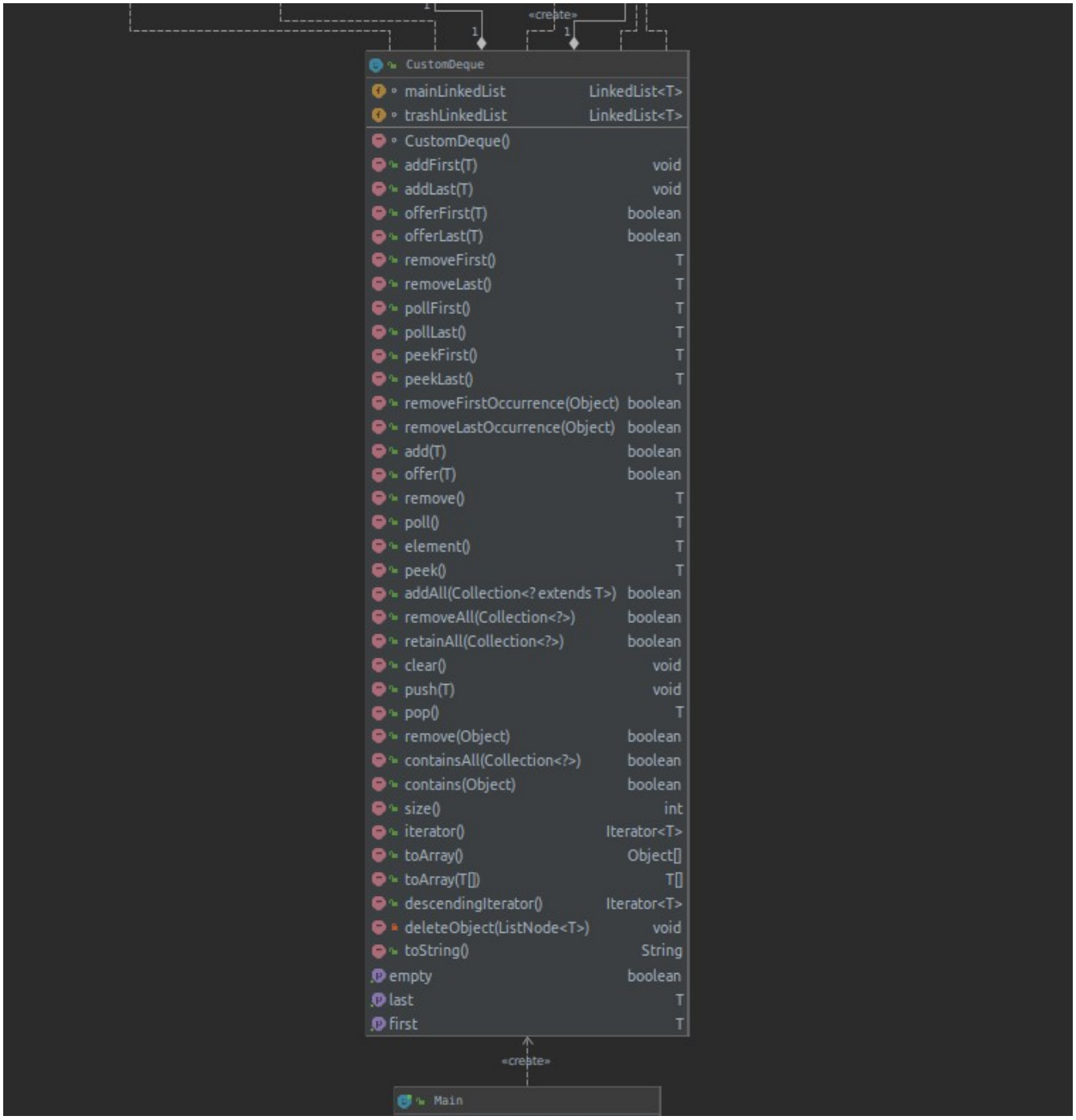
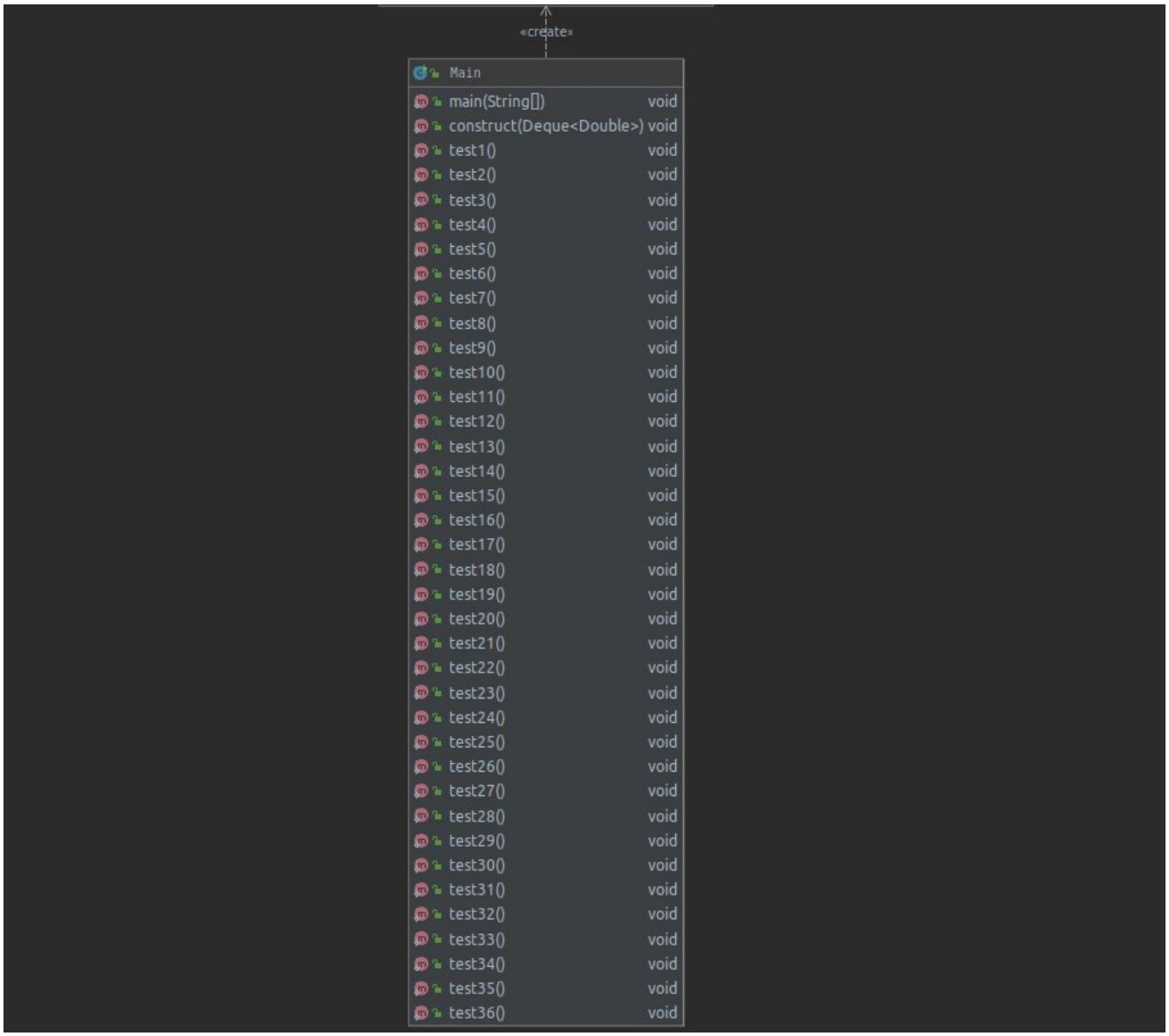
# GIT Department of Computer Engineering
# CSE 222/505 - Spring 2020
# Homework 4 Question 2 Report

