University of Newcastle
Discipline of Computing and Information Technology
Semester 2, 2020 - SENG1120

Assignment 1

Due using the Blackboard Assignment submission facility:
11:59PM – September 16th, 2020

Version 1.01 – changes/corrections have been highlighted; deprecated text remains, but has been ~~struck-through~~.

NOTE: *The important information about submission and code specifics at the end of this assignment specification***.**

INTRODUCTION

You are required to build the infrastructure to manipulate data related to text. Your client further specifies that you are to create a class named `LinkedList` to store individual tokens. The `LinkedList` will store each token in a `Node` of the list, using the class `string`.

ASSIGNMENT TASK

You are required to use a **doubly-linked list**, as discussed in lectures, to create your own implementation of the `LinkedList` class. It will use instances of `Node` to store instances of `value_type` (in this assignment, each `Node` will be used to store an instance of `string`).

The `LinkedList` class will be used by a main program, to be supplied to you, as well as a `makefile`. You will need to design `LinkedList` and `Node` in a way that it communicates seamlessly with the main program and compiles with the `makefile` also supplied. Please refer to the lecture slides and recordings for guidance on how to implement both classes.

You will need to implement constructors, overloaded operators, and a method that sorts the content of the linked list. You can use any sorting algorithm, but we recommend bubble sort. It is not efficient, but is very easy to implement (`https://www.geeksforgeeks.org/bubble-sort/`).

For students in SENG6120, there is an extra requirement:

- (3.0 marks) Extend the member method `void remove(string)` inside `LinkedList`. The extended method removes all occurrences of an input sentence. For instance, if the linked list contains the sentence “`The black cat was sitting on the black mat that was on the black floor`”, and the input string for `remove(string)` is “`on the black`”, then the resulting linked list must be “`The black cat was sitting mat that was floor`”.

For SENG1120 students who want to be challenged more, the above requirement becomes a bonus question, also worth 3.0 marks; however you can still only score a MAXIMUM of 15.0/15.0.

SUBMISSION

Make sure your code works with the files supplied, and DO NOT change them. For marking, we will add the demo .cpp and makefile to the project and compile everything using this makefile, together with your own files. If it does not compile or run, **your mark will be zero**.

Your submission should be made using the Assignments section of the course Blackboard site. **Incorrectly submitted assignments will not be marked.** You should provide the `.h` and `.cpp` files related to the linked list and node classes, only. Also, if necessary, provide a `readme.txt` file containing instructions for the marker. Each program file should have a proper header section including your name, course and student number, and your code should be properly documented. ~~There is no need for a coversheet.~~

Remember that your code should compile and run correctly using Cygwin. There should be no segmentation faults or memory leaks during or after the execution of the program.

Compress all your files into a *single .zip file*, using your student number as the filename. For example, if your student number is **`c9876543`**, you would name your submission:

`c9876543.zip`

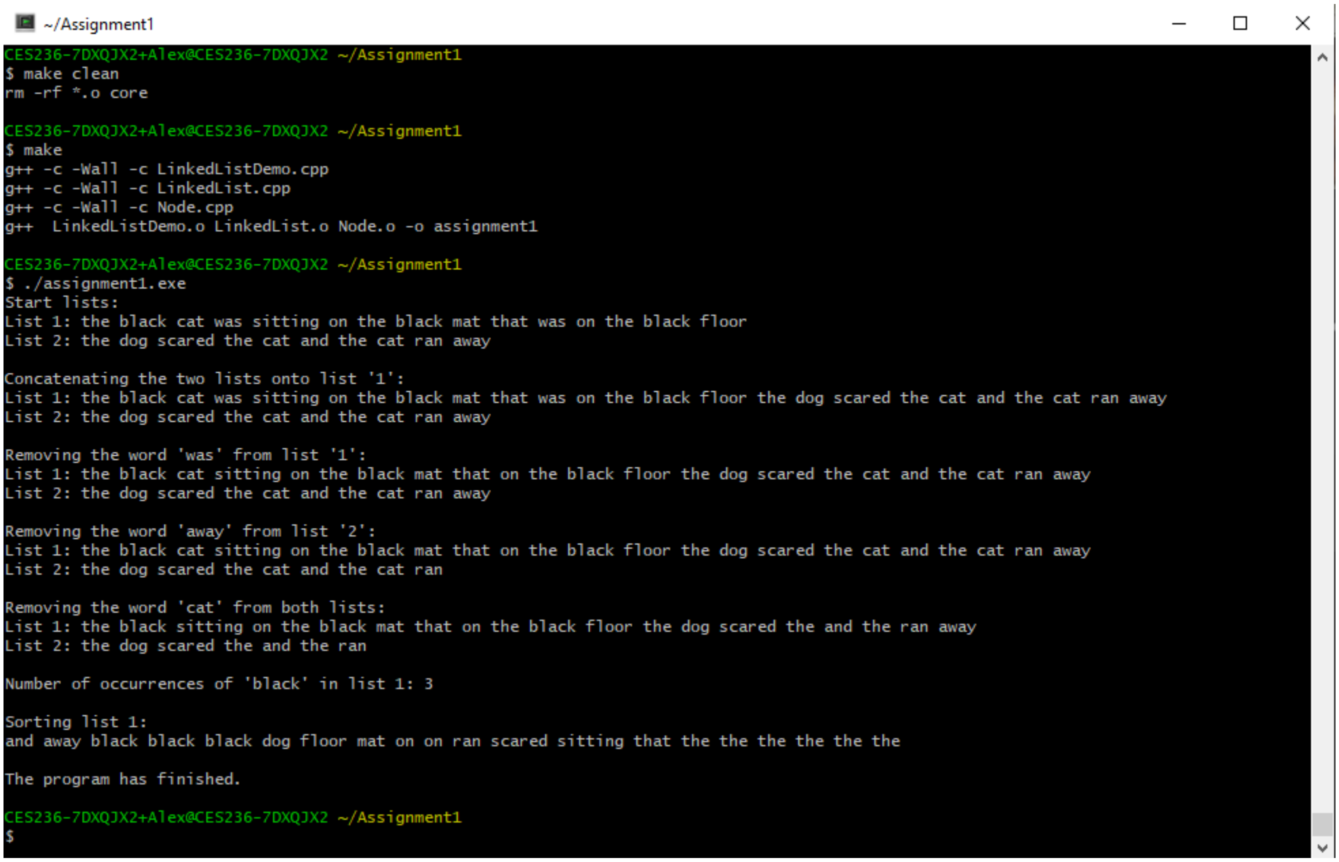
If you have attempted the Bonus Requirement (or you are a 6120 student), please include a blank text file in the same folder as your source files, simply called **`Bonus.txt`** – this is to make it clear to the marker that you are attempting this.

