

Gebze Technical University
Computer Engineering

CSE 222
2017 Spring

HOMEWORK 03 REPORT

MUSTAFA BİNGÜL 141044077

Course Assistant:

Nur Banu ALBAYRAK

Q1-/

1. Requirements

1.1-Overall Description

Ödevde StringBuilder gibi bir myStringBuilder sınıfı implement etmem istenmiştir. Ve implement ettiğim sınıf bir SingleLindedList yapısına sahip olmalı, append methodu ve 3 farklı toString() Methodu implement edilmeliydi. Ve programım 100.000 adet sayıyı numbers.txt dosyasından okuyup 3 farklı yazmış olduğum toString() methodları ile result1.txt,result2.txt ve result3.txt dosyalarına yazdırmalı.

1.2-System Requirements

numbers.txt.

2. Problem Solutions Approach

Öncelikle myStringBuilder ı yazabilmem için SingleLinkedList implement ettim. SingleLinkedList I implement ederken data ları tutacağım Node inner class ı yazdım. Node inner class ım bir sonraki node referansını ve data sını tutmakta. SingleLinkedList içinde bir tane Node tanımladım ve Node ların size ını tutabileceğim bir size değişkeni tanımladım. Verilerimi Node lara ekleyebileceğim gerekli tüm methodları tespit edip implement ettim. Aynı zamanda Node larımın üzerinde gezebileceğim bir Iterator sınıfı yazdım. Onu internetten yazarken internetten yardım aldım. (Referans gösterilmiştir.) SingleLinkedList sınıfımın implementi bittikten sonra myStringBuilder sınıfımı implement etmeye başladım. MyStringBuilder ımda datafield olarak implement ettiğim SingleLinkedList değişkenimi tanımladım.(SingleLinkedList sınıfımı ilkte Character array olarak tanımlamıştım. Ondan sonra değiştirip String yaptım. Hangisi doğru bilmiyorum ama PDF de de birşey belirtilmemişti.) myStringBuilder ıma da gerekli fonksiyonları implement ettim. Main methoduma da gerekli fonksiyoları çağırıp testlerimi gerçekleştirdim.

3.Analiyses

```
public String toString1(){
String str=new String();
for(int i=0; i<newstr.size(); i++) {
    if(newstr.get(i)!=null){
        str+=newstr.get(i);
    }
}
return str;
}
```

Yukarıdaki fonksiyonda newstr.size() bize SingleLinkedList in size sayısını verir yani Node sayımızdır. n tane Node var dersek, FOR için zaten n kez deneceğini söyleyebiliriz ancak IF statementının her seferinde gerçekleştiğini varsayarsak eğer içerisindeki get() methodunun çalışma süresi de n kez olur. Bu durumda bu fonksiyonun çalışma süresi için (worst-case) $BIG-O(n^2)$ deriz. Ancak FOR n kez döndüğünde IF statementinin hiç gerçekleşmediğini düşünürsek sadece FOR n kez döneceğinden bu fonksiyon için (best-case) $OMEGA(n)$ deriz.