**Melihcan Çilek**
**1801042092**

# Class Diagrams




«create»
1
1
CustomDeque
mainLinkedList LinkedList<T>
trashLinkedList LinkedList<T>
CustomDeque()
addFirst(T) void
addLast(T) void
offerFirst(T) boolean
offerLast(T) boolean
removeFirst() T
removeLast() T
pollFirst() T
pollLast() T
peekFirst() T
peekLast() T
removeFirstOccurrence(Object) boolean
removeLastOccurrence(Object) boolean
add(T) boolean
offer(T) boolean
remove() T
poll() T
element() T
peek() T
addAll(Collection<? extends T>) boolean
removeAll(Collection<?>) boolean
retainAll(Collection<?>) boolean
clear() void
push(T) void
pop() T
remove(Object) boolean
containsAll(Collection<?>) boolean
contains(Object) boolean
size() int
iterator() Iterator<T>
toArray() Object[]
toArray(T[]) T[]
descendingIterator() Iterator<T>
deleteObject(ListNode<T>) void
toString() String
empty boolean
last T
first T
«create»
Main


«create»
Main
main(String[]) void
construct(Deque<Double>) void
test1() void
test2() void
test3() void
test4() void
test5() void
test6() void
test7() void
test8() void
test9() void
test10() void
test11() void
test12() void
test13() void
test14() void
test15() void
test16() void
test17() void
test18() void
test19() void
test20() void
test21() void
test22() void
test23() void
test24() void
test25() void
test26() void
test27() void
test28() void
test29() void
test30() void
test31() void
test32() void
test33() void
test34() void
test35() void
test36() void

# Problem Solution Approachs

For second question of 4th homework, we have to İmplement our own Deque class with generics. Inside our deque class, we have two linkedlist field that holds values. These Linkedlists are implemented by custom and Nodes inside linkedlists are also my own implementation. Good thing about this implementation is we keep another linkedlist which we can say trash linkedlist. Aim of this linkedlist inside Deque class is that if we want to delete a node from our linkedlist inside deque class, we move that node to trash linkedlist. After that, if we want to add a node beginning or end of deque, we don't have to create node with new, we can move node end of the trash linkedlist to our linkedlist so we save our time for node that has to be created. Inside custom deque class, we implement an iterator for deque. For solving that problem, First, I look java documentation and understand how each method works. Second, I started implementing deque class. While implementing this class, first,I was hiding ListNode class inside Linkedlist class as static but after implemented some methods and add trash linkedlist, I obtained that I have to use my ListNode class Public static and I changed that implementation style. After that I realize another thing that I can move all ListNode class inside of CustomDeque class that I implement Deque class static and it worked as package-private. So that I solved this problem in this way.

# Test Cases:

```
public static void test1(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.addFirst( e: 192.332);
    System.out.println(deque);
}
```

```
public static void test2(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.addLast( e: 192.332);
    System.out.println(deque);
}
```

```
public static void test3(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.offerFirst( e: 192.332);
    System.out.println(deque);
}
```

```
public static void test4(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.offerLast( e: 192.332);
    System.out.println(deque);
}
```

```
public static void test5(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.removeFirst();
    System.out.println(deque);
}
```

```
public static void test6(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.removeLast();
    System.out.println(deque);
}
```

```
public static void test7(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.pollFirst();
    System.out.println(deque);
}
```

```
public static void test8(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.pollLast();
    System.out.println(deque);
}
```

```
public static void test9(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("First value is " + deque.getFirst());
    System.out.println(deque);
}
```

```
public static void test10(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("Last value is " + deque.getLast());
    System.out.println(deque);
}
```

```
public static void test11(){
    Deque<Double> deque = new CustomDeque<>();
    System.out.println(deque);
    System.out.println("First value is " + deque.peekFirst());
}
```

```java
public static void test12(){
    Deque<Double> deque = new CustomDeque<>();
    System.out.println(deque);
    System.out.println("Last value is " + deque.peekLast());
}
public static void test13(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("First value is " + deque.peekFirst());
    System.out.println(deque);
}
public static void test14(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    deque.addLast( e: 45.45);
    System.out.println(deque);
    deque.removeFirstOccurrence( o: 45.45);
    System.out.println(deque);
}
public static void test15(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    deque.addLast( e: 45.45);
    System.out.println(deque);
    deque.removeLastOccurrence( o: 45.45);
    System.out.println(deque);
}
public static void test16(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.add(192.332);
    System.out.println(deque);
}
public static void test17(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.offer( e: 192.332);
    System.out.println(deque);
}
public static void test18(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.remove();
    System.out.println(deque);
}
public static void test19(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.poll();
    System.out.println(deque);
}
public static void test20(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("Result of element method is " + deque.element());
}
public static void test21(){
    Deque<Double> deque = new CustomDeque<>();
    System.out.println(deque);
    System.out.println("Result of peek method of empty method is " + deque.peek());
}
public static void test22(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("Result of peek method of empty method is " + deque.peek());
}
```

```java
public static void test23(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.clear();
    System.out.println(deque);
}
```

```java
public static void test24(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.push( e: 192.332);
    System.out.println(deque);
}
```

```java
public static void test25(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    deque.pop();
    System.out.println(deque);
}
```

```java
public static void test26(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    deque.addLast( e: 45.45);
    System.out.println(deque);
    System.out.println("Result of remove method is " + deque.remove(Double.valueOf(4.26)));
    System.out.println(deque);
}
```

```java
public static void test27(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    deque.addLast( e: 45.45);
    System.out.println(deque);
    System.out.println("Result of remove method is " + deque.remove(Double.valueOf(45.45)));
    System.out.println(deque);
}
```

```java
public static void test28(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("Does Deque contains 4.26?\n" + deque.contains(Double.valueOf(4.26)));
}
```

```java
public static void test29(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("Does Deque contains 3.12?\n" + deque.contains(Double.valueOf(3.12)));
}
```

```java
public static void test30(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("Size of Deque is " + deque.size());
}
```

```java
public static void test31(){
    Deque<Double> deque = new CustomDeque<>();
    System.out.println(deque);
    System.out.println("Is Deque empty?\n" + deque.isEmpty());
}
public static void test32(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    System.out.println("Is Deque empty?\n" + deque.isEmpty());
}
public static void test33(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    Iterator<Double> iterator = deque.iterator();
    while (iterator.hasNext()){
        System.out.println(iterator.next());
    }
}
public static void test34(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    Object [] array = deque.toArray();
    for (Object o : array) {
        System.out.println(o);
    }
}
public static void test35(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    System.out.println(deque);
    Object [] array = deque.toArray();
    deque.toArray(array);
}
public static void test36(){
    Deque<Double> deque = new CustomDeque<>();
    construct(deque);
    Iterator<Double> iterator = deque.descendingIterator();
    while (iterator.hasNext()){
        System.out.println(iterator.next());
    }
}
```

# Running Commands And Results:

## Test 1:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[192.332,37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

## Test 2:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6,192.332]
```

## Test 3:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[192.332,37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

## Test 4:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6,192.332]
```

## Test 5:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

## Test 6:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45]
```

## Test 7:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

## Test 8:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45]
```

Test 9:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
First value is 37.12
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 10:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Last value is 5.6
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 11:

```
Deque=[]
First value is null
```

Test 12:

```
Deque=[]
Last value is null
```

Test 13:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
First value is 37.12
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 14:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6,45.45]
Deque=[37.12,5.46,3.12,42.45,51.6,3.12,4.45,5.6,45.45]
```

Test 15:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6,45.45]
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 16:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[192.332,37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 17:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[192.332,37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 18:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 19:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 20:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Result of element method is 37.12
```

Test 21:

```
Deque=[]
Result of peek method of empty method is null
```

Test 22:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Result of peek method of empty method is 37.12
```

Test 23:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[]
```

Test 24:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[192.332,37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 25:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Deque=[45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 26:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6,45.45]
Result of remove method is false
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6,45.45]
```

Test 27:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6,45.45]
Result of remove method is true
Deque=[37.12,5.46,3.12,42.45,51.6,3.12,4.45,5.6,45.45]
```

Test 28:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Does Deque contains 4.26?
false
```

Test 29:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Does Deque contains 3.12?
true
```

Test 30:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Size of Deque is 9
```

Test 31:

```
Deque=[]
Is Deque empty?
true
```

Test 32:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
Is Deque empty?
false
```

Test 33:

```
37.12
45.45
5.46
3.12
42.45
51.6
3.12
4.45
5.6
```

Test 34:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
```

Test 35:

```
Deque=[37.12,45.45,5.46,3.12,42.45,51.6,3.12,4.45,5.6]
CustomNotImplementedException: NotImplementedException
    at CustomDeque.toArray(CustomDeque.java:314)
    at Main.test35(Main.java:295)
    at Main.main(Main.java:42)
```

Test 36:

```
5.6
4.45
3.12
51.6
42.45
3.12
5.46
45.45
37.12
```