Submit by selecting the **Assignment 1** link that will be found in the **Assessment** section on **Blackboard**.

Late submissions are subject to the rules specified in the Course Outline. Finally, a completed **Assignment Cover Sheet** should accompany your submission.

This assignment is worth 15 marks of your final result for the course (including bonus marks).

Here is a screenshot of the expected output (*without the bonus question included*).



```
CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$ make clean
rm -rf *.o core

CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$ make
g++ -c -Wall -c LinkedListDemo.cpp
g++ -c -Wall -c LinkedList.cpp
g++ -c -Wall -c Node.cpp
g++  LinkedListDemo.o LinkedList.o Node.o -o assignment1

CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$ ./assignment1.exe
Start lists:
List 1: the black cat was sitting on the black mat that was on the black floor
List 2: the dog scared the cat and the cat ran away

Concatenating the two lists onto list '1':
List 1: the black cat was sitting on the black mat that was on the black floor the dog scared the cat and the cat ran away
List 2: the dog scared the cat and the cat ran away

Removing the word 'was' from list '1':
List 1: the black cat sitting on the black mat that on the black floor the dog scared the cat and the cat ran away
List 2: the dog scared the cat and the cat ran away

Removing the word 'away' from list '2':
List 1: the black cat sitting on the black mat that on the black floor the dog scared the cat and the cat ran away
List 2: the dog scared the cat and the cat ran

Removing the word 'cat' from both lists:
List 1: the black sitting on the black mat that on the black floor the dog scared the and the ran away
List 2: the dog scared the and the ran

Number of occurrences of 'black' in list 1: 3

Sorting list 1:
and away black black black dog floor mat on on ran scared sitting that the the the the the the

The program has finished.

CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$
```

```
CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$ make clean
rm -rf *.o core

CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$ make
g++ -c -Wall -c LinkedListDemo.cpp
g++ -c -Wall -c LinkedList.cpp
g++ -c -Wall -c Node.cpp
g++  LinkedListDemo.o LinkedList.o Node.o -o assignment1

CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$ ./assignment1.exe
Start lists:
List 1: the black cat was sitting on the black mat that was on the black floor
List 2: the dog scared the cat and the cat ran away

Concatenating the two lists onto list '1':
List 1: the black cat was sitting on the black mat that was on the black floor the dog scared the cat and the cat ran away
List 2: the dog scared the cat and the cat ran away

Removing the word 'was' from list '1':
List 1: the black cat sitting on the black mat that on the black floor the dog scared the cat and the cat ran away
List 2: the dog scared the cat and the cat ran away

Removing the word 'away' from list '2':
List 1: the black cat sitting on the black mat that on the black floor the dog scared the cat and the cat ran away
List 2: the dog scared the cat and the cat ran

Removing the word 'cat' from both lists:
List 1: the black sitting on the black mat that on the black floor the dog scared the and the ran away
List 2: the dog scared the and the ran

Number of occurrences of 'black' in list 1: 3

Sorting list 1:
and away black black black dog floor mat on on ran scared sitting that the the the the the the

The program has finished.

CES236-7DXQJX2+Alex@CES236-7DXQJX2 ~/Assignment1
$
```

Dan and Alex

v1.0 2020-08-20

v1.01 2020-09-08