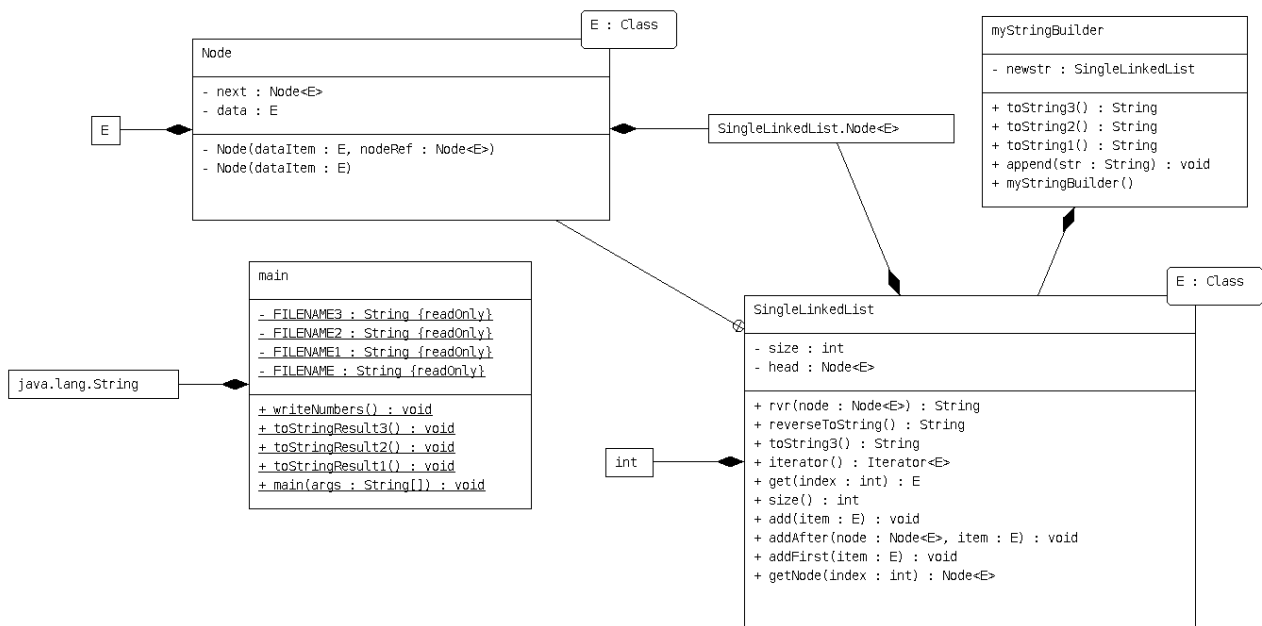
```
public String toString2(){  
  
    String str=new String();  
    Iterator itr=newstr.iterator();  
    int i=0;  
    while(itr.hasNext()){  
        str+=itr.next();  
        ++i;  
    }  
    return new String(str);  
}
```

Yukarıda ki fonksiyonda n adet Node var dersek, implement ettiğim iterator sınıfının hasNext() ve next() methodları constant time da gerçekleşir. Bu durumda bu method için (worst-case) BİG-O(n) diyebiliriz. Ancak eğer iterator herhangi bir noda sahip değilse hasNext() methodu çalışmayacağından ötürü bu method için (best-case) OMEGA(1) deriz.(constant time olur.)

```
public String toString3(){  
  
    String str=new String();  
    Node<E> nodeRef = head;  
    while(nodeRef!=null){  
        str+=nodeRef.data;  
        nodeRef=nodeRef.next;  
    }  
    return str.toString();  
}
```

Yukarıda ki method SingleLinkedList toString() idir. WHILE kaç adet Node var ise o kadar döner. n tane Node var dersek, method için (worst-case) BİG-O(n) deriz, (best-case) durumu için ise hiç Node olmamasıdır o da OMEGA(1) şeklinde ifade edilir.

4. Class Diagram



5. Test Cases

The screenshot shows the implementation of the Singly Linked List in an IDE. The **main.java** file contains the `main` method, which initializes a `SingleLinkedList` with the numbers 1, 2, 3, 4, 5 and tests its `reverseToString()`, `toStringResult1()`, `toStringResult2()`, and `toStringResult3()` methods. The **myStringBuilder.java** file contains the `myStringBuilder` class, which implements the `toString1()`, `toString2()`, and `toString3()` methods. The **SingleLinkedList.java** file contains the `SingleLinkedList` class, which implements the `add()`, `addAfter()`, `addFirst()`, `getNode()`, `iterator()`, `reverseToString()`, `toString3()`, and `size()` methods. The **numbers.txt**, **result1.txt**, **result2.txt**, and **result3.txt** files show the output of the tests.

```

// main.java
import java.io.*;

public class main {
    private static final String FILENAME = "numbers.txt";
    private static final String FILENAME1 = "result1.txt";
    private static final String FILENAME2 = "result2.txt";
    private static final String FILENAME3 = "result3.txt";

    public static void main(String args[]) {
        /*SingleLinkedList l = new SingleLinkedList();
        l.add("1");
        l.add("2");
        l.add("3");
        l.add("4");
        l.add("5");
        System.out.println(l.reverseToString());*/
        writeNumbers();
        toStringResult1();
        toStringResult2();
        toStringResult3();
    }

    /*MAIN END.*/

    /**
     * toString1();
     * Indexes ve get methodu ile yazılmış toString() ile d
     */
    public static void toStringResult1() {
        myStringBuilder mb = new myStringBuilder();

        try {
            BufferedReader br = new BufferedReader(new File
            String line = br.readLine();

            while (line != null) {
                mb.append(line);
                mb.append("\n");

                line = br.readLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

        FileWriter fw = new FileWriter(FILENAME1);
        fw.write(mb.toString());
        fw.close();
    }

    /**
     * toString2();
     * Indexes ve get methodu ile yazılmış toString() ile d
     */
    public static void toStringResult2() {
        myStringBuilder mb = new myStringBuilder();

        try {
            BufferedReader br = new BufferedReader(new File
            String line = br.readLine();

            while (line != null) {
                mb.append(line);
                mb.append("\n");

                line = br.readLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

        FileWriter fw = new FileWriter(FILENAME2);
        fw.write(mb.toString());
        fw.close();
    }

    /**
     * toString3();
     * Indexes ve get methodu ile yazılmış toString() ile d
     */
    public static void toStringResult3() {
        myStringBuilder mb = new myStringBuilder();

        try {
            BufferedReader br = new BufferedReader(new File
            String line = br.readLine();

            while (line != null) {
                mb.append(line);
                mb.append("\n");

                line = br.readLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

        FileWriter fw = new FileWriter(FILENAME3);
        fw.write(mb.toString());
        fw.close();
    }
}
  
