Gebze Technical University
Computer Engineering

CSE 222
2017 Spring

HOMEWORK 3 REPORT

Amine Yeşilyurt
131044004

Course Assistants: Nur Banu Albayrak / Şeyma Yücer

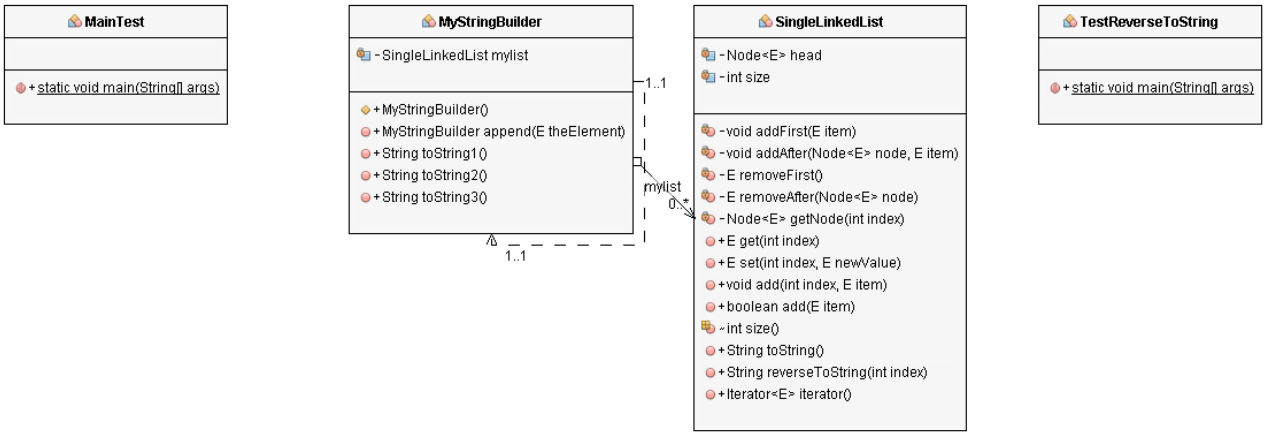
Q1:

1. System Requirements

Implemented a myStringBuilder class which works like the StringBuilder class. This class have a single linked list, append method to append anything and 3 different toString methods.
toString1( ) : Uses indexes and get method
toString2( ) : Uses iterator
toString3( ) : Uses toString method of the linked list

Written a main class which reads 100.000 integers from the numbers.txt file, uses myStringBuilder to create a string using 3 different toString methods and Prints their output to reult1.txt, result2.txt and result3.txt files.

2. Class Diagrams



3.Test Cases and Running and Results

All test cases written in one main test. Main test :

while (scanner.hasNext()) {

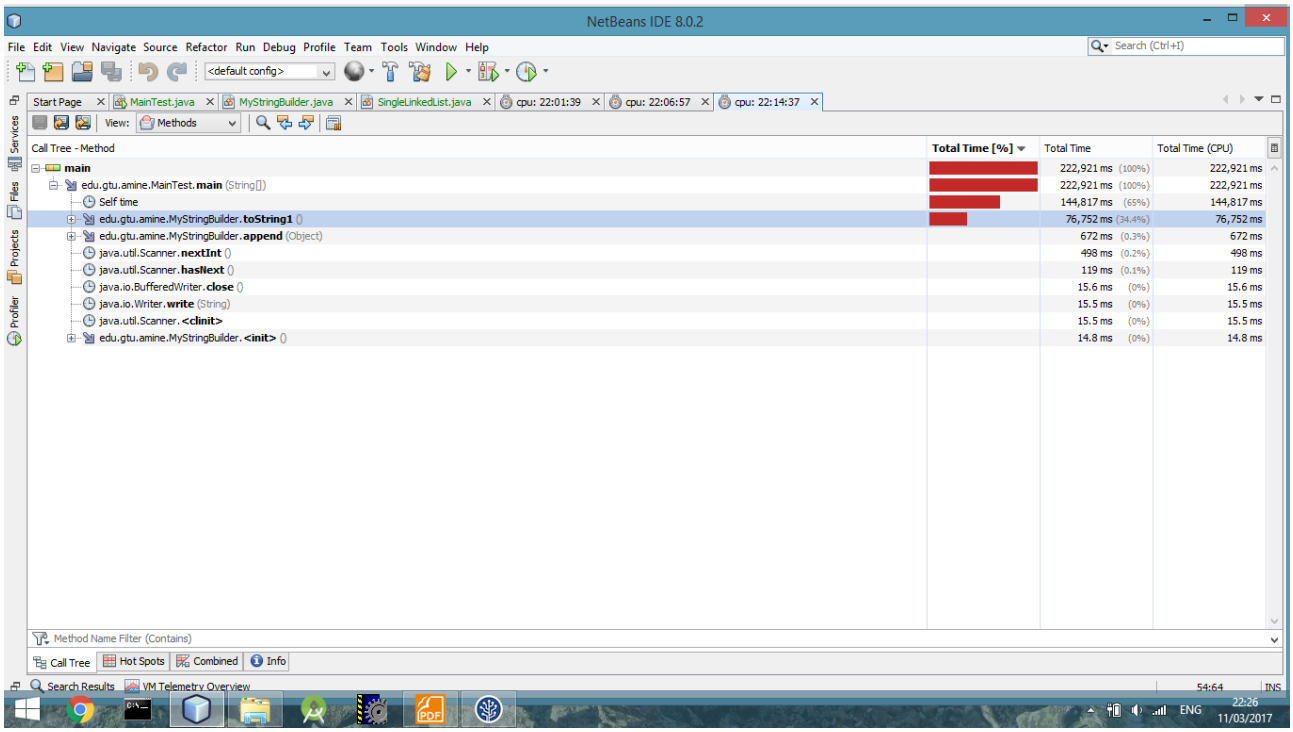
stringbuilderX.append(scanner.nextInt());

}

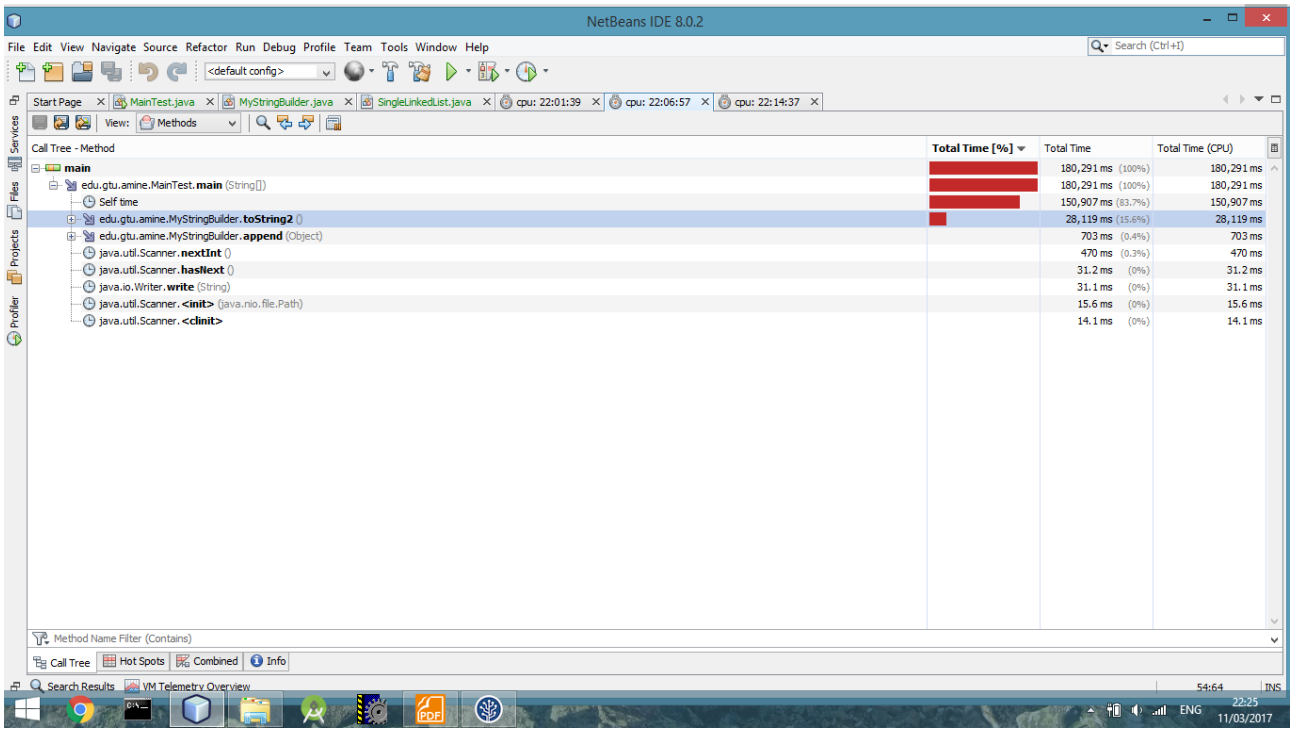
In the above code fragment , the numbers readed from the file one by one and appended to list using append method.

writer.write(sb2.toStringX()); // calling toString method from MainTest

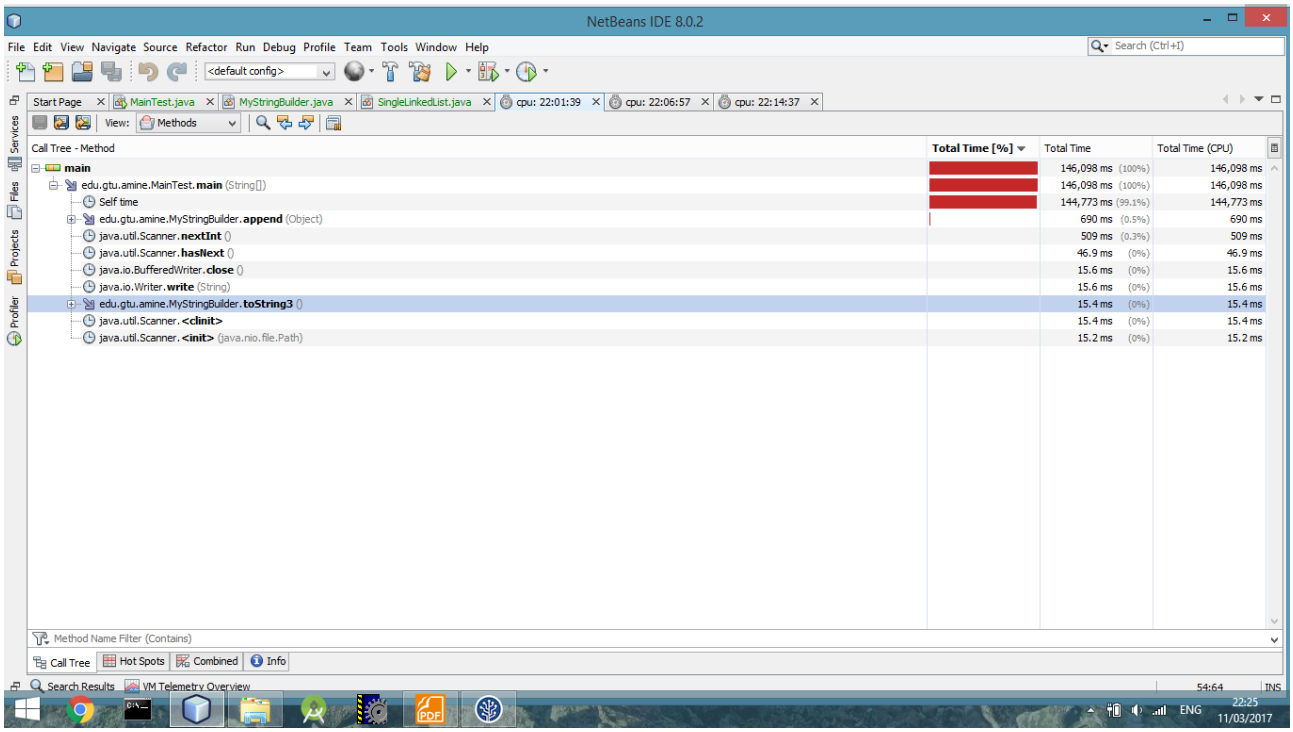
Testing toString1 ( ) Method : Uses indexes and get method



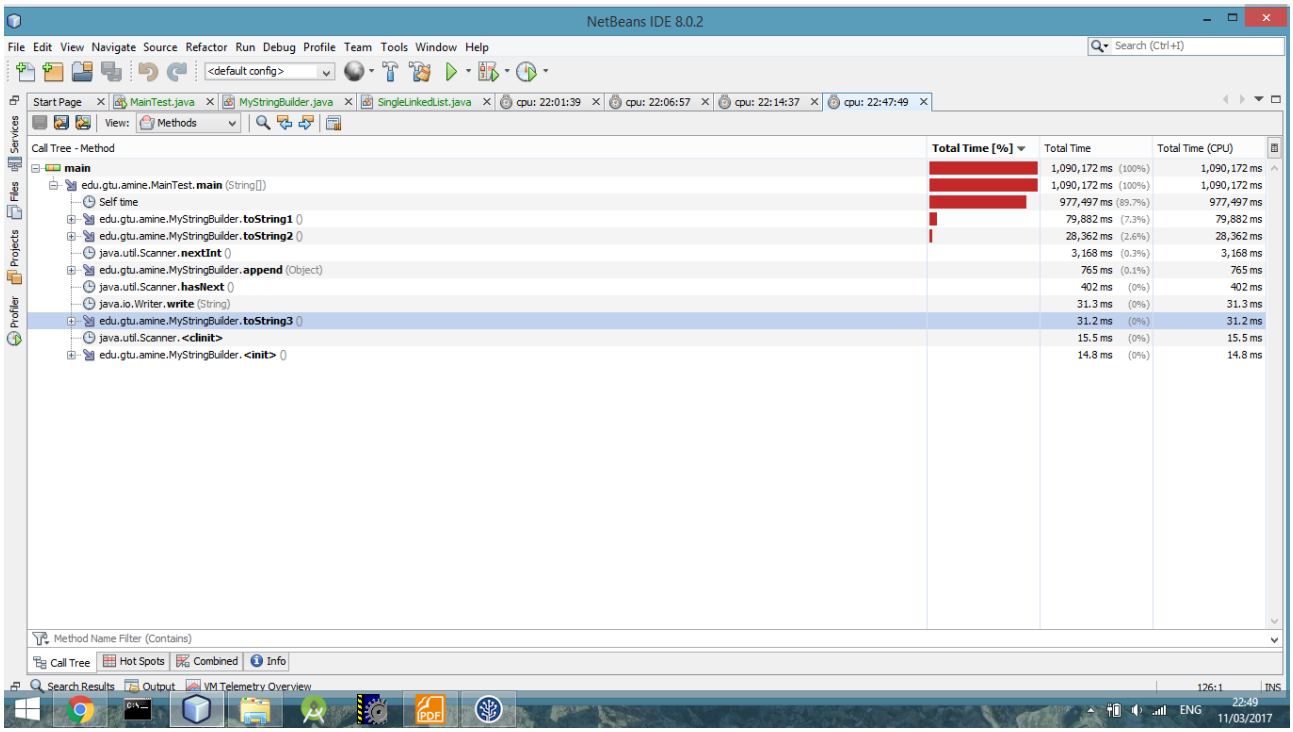
Testing toString2 ( ) Method : Uses iterator



Testing toString2 ( ) Method : Uses toString method of the linked list



Testing all toString methods together :



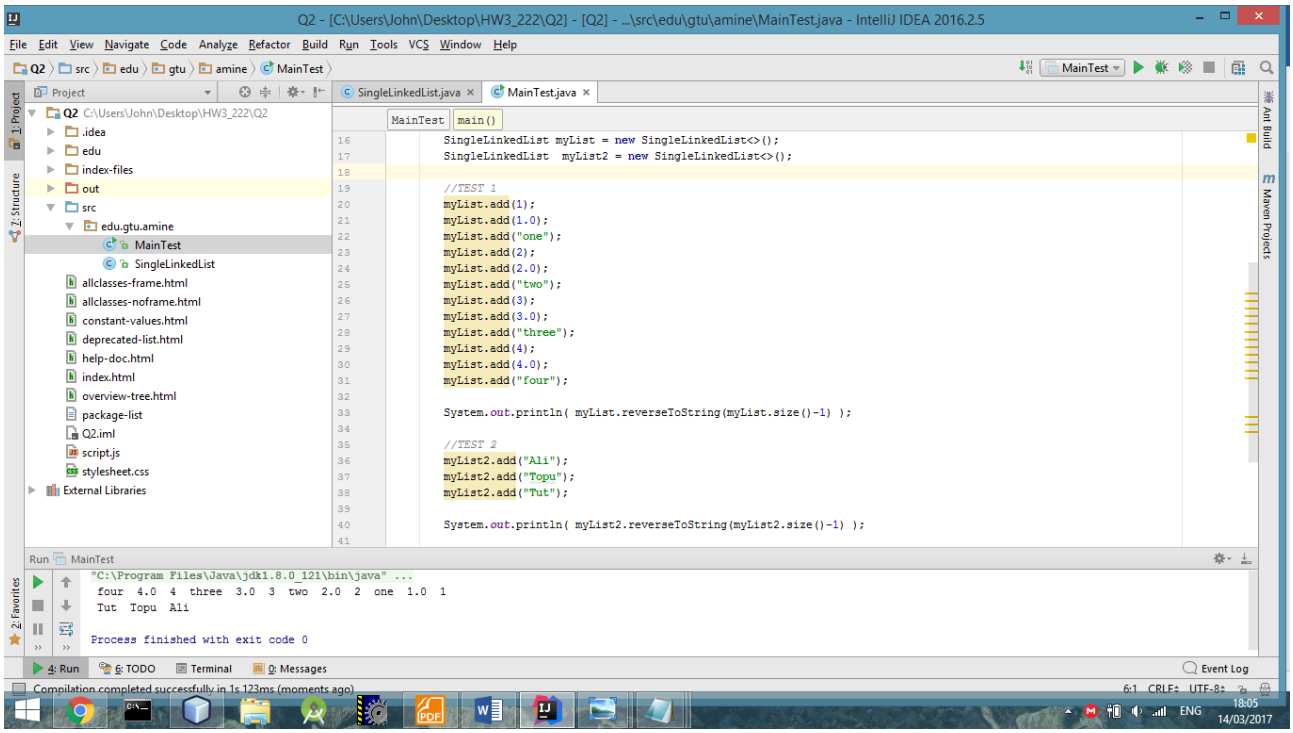
toString3( ) : 0% of total time
toString2( ) : 2.6% of total time
toString1( ) :7.3 % of total time

As we can see from CPU snapshots , toString1( ) method almost took double time according to toString2( ). And toString3( ) methods took very shorter time than others.

Q2:

Implemented a reverseToString method for the SingleLinkedList class which creates a reversed String. This method works recursively.
Written a main class to test this method.

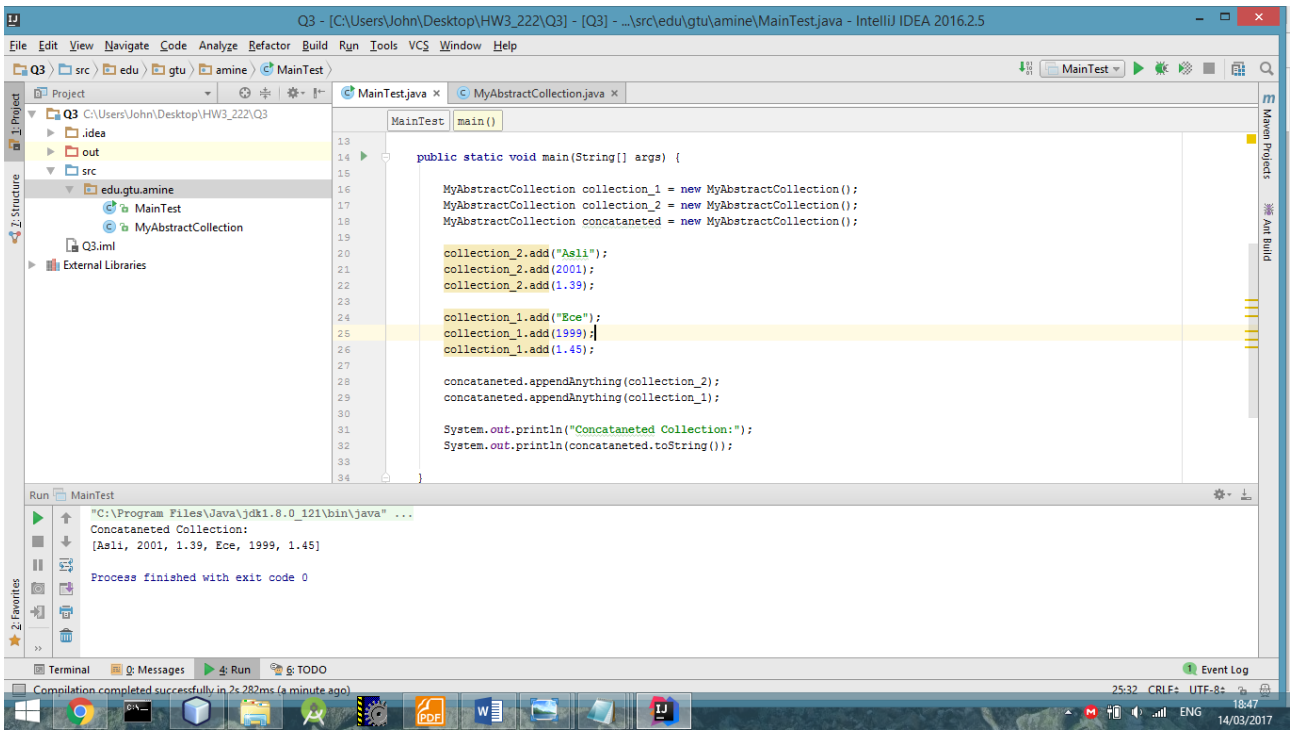
Main Test and Result:



Q3:

Extended AbstractCollection class as myAbstarctCollection and implemented an appendAnything method for myAbstractCollection class which appends any myAbstractCollection object to any other myAbstractCollection object by concatenating them.

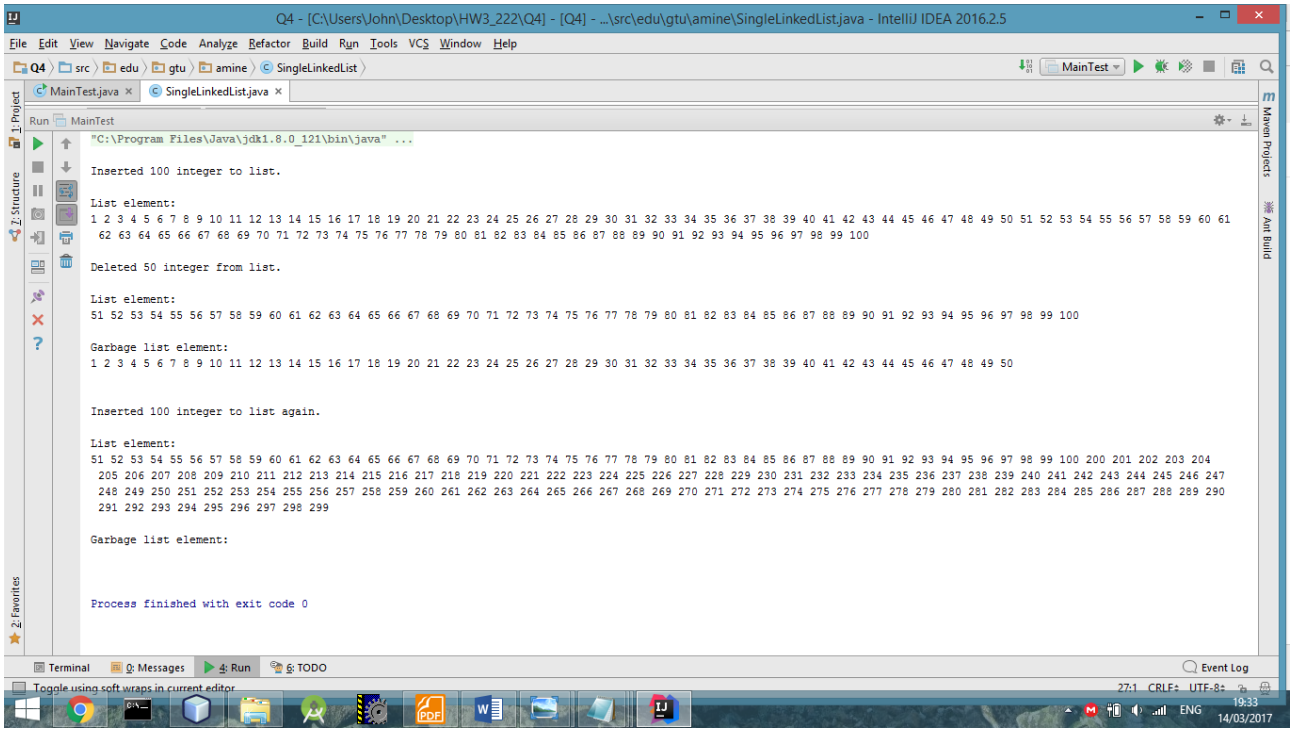
Resulsts:



Q4:

Implemented a SingleLinkedList class which reuses deleted nodes. It keeps deleted nodes. When required, instead of creating a new node, it uses one of the deleted nodes. This way it does less garbage collection.
It has a deletedToString method which creates a String of deleted nodes.
Written a main class to test SingleLinkedList. First of all inserted100 integers then deleted 50 of them and finally insered 100 more integers.

Results:



Problem solutions approach:
I used books's code for Single Linked List implementation. I implemented private class MyIterator<E> implements Iterator<E> as inner class. And i declare another list (deletedHead node and deletedSize)to keep the deleted nodes. This second list has own methods. These methods are:

- `public E getFromDeletedList(int index)`
- `public E setToDeletedList(int index, E newValue)`
- `public void addtoDeletedList(int index, E item)`
- `public boolean addtoDeletedList(E item)`
- `public String deletedToString()`

By using above methods, when a node deleted from list ,it inserted to second list (which keeps deleted nodes) and When required , it uses one of the deleted nodes. If list is empty which keeps deleted nodes, then new node created using **new keyword ,** otherwise this keyword does not use.