

Gebze Technical University

Computer Engineering

CSE 222
2017 Spring

HOMEWORK 3 REPORT

Gözde DOĞAN
131044019

Contents

1. Question1	4
1. System Requirements.....	4
StackInterface:.....	4
StackA:.....	4
StackB:	4
StackC:	4
StackD:.....	4
2. Class Diagrams.....	5
StackA:.....	5
StackB:	5
StackC:	6
StackD:.....	6
3. Problem Solution Approach	7
StackInterface:.....	7
StackA:	7
StackB:	7
StackC:	7
StackD:.....	7
4. Test Cases	7
5. Running Command and Results.....	8
Input file:	8
Output file:	8
2. Question2	9
1. System Requirements.....	9
2. Class Diagrams.....	9
3. Problem Solution Approach	10
4. Test Cases	10
5. Running Command and Results.....	11
Input file:	11
Output File:.....	11
3. Question3	12
1. System Requirements.....	12
PriorityQueueA:.....	12
PriorityQueueB:	12
2. Class Diagrams.....	12

PriorityQueueA:.....	12
PriorityQueueB:.....	13
3. Problem Solution Approach	13
PriorityQueueA:.....	13
PriorityQueueB:.....	13
4. Test Cases	14
5. Running Command and Results.....	14
Input File:.....	14
Output File:.....	14
deleteMin test results:	15

1. Question1

1. System Requirements

StackInterface:

- StackA, StackB, StackC ve StackD class'ları bu interface'ı implement edecek.
- Interface içinde stack yapısı için istenilen push, pop, size, isEmpty, peek metotları tanımlanacak.

StackA:

- Stack yapısı ArrayList yapısının extend ederek uygulanacak!
- ArrayList yapısının sahip olduğu diğer metotların kullanılmamasına özen gösterilmeli.

StackB:

- ArrayList yapısından oluşturulan bir attribute'e sahip stack yapısı olacak.

StackC:

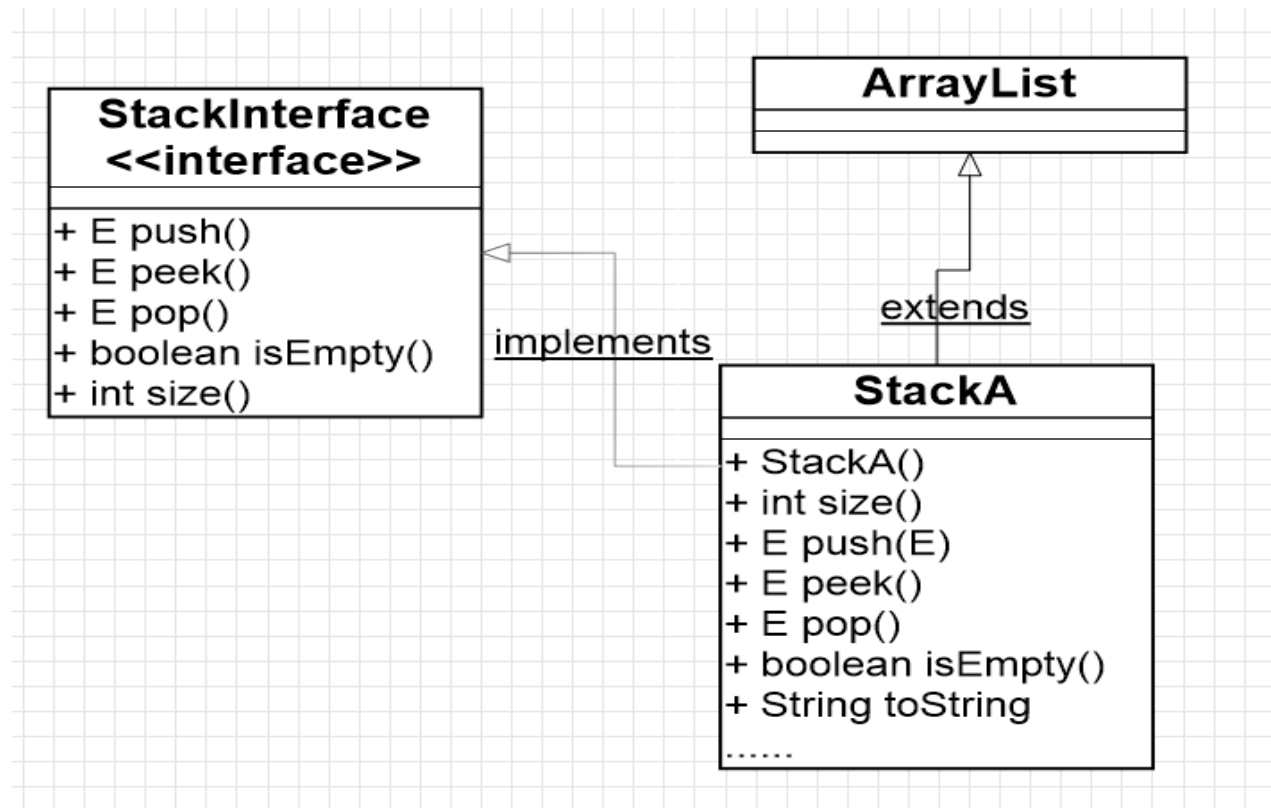
- Node'ların kullanılacağı bir stack yapısı.
- İçinde Node inner class'ı oluşturulmalı.