```

writeNumbers(); methodu ile 100.000 adet sayı numbers.txt file ına yazılmaktadır. Aynı şekilde toStringResult1(); ,toStringResult2(); ve toStringResult3(); methodlarıda sırası ile belirtilen result1.txt,result2.txt ve result3.txt lere yazılmaktadır.

6. Running and Results

writeNumbers() methodu ile sayılar yazdırılır. toStringResult1(); ,toStringResult2(); ve toStringResult3(); methodlarıda çağırılarak toString() methodlarının dosyalara yazma işlemi gerçekleştirilir.

Sonuç olarak baktığımızda;

toStringResult1() methodu get() ve index ler ile gerçekleştirilmiştir ve worst-case i $BIG-O(n^2)$ dir.

toStringResult2() methodu iterator ile gerçekleştirilmiştir ve worst-case i $BIG-O(n)$ dir.

toStringResult3() methodu SingleLinkedList in toString() methodu ile gerçekleşmiştir ve worst case i $BIG-O(n)$ dir.

Bu durumda en yavaş olan toStringResult1() methodu dur.

Q2-/

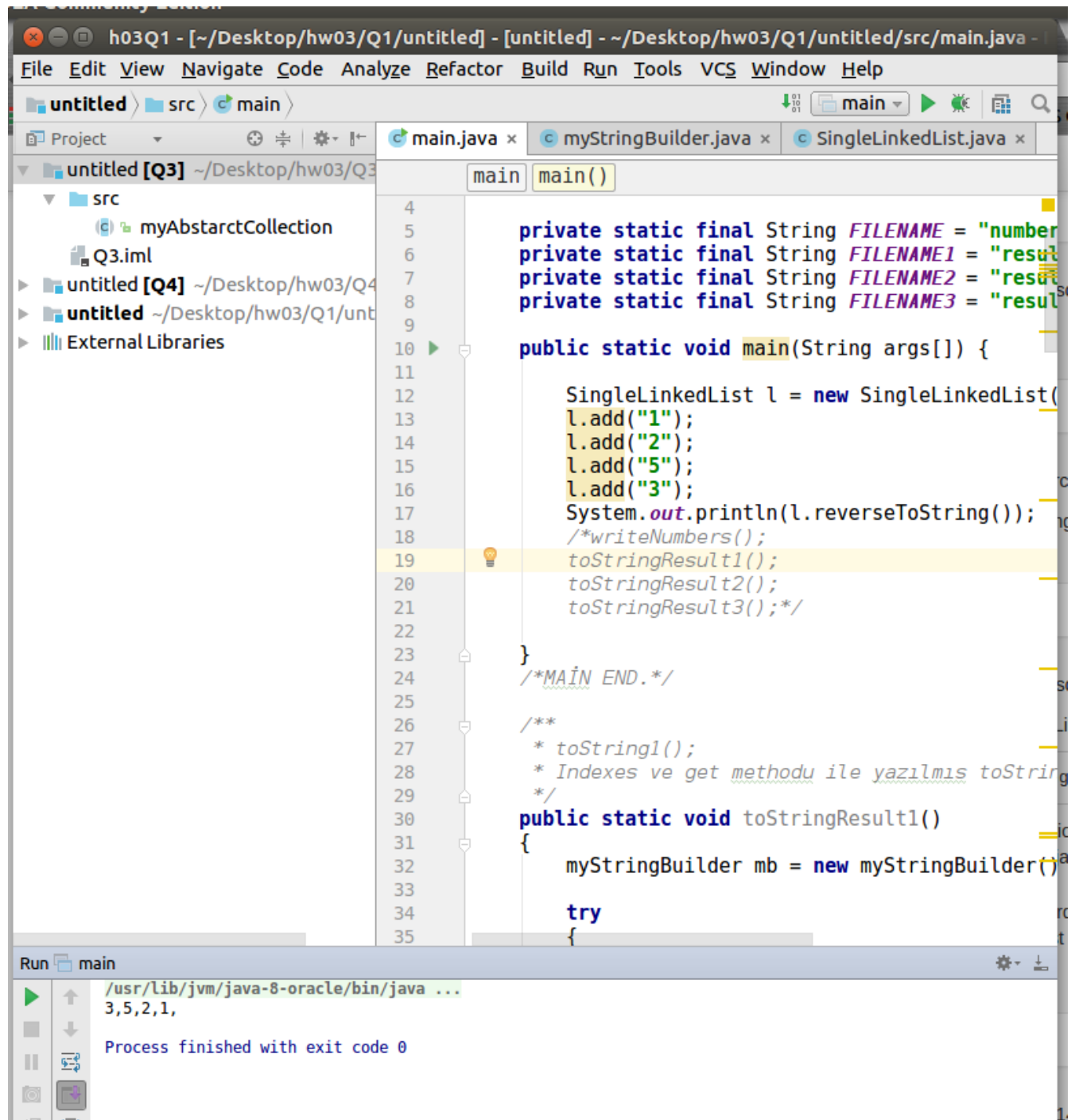
1. Requirements

1.1-Overall Description

SingleLinkedList için Node ları tersine çevirecek bir reverseToString() methodu implement etmem istenmiş. Recursive bir fonksiyon olacak.

NOTE: Kodu ayrı bir proje olarak yazmadım 1.(Proje) Sorunun SingleLinkedList sınıfının içine implement ettim ve 1.(Proje) soruda test ettim.

2. Test Cases



The screenshot shows an IDE window titled "h03Q1 - [~/Desktop/hw03/Q1/untitled] - [untitled] - ~/Desktop/hw03/Q1/untitled/src/main.java". The menu bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, and Help. The project explorer on the left shows a project named "untitled [Q3]" with a source folder "src" containing "myAbstractCollection", "Q3.iml", and "untitled [Q4]". The main editor displays the code for "main.java", with the "main" method selected. The code defines a "SingleLinkedList" and a "myStringBuilder" to reverse a linked list. The "main" method creates a "SingleLinkedList", adds elements "1", "2", "5", and "3", and prints the reversed string. The "toStringResult1" method uses a "myStringBuilder" to build the reversed string. The run console at the bottom shows the command "/usr/lib/jvm/java-8-oracle/bin/java ..." and the output "3,5,2,1,".

```
4 private static final String FILENAME = "number"
5 private static final String FILENAME1 = "result"
6 private static final String FILENAME2 = "result"
7 private static final String FILENAME3 = "result"
8
9
10 public static void main(String args[]) {
11
12     SingleLinkedList l = new SingleLinkedList()
13     l.add("1");
14     l.add("2");
15     l.add("5");
16     l.add("3");
17     System.out.println(l.reverseToString());
18     /*writeNumbers();
19     toStringResult1();
20     toStringResult2();
21     toStringResult3();*/
22
23 }
24 /*MAIN END.*/
25
26 /**
27  * toString1();
28  * Indexes ve get methodu ile yazılmış toString
29  */
30 public static void toStringResult1()
31 {
32     myStringBuilder mb = new myStringBuilder()
33
34     try
35     {
```

Run main

/usr/lib/jvm/java-8-oracle/bin/java ...
3,5,2,1,
Process finished with exit code 0

Q3-/

1. Requirements

1.1-Overall Description

AbstractCollection class ından extend edip bir myAbstractCollection classı implement etmemiz istenmiş. Ve bu abstract classımızın içinde iki tane myAbstractCollection objesini birbirine ekleyen appendAnything methodu implement etmemiz istenmiş.

2. Problem Solutions Approach

İki tane myAbstractCollection objesini birbirine toplamamız istenmiş. Bunun için kendi sınıfım için AbstractCollection sınıfını extend edip myAbstractCollection abstract sınıfını tanımladım ve iki myAbstractCollection objesini birbirine toplayabilmek için appendAnything(myAbstractCollection<E> other) methodunu implement ettim. Ancak bu yazdığım sınıfı kullanmak isteyenler Iterator ini ve add methodunu kendileri implement edecekler.

Q4-/

1. Requirements

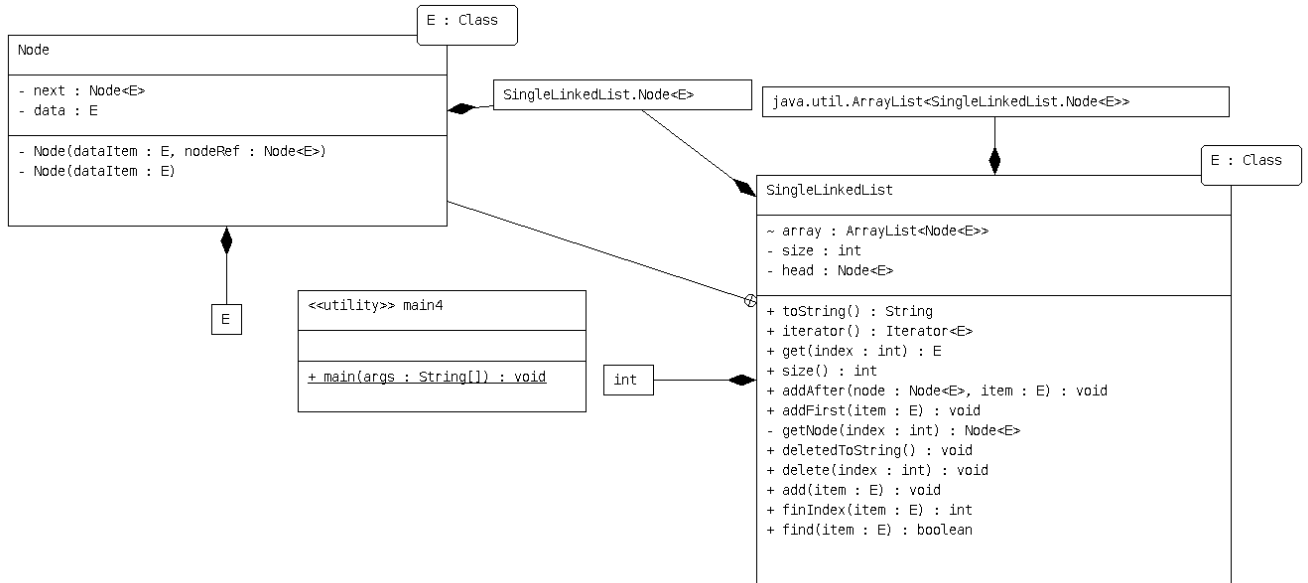
1.1-Overall Description

Bir SingleLinkedList class ı implement etmemiz ve bu class silinen node larını tekrar kullanılabilmesi istenmiş. Böylelikle garbage collector daha az kullanılacak. Aynı zamanda silinen node ları gösteren bir deletedToString methodu implement etmemiz istenmiş.

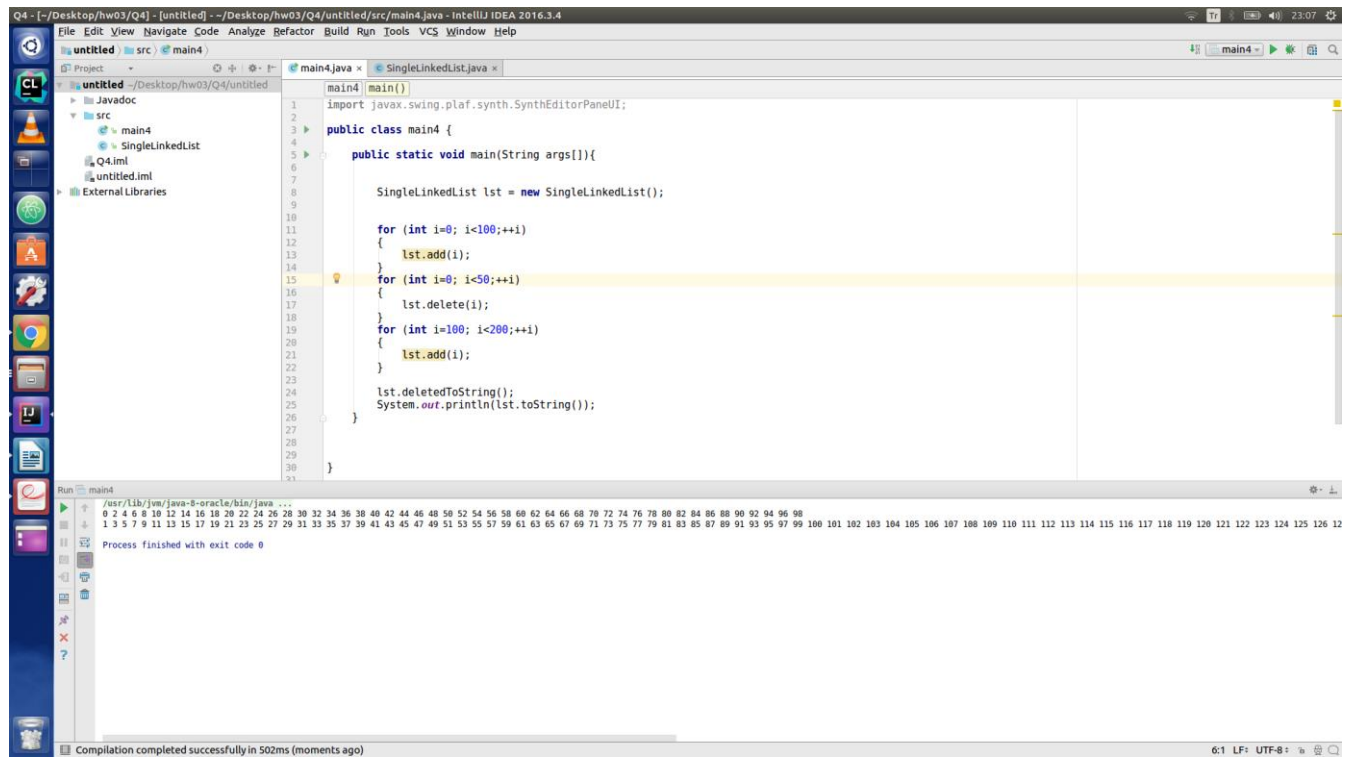
2. Problem Solutions Approach

Node ları tekrar kullanılabilir SingleLinkedList implement ettim. Ancak ilk te silinene node lar için bir Node tutuyordum fakat silindikten sonra tekrardan kendi asıl yerine node ları bağlayamayınca silinen Node lar için Node yerine ArrayList tuttum. Node tutmak kadar az olmasa da ArrayList te de garbage collector az kullanılıyor.

3.Class Diagram



4.Test Cases



```
Q4 - [-/Desktop/hw03/Q4] - [untitled] - ~/Desktop/hw03/Q4/untitled/src/main4.java - IntelliJ IDEA 2016.3.4
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Project: ~/Desktop/hw03/Q4/untitled
src
  main4
  SingleLinkedList
Q4.iml
untitled.iml
External Libraries

main4.java
1 import javax.swing.plaf.synth.SynthEditorPaneUI;
2
3 public class main4 {
4
5     public static void main(String args[]){
6
7         SingleLinkedList lst = new SingleLinkedList();
8
9
10
11         for (int i=0; i<100;++i)
12         {
13             lst.add(i);
14         }
15         for (int i=0; i<50;++i)
16         {
17             lst.delete(i);
18         }
19         for (int i=100; i<200;++i)
20         {
21             lst.add(i);
22         }
23
24         lst.deletedToString();
25         System.out.println(lst.toString());
26     }
27
28
29
30 }

Run main4
/usr/lib/jvm/java-8-oracle/bin/java ...
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
Process finished with exit code 0

Compilation completed successfully in 502ms (moments ago) 6:1 LF: UTF-8
```

For ile 100 adet sayı ekleyip 50 tanesini siliyorum. Sonra tekrardan 100 adet sayı ekliyorum.

5. Running and Results

add methodu ile verileri ekliyorum ve delete methodu ile siliyorum. Fakat silme işlemini index e göre yaptığımdan size boyutu yarıya indiğinde size boyutundan büyük index tekini silmeye kalkınca silmiyor hata veriyor. Yani silme işlemi indexe ve sizea bağlıdır.

<https://github.com/mstfbngl/HW03>