CS102
Template Class Lab

First, in this lab you are to create a Node class as a template class that can store various types of data. The Node class will also have a pointer to the next node, so essentially a list of connected nodes can be created (it can be looked upon as a linked list). Methods to access the data and the next node in the list should also be provided. Lastly, a static class method display() should be implemented that would traverse the list and print out the data stored in it.

Second, you will have to create a simple Student class that stores the information about a student's name (null-terminated string with a maximum of 10 characters) and its student number (unsigned integer). To keep things simple, you don't have to provide set methods for the data members, but you should allow the data to be initialized during the object construction. Furthermore, you should overload << and provide a print() method, which will basically, when invoked, print out the information about the student according to the following format:

SID: 1234 Name: Andrea

(SID:@##…#\tName:@ABC…YZ\n

where @-blank
\t tab
\n newline
- digit
A-Z letter)

Once both of these classes are created, a driver program should be created.
In the driver, you are going to create a list of Node objects (Node objects linked together) of length 5 or more, that are going to store Student objects in them. Lastly, you should traverse the list using the Node class display() static method that would essentially display all of the Students that are currently stored in it.

An example of how to create a Node class and a driver that stores int primitive data type is given on the following URL:

http://www.bridgeport.edu/~dichter/cs102/templateClass.handout.doc

It is also repeated on the next page of this Lab assignment.

Example of a Node Template class with Simple Example

```
// Template Class code
// Written by Julius Dichter
// © 2003

#include <iostream>
using namespace std;

template <class T>
class Node {
public:

Node(T, Node<T> * = NULL);

T getData() { return data; }
Node<T> * getNext() { return next; }
void static display(Node<T> *);

private:
T data;
Node<T> * next;  } ;

// implementation section

template <class T>
Node<T>::Node(T data, Node<T> * next) : data(data), next(next) { }

template <class T>
void static display(Node<T> * head) {

        while (head != NULL) {
                cout  << head->getData() << endl;
                head = head->getNext(); } }
```

```
// Template Test code
// Written by Julius Dichter
// © 2003

#include <iostream>
#include "listT.h"

// using namespace std;

int main() {
```

Output of the Code

```
7
3
-5
25
14
Press any key to continue
```

```
Node<int> * list;

list = new Node<int>(14);
list = new Node<int>(25,list);
list = new Node<int>(-5,list);
list = new Node<int>(3,list);
list = new Node<int>(7,list);

display(list);

return 0;  }
```