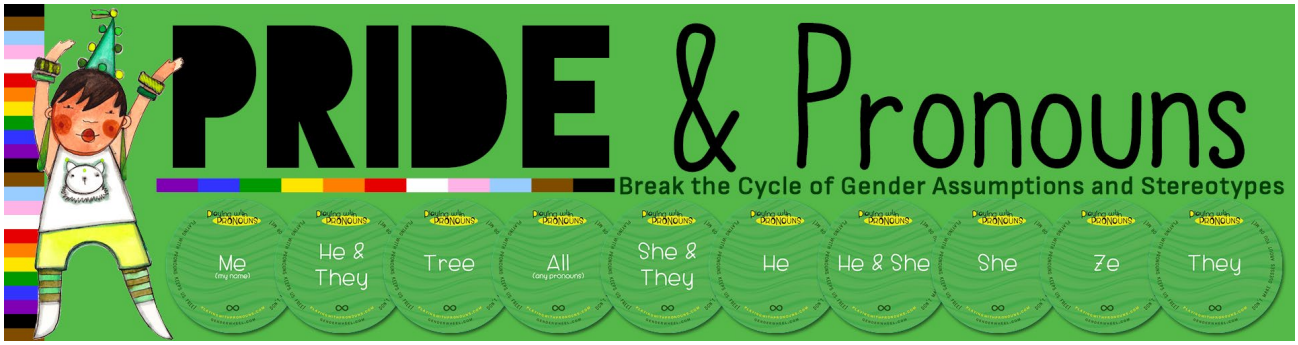
CS 202: Computer Science II
Assignment 8: "Pride is for everyone"



In this assignment you will create a templated doubly linked list that will be able to open and read a text file containing a list of pronouns (https://www.lgbtqnation.com/2022/08/incomplete-list-gender-pronouns/) and insert them alphabetically into the linked list. As usual you must not have memory leaks or errors (test it with valgrind!)

There is no skeleton code outside of just the main function, so you have flexibility on how you want to structure your code and manage variable naming convention as long as the provided main works, but let's go over the core components of the program:

node Class:

This is a templated class that will contain 3 public components, a templated variable called data, and two node pointers called next and prev. Again, you could call them differently but that is what I will refer to for the rest of the code. I highly suggest you have a constructor that initializes the pointers to null and requires the data variable via a parameter.

DoublyLinkedListClass:

This is the main class of your program, it is also a templated class, and should have two pointer variables of the node class called **head** and **tail**. It should also contain the following functions:

-Constructor to initialize your head and tail to null
-Destructor to handle de-allocation of your linked list to avoid memory leaks

void insert(T)
This function should insert the provided string by first creating a node and then inserting it into the sorted linked list. This means it must look in the existing linked list and insert it in the correct spot alphabetically. The linked list is sorted alphabetically.

bool searchDelete(T)
This function should search and remove all instances of the provided string. Meaning if there are duplicates it should delete those too. Make sure you don't break the linked list when using this function by considering all possibilities.

void print()
Prints the existing linked list in order. It should print alphabetically. See sample output for formatting

void backwardsPrint()
Prints the existing linked list in reverