CSCE 221 Assignment 3 Cover Page

First Name Jialu Last Name Zhao UIN 426000712

User Name zhaojialu123 E-mail address zhaojialu123@tamu.edu

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero, read more on Aggie Honor System Office website: `http://aggiehonor.tamu.edu/`

Type of sources				
People				
Web pages (provide URL)				
Printed material				
Other Sources				

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.
On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work.

Your Name Jialu Zhao Date 02/26/2017

CSCE 221 Assignment 3 – Part 1

Part 1 due to CSNet by March 1 with demonstration in labs on February 27/28.

Objective

This is an individual assignment which has three parts.

1. Part 1: C++ implementation of a Doubly Linked List for `int` type and next writing its templated version. The supplementary code is provided (download it from the class website).
2. Part 2: C++ implementation of queue and stack classes based on a templated Doubly Linked List (implemented in Part 1.
3. Part 3: C++ implementation of a simple calculator for evaluating an algebraic expression based on its postfix form. The queue and stack classes (implemented in Part 2) should be applied for obtaining the postfix form and expression evaluation.

Part 1: Implementation of Doubly Linked List

- Download the program `221-A3-code.tar` from the course website. Use the 7-zip software to extract the files in Windows, or use the following command in Linux.

```
tar xfv 221-A3-code.tar
```

- Two programs in separate folders are included.

1. "Doubly Linked List" for integers
   (a) Most code is extracted from the lecture slides. An exception structure is defined to complete the program.
   (b) You need to complete the following functions in the `DoublyLinkedList.cpp`.
      i. copy constructor
      ii. assignment operator
      iii. output operator
      iv. `insertFirst`
      v. `removeFirst`
      vi. `first`

      Make sure the i. and ii. functions do a deep copy of the input list, that is, copying each node one by one.
   (c) Type the following commands to compile the program

```
make
```

   (d) The main program includes examples of creating doubly linked lists, and demonstrates how to use them. Type the following command to execute.

```
./run-dll
```

2. Templated "Doubly Linked List" (general type)

(a) Convert the doubly linked list in the previous program to a template, so it creates lists of general types other than integer.

(b) Read C++ slides, page 16-22 at http://www.stroustrup.com/Programming/19_vector.ppt

(c) Follow the instructions below:

i. Templates should be declared and defined in a `.h` file. Move the content of `DoublyLinkedList.cpp` and `DoublyLinkedList.h` to `TemplateDoublyLinkedList.h`

ii. Replace `int obj` by `T obj` in the class `DListNode` so list nodes store general `T` objects instead of integers. Later when a `DListNode` object is created, say, in the main function, `T` can be specified as a `char`, a `string` or a user-defined class.

iii. To use a general type `T`, you must change each type declaration.

A. Replace variable declaration, input type and output type of functions `int` by general type `T`, except for the `count` variable.

B. Replace variable declaration, input type and output type of functions `DListNode` by `DListNode<T>` ,

C. Replace variable declaration, input type and output type of functions `DoublyLinkedList` by `DoublyLinkedList<T>`, including the friend class declaration

iv. Assign general default value `T()` instead of the original 0

v. To use general type `T` anywhere throughout the class `DListNode` and `DoublyLinkedList`, you must declare (add) `template <typename T>` before classes and the member functions defined outside the class declaration

vi. In each member function signature, replace `DoublyLinkedList::` by `DoublyLinkedList<T>::`

(d) Type the following commands to compile the program.

```
make
```

(e) The main program includes examples of creating doubly linked lists of "strings", and demonstrates how to use them. Type the following command to execute.

```
./run-tdll
```

- **Complexity Analysis**

  Comment each class member function you implemented with its time complexity using big-O notation. Specifically, comment on the loops.

- **What to submit to CSNet?**

  - Your source code for `DoublyLinkedList` and `TemplateDoublyLinkedList`
  - Written report with complexity using big-O notation for all the functions.
  - The screenshots of testing all the cases should be included in your reports.

Report

- Screen shots and explanation

1.DoublyLinkedList

```
[zhaojialu123]@linux2 ~/A3/221-A3-17a-code/DoublyLinkedList> (14:35:35 02/
:: ./run-dll
Create a new list
list:

Insert 10 nodes at tail with value 10,20,30,..,100
list: 10 20 30 40 50 60 70 80 90 100

Insert 10 nodes at front with value 10,20,30,..,100
list: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100

Copy to a new list
list2: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100

Assign to another new list
list3: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100

Delete the last 5 nodes
list: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50

Delete the first 5 nodes
list: 50 40 30 20 10 10 20 30 40 50

Make sure the other two lists are not affected.
list2: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100
list3: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100
```

(1)Create a new list: test constructor
(2)Insert 10 nodes at tail: test insertLast function
(3)Insert 10 nodes at front : test insertFirst function
(4)Copy to a new list: test copy constructor
(5)Assign to another new list: test assignment operator
(6)Delete the last 5 nodes: test removeLast function
(7)Delete the first 5 nodes: test removeFirst function
Note: All of them for integers
2.Template LinkedList

```
[zhaojialu123]@linux2 ~/A3/221-A3-17a-code/TemplateDoublyLinkedList> (14:3
:: ./run-tdll
Create a new list
list:

Insert 10 nodes at back with value 10,20,30,..,100
list: 10 20 30 40 50 60 70 80 90 100

Insert 10 nodes at front with value 10,20,30,..,100
list: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100

Copy to a new list
list2: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100

Assign to another new list
list3: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100

Delete the last 10 nodes
list: 100 90 80 70 60 50 40 30 20 10

Delete the first 10 nodes
list:

Make sure the other two lists are not affected.
list2: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100
list3: 100 90 80 70 60 50 40 30 20 10 10 20 30 40 50 60 70 80 90 100
```

(1)Create a new list: test constructor
(2)Insert 10 nodes at tail: test insertLast function
(3)Insert 10 nodes at front : test insertFirst function
(4)Copy to a new list: test copy constructor
(5)Assign to another new list: test assignment operator
(6)Delete the last 10 nodes: test removeLast function
(7)Delete the first 10 nodes: test removeFirst function
Note: All of them for strings

- Running time for functions

1. Copy constructor: O(n)
2. Assignment operator: O(n)
3. insertFirst function: O(1)
4. insertLast function: O(1)
5. removeFirst function: O(1)
6. removeLast function: O(1)
7. Destructor: O(n)
8. first function:O(1)
9. last function:O(1)
10. output operator <<: O(n)