StackD:

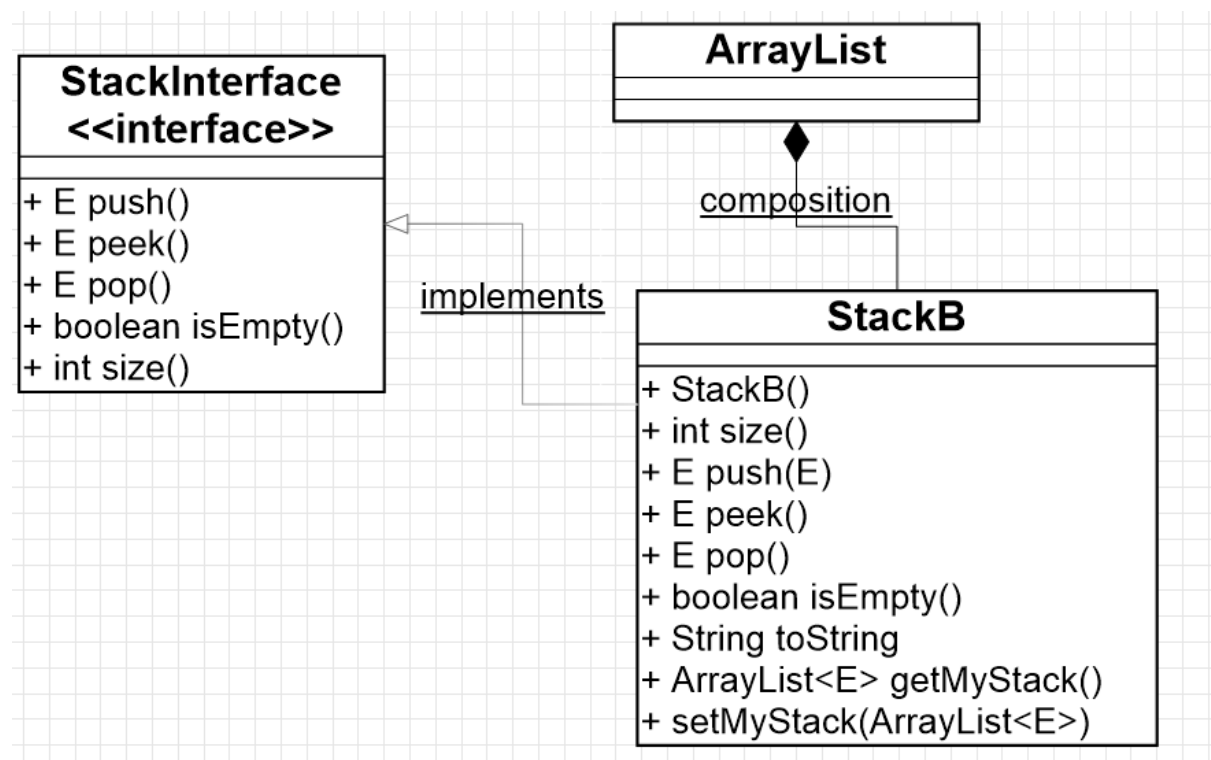
- Queue yapısından oluşturulan attribute'e sahip stack yapısı.

2. Class Diagrams

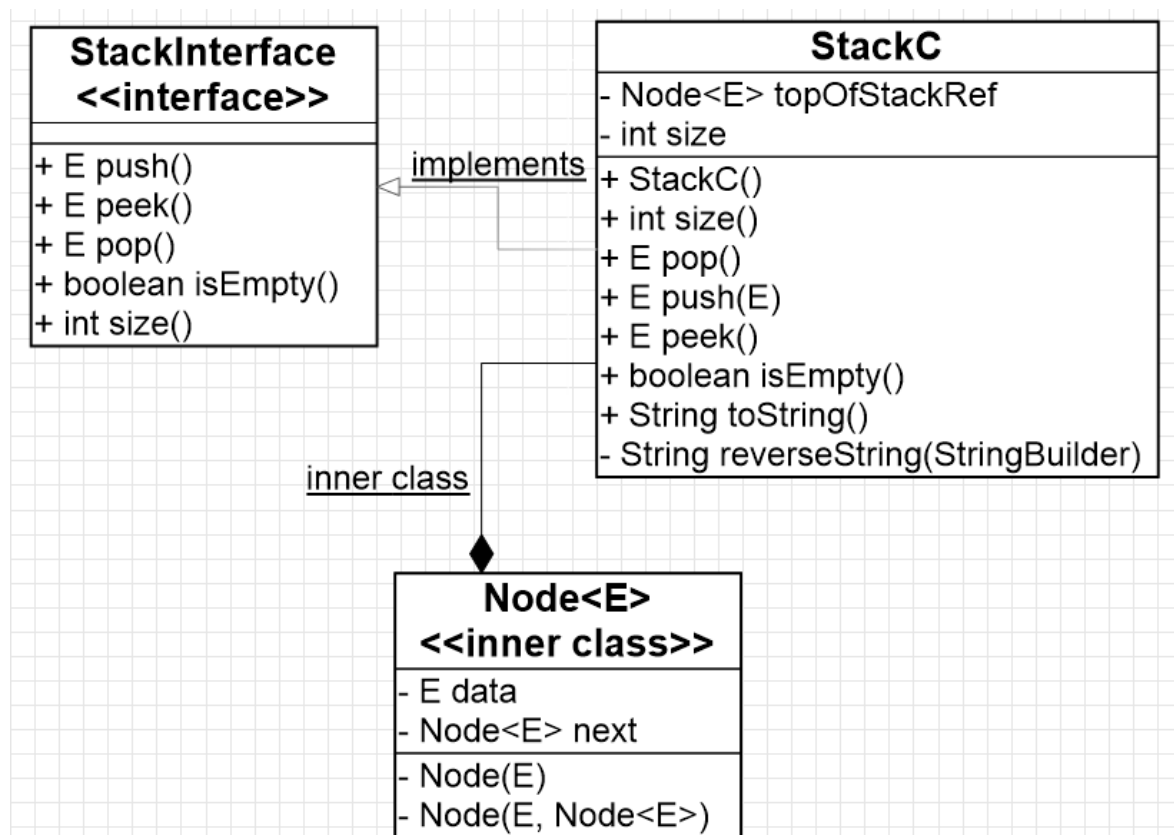
StackA:



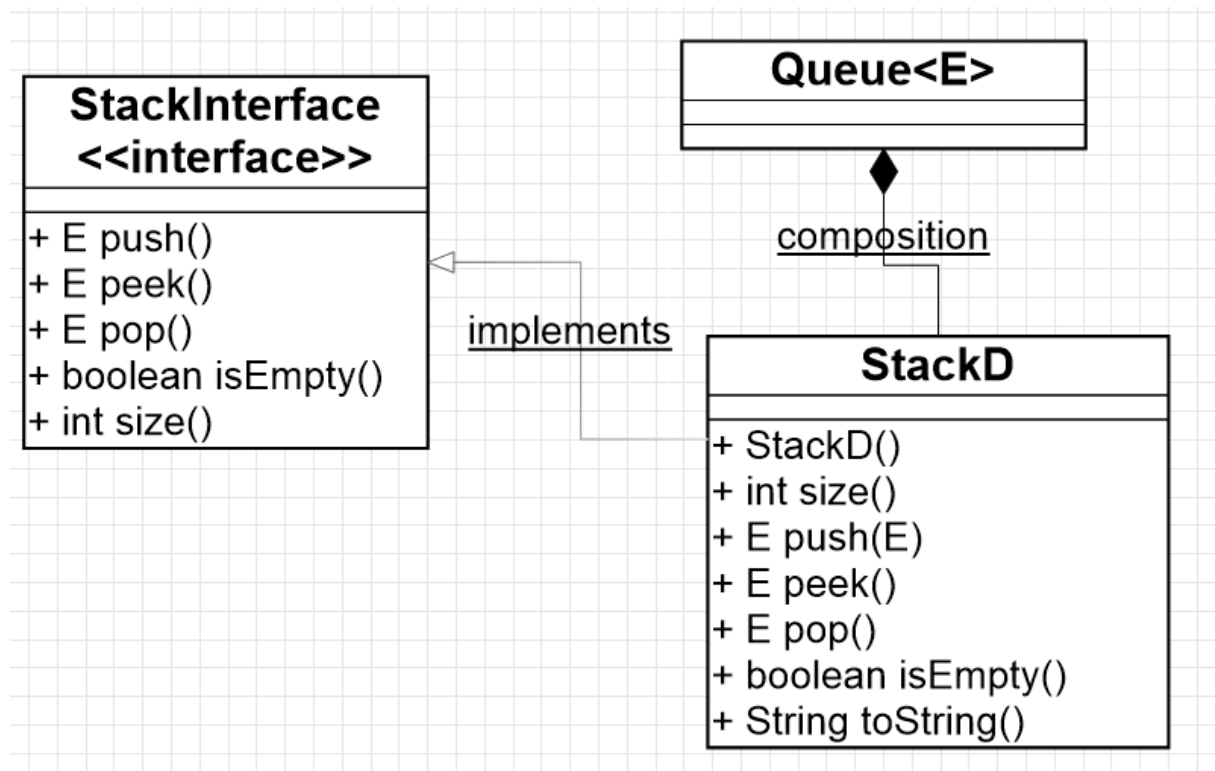
StackB:



StackC:



StackD:



3. Problem Solution Approach

StackInterface:

- Bütün stack yapıları için oluşturulup kullanılan bir interface.
- Abstract Class yapılmasına gerek yok. Çünkü bütün stack yapılarının içerdiği metotlar dışındaki her şey farklı. Buna metotların implementasyonları da dahil.
- Interface içinde sadece bütün stack yapıları için ortak olan metotlar tanımlandı!

StackA:

- ArrayList yapısını extend eden stack yapısı.
- ArrayList yapısını extend ettiği için içinde bir attribute yok.
- Metotlar ArrayList yapısının sahip olduğu metotlar kullanılarak implement edildi.
- ArrayList yapısından gelen metotlar kullanılmayacak şekilde ayarlandı.

StackB:

- ArrayList yapısını attribute(member) yaparak kullanan yapı.
- ArrayList attribute üzerinde işlemler gerçekleştirildi. İşlemlerden kastım StackInterface'inden gelen metotlar içinde istenilen şeyler ArrayList yapısına uygulandı.

StackC:

- İçerisinde Node inner class'ı oluşturulan ve bu Node class'ının kullanımı ile gerçekleştirilen stack yapısı
- İçerisinde bir node class'ı oluşturuldu. (Node class'ı sadece sonraki elemana ulaşabilen bir Node yapısı)
- Node class'ı ile stack yapısının başını gösteren bir attribute (topOfStackRef) tanımlandı.
- Metotlar , yine StackInterface interface'inden gelen metotlar, implement edildi.

StackD:

- İçinde Queue yapısından bir attribute tanımlanarak oluşturulan stack yapısı.
- Stack için yapılması istenilen işlemler bu member üzerinde uygulandı.
- StackInterface interface'inden gelen metotlar implement edildi.

4. Test Cases

- Okunacak dosya içindeki ilk satır Integer olarak, ikinci satır Double olarak, üçüncü satır Character olarak ve dördüncü satırda String olarak okunacak! (Okunacak dosya da ilk satıra integer, ikinci satıra double, üçüncü satıra karakter ve son satıra da string olarak degerler girdim.)
- Okunma sırası integer, double, character ve string olduğu için dosyada da değerlerin bu sırada olmasına dikkat edin.

5. Running Command and Results

Input file:

```
_q1.iml x StackA.java x StackA.java x
1 1,2,3,4,5,6,8,9,10
2 1.1,2.5,3.6,4.45,5.26
3 g,o,z,d,e,d,o,g,a,n
4 gozde, dogan, 131044019
5
```

Output file:

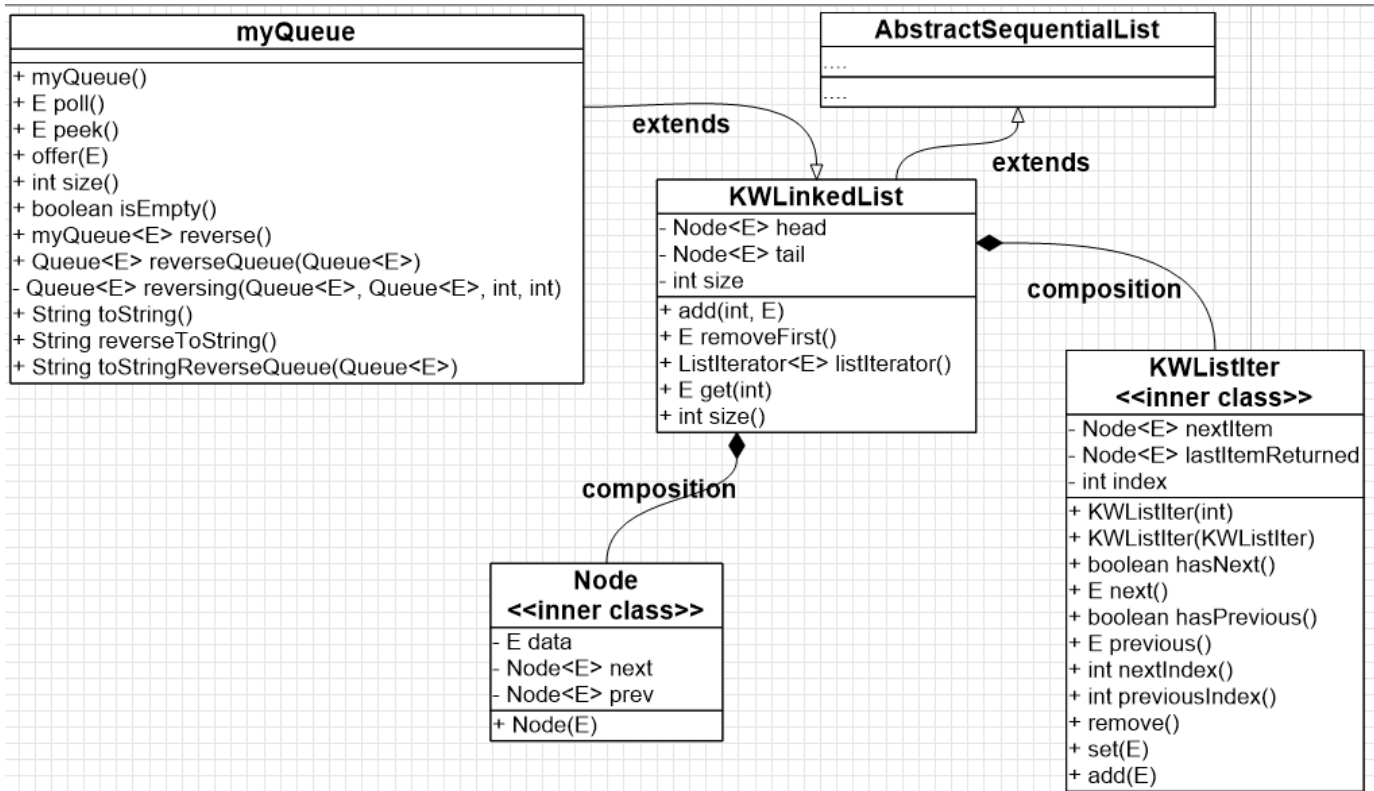
```
131044019_hw4_q1.iml x StackA.java x StackA.java x
3 1 9 ==> 1,2,3,4,5,6,8,9,10
2 5 ==> 1.1,2.5,3.6,4.45,5.26
3 10 ==> g,o,z,d,e,d,o,g,a,n
4 3 ==> gozde, dogan, 131044019
5 9 ==> 1,2,3,4,5,6,8,9,10
6 5 ==> 1.1,2.5,3.6,4.45,5.26
7 10 ==> g,o,z,d,e,d,o,g,a,n
8 3 ==> gozde, dogan, 131044019
9 9 ==> 1,2,3,4,5,6,8,9,10
10 5 ==> 1.1,2.5,3.6,4.45,5.26
11 10 ==> g,o,z,d,e,d,o,g,a,n
12 3 ==> gozde, dogan, 131044019
13 9 ==> 1,2,3,4,5,6,8,9,10
14 5 ==> 1.1,2.5,3.6,4.45,5.26
15 10 ==> g,o,z,d,e,d,o,g,a,n
16 3 ==> gozde, dogan, 131044019
17
```


2. Question2

1. System Requirements

- myQueue class'ı implement edilecek.
- myQueue class'ı KWLInkedList class'ını extend edecek.
- KWLInkedList class'ı ders kitabında yer alan class!
- myQueue class'ının implementation'una ek olarak myQueue'yu ters çeviren(reverse) bir metot(reverse) implement edilecek!
- myQueue class'ı içinde Queue objesi alan ve bu objeyi ters çeviren bir reverseQueue metodu implement edilecek!
- reverseQueue metodu recursive olarak implement edilecek!

2. Class Diagrams



3. Problem Solution Approach

- myQueue class'ını Queue gibi düşünüp Queue'nun sahip olduğu her şeyi gerçekleştirmesini sağladım.
- myQueue class'ım KWLinkedList class'ını extend ettiği için bir attribute'u yok.
- Attribute yerine direkt olarak extend edilen class üzerinden işlemlerimi gerçekleştirdim.
- poll, offer, size ve isEmpty metotlarını implement ettim.
- poll metodu queue yapısının ilk elemanını çıkarır.
- offer metodu queue yapısının sonuna eleman ekler.
- size metodu queue yapısının kaç elemana sahip olduğunu return eder.
- isEmpty metodu ise queue yapısının boş olup olmama durumu kontrol eder.
- poll, offer, size ve isEmpty metotlarının implementasyonu KWLinkedList yapısının metotları kullanılarak gerçekleştirildi.
- myQueue class'ı içine Queue objesi alıp bu objeyi ters çeviren bir reverseQueue metodunun implementasyonu yapıldı. Alınan Queue objesi ters çeviriliyor ve aynı zamanda return de ediliyor.
- reverseQueue metodunu recursive olarak yazarken helper metot kullanıldı.

4. Test Cases

- Okunacak dosya içindeki ilk satır Integer olarak, ikinci satır Double olarak, üçüncü satır Character olarak ve dördüncü satırda String olarak okunacak! (Okunacak dosya da ilk satıra integer, ikinci satıra double, üçüncü satıra karakter ve son satıra da string olarak degerler girdim.)
- Okunma sırası integer, double, character ve string olduğu için dosyada da değerlerin bu sırada olmasına dikkat edin.
- Programı çalıştırmak dışında bir şey yapmanıza gerek yok! Çalıştırdıktan sonra oluşan testResults_2.csv dosyasına bakmalısınız.

5. Running Command and Results

Input file:

```
KWLinkedList.java x myQueue.java x
1 1, 2, 3, 4, 5, 6, 8, 9, 10
2 1.1, 2.5, 3.6, 4.45, 5.26
3 g, o, z, d, e, d, o, g, a, n
4 gozde, dogan, 131044019
```

Output File:

```
KWLinkedList.java x myQueue.java x testir
1 myQueue ->>
2 3 ==> gozde, dogan, 131044019
3 10 ==> g, o, z, d, e, d, o, g, a, n
4 5 ==> 1.1, 2.5, 3.6, 4.45, 5.26
5 9 ==> 1, 2, 3, 4, 5, 6, 8, 9, 10
6
7 reverse myQueue->>
8 3 ==> 131044019, dogan, gozde
9 10 ==> n, a, g, o, d, e, d, z, o, g
10 5 ==> 5.26, 4.45, 3.6, 2.5, 1.1
11 9 ==> 10, 9, 8, 6, 5, 4, 3, 2, 1
12
13
14 Queue ->>
15 3 ==> gozde, dogan, 131044019
16 10 ==> g, o, z, d, e, d, o, g, a, n
17 5 ==> 1.1, 2.5, 3.6, 4.45, 5.26
18 9 ==> 1, 2, 3, 4, 5, 6, 8, 9, 10
19
20 reverse Queue->>
21 3 ==> 131044019, dogan, gozde
22 10 ==> n, a, g, o, d, e, d, z, o, g
23 5 ==> 5.26, 4.45, 3.6, 2.5, 1.1
24 9 ==> 10, 9, 8, 6, 5, 4, 3, 2, 1
25
```

3. Question3

1. System Requirements

- Minimum elemanı en başta tutan bir queue yapısı tasarlanacak. 2 farklı şekilde;

PriorityQueueA:

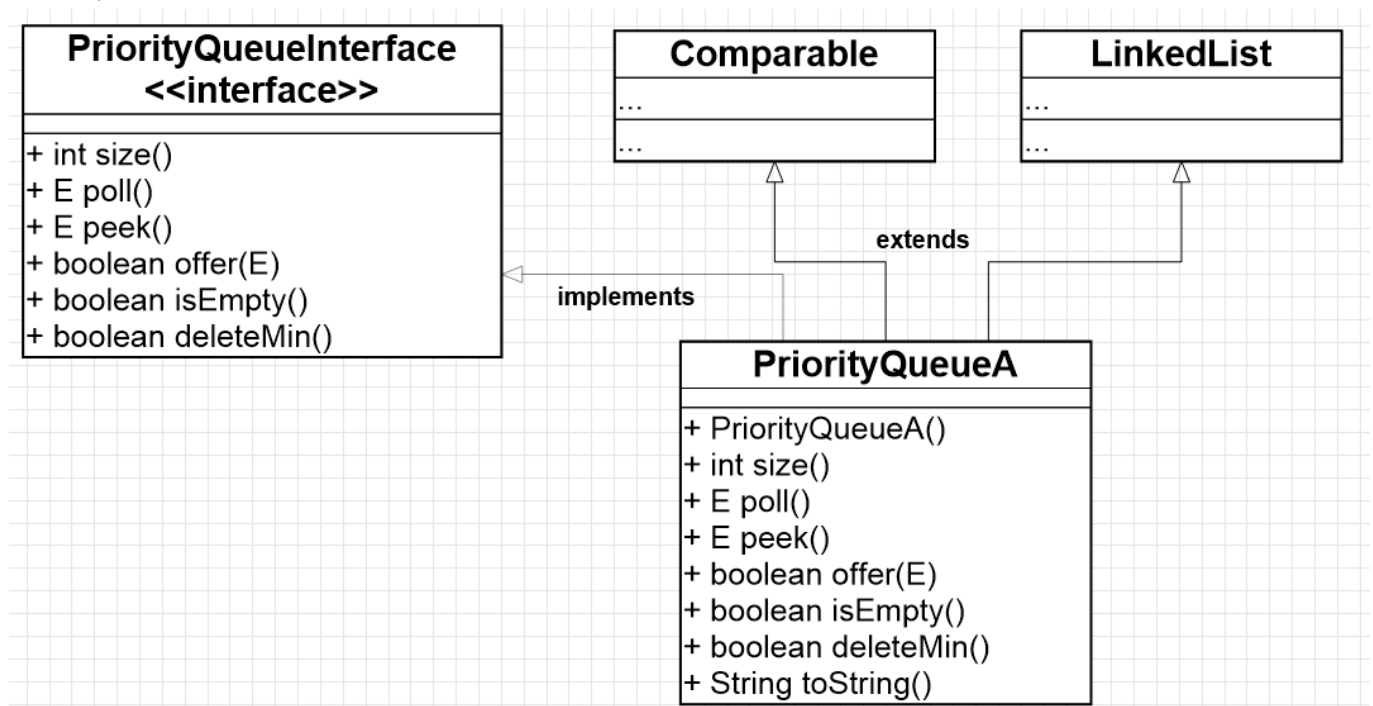
- LinkedList yapısının extend eden priority queue yapısı.
- Minimum elemanı hep başta tutacak ve deleteMin metodu çağrıldığında minimum elemanı silecek.

PriorityQueueB:

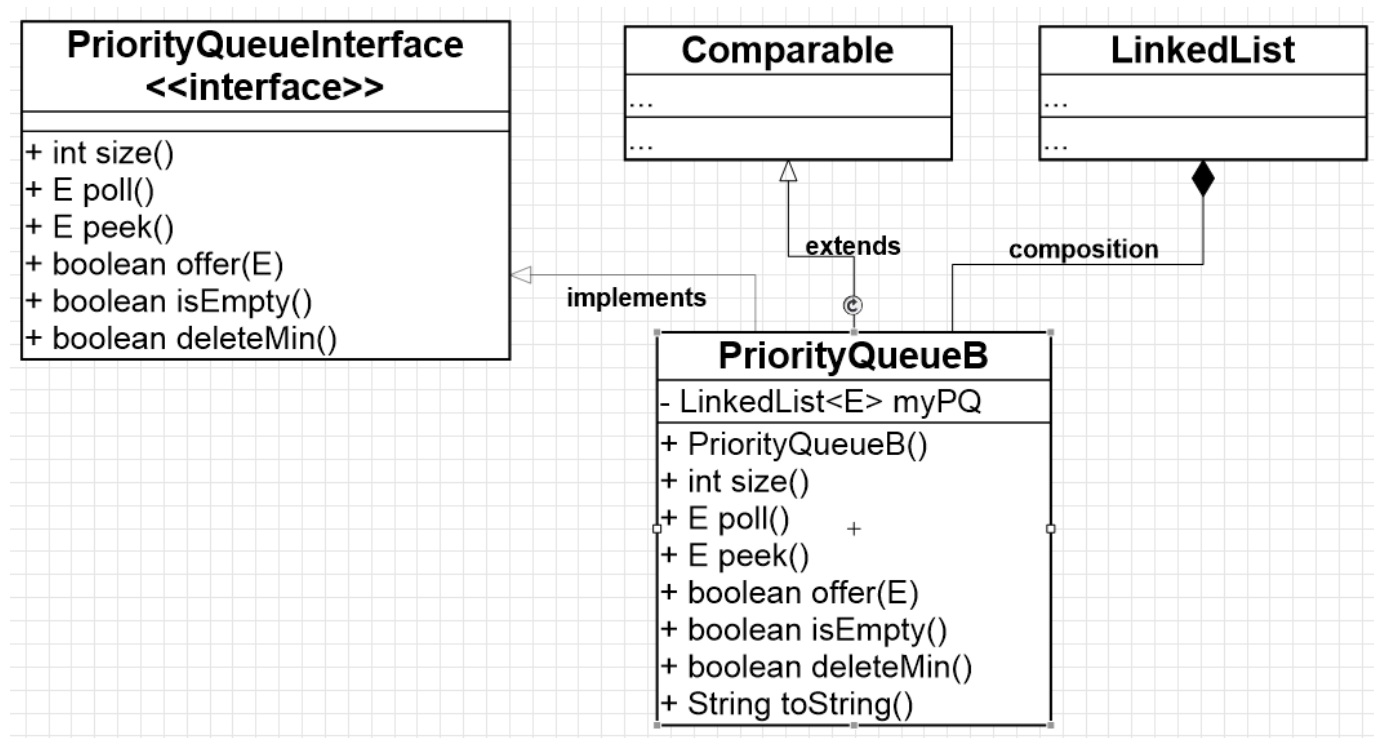
- LinkedList yapısından attribute'e sahip queue yapısı
- Minimum elemanı hep başta tutacak ve deleteMin metodu çağrıldığında minimum elemanı silecek.

2. Class Diagrams

PriorityQueueA:



PriorityQueueB:



3. Problem Solution Approach

PriorityQueueA:

- LinkedList yapısının extend eden priority queue yapısı.
- Minimum elemanı hep başta tutacak ve deleteMin metodu çağrıldığında minimum elemanı silecek.
- PriorityQueueInterface interface'ini implement eder.
- Kıyaslama yapabilmek için Comparable class'ını extend eder.

PriorityQueueB:

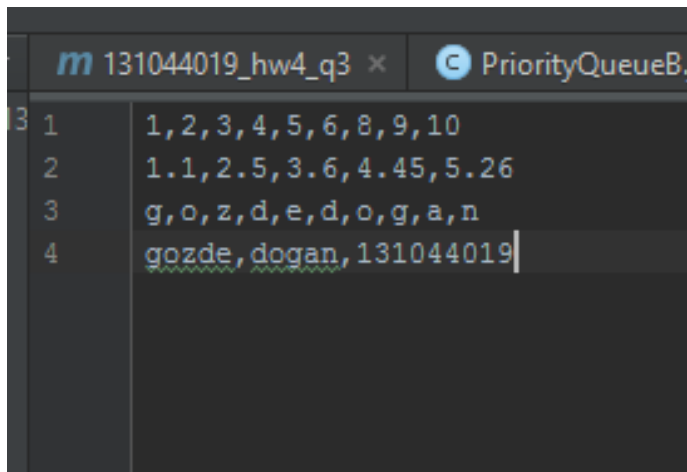
- LinkedList yapısından attribute'e sahip queue yapısı
- Minimum elemanı hep başta tutacak ve deleteMin metodu çağrıldığında minimum elemanı silecek.
- PriorityQueueInterface interface'ini implement eder.
- Kıyaslama yapabilmek için Comparable class'ını extend eder.

4. Test Cases

- test.csv dosyasından verilen okunuyor.
- 2 priority queue yapısı için de okuma işlemi gerçekleştirildi.
- Dosya içinde 1. Satır Integer, 2. Satır Double, 3. Satır Character ve 4. Satır da String kabul edilerek okuma yapıldı!
- PriorityQueueA ve PriorityQueueB yapıları(ikisi içinde Integer, Double, Character ve String tiplerinde obje oluşturuldu.) testResults_3.csv dosyasına yazılır.

5. Running Command and Results

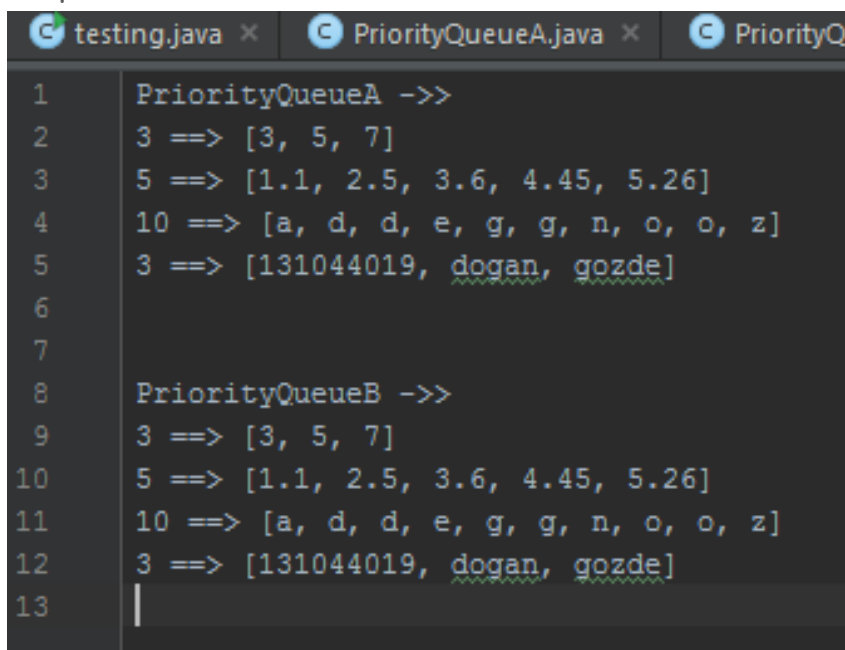
Input File:



The screenshot shows an IDE window with a tab labeled '131044019_hw4_q3'. The code editor displays the following content:

```
1 1,2,3,4,5,6,8,9,10
2 1.1,2.5,3.6,4.45,5.26
3 g,o,z,d,e,d,o,g,a,n
4 gozde,dogan,131044019
```

Output File:



The screenshot shows an IDE window with three tabs: 'testing.java', 'PriorityQueueA.java', and 'PriorityQueueB.java'. The 'testing.java' tab is active and displays the following output:

```
1 PriorityQueueA ->>
2 3 ==> [3, 5, 7]
3 5 ==> [1.1, 2.5, 3.6, 4.45, 5.26]
4 10 ==> [a, d, d, e, g, g, n, o, o, z]
5 3 ==> [131044019, dogan, gozde]
6
7
8 PriorityQueueB ->>
9 3 ==> [3, 5, 7]
10 5 ==> [1.1, 2.5, 3.6, 4.45, 5.26]
11 10 ==> [a, d, d, e, g, g, n, o, o, z]
12 3 ==> [131044019, dogan, gozde]
13
```

deleteMin test results:

For integer;

```
testing
"C:\Program ...
deleteMin testing----->>>>

For Integer -->>

queueAInt ->
BEFORE delete -> [3, 5, 7]
AFTER delete -> [5, 7]

queueBInt->
BEFORE delete -> [3, 5, 7]
AFTER delete -> [5, 7]
```

For double;

```
testing

For Double -->>

queueADouble->
BEFORE delete -> [1.1, 2.5, 3.6, 4.45, 5.26]
AFTER delete -> [2.5, 3.6, 4.45, 5.26]

queueBDouble->
BEFORE delete -> [1.1, 2.5, 3.6, 4.45, 5.26]
AFTER delete -> [2.5, 3.6, 4.45, 5.26]
```

For character;

```
For Character -->>

queueAChar->
BEFORE delete -> [a, d, d, e, g, g, n, o, o, z]
AFTER delete -> [d, d, e, g, g, n, o, o, z]

queueBChar->
BEFORE delete -> [a, d, d, e, g, g, n, o, o, z]
AFTER delete -> [d, d, e, g, g, n, o, o, z]
```

For string;

```
For String -->>

queueAStr->
BEFORE delete -> [131044019, dogan, gozde]
AFTER delete -> [dogan, gozde]

queueBStr->
BEFORE delete -> [131044019, dogan, gozde]
AFTER delete -> [dogan, gozde]

Process finished with exit code 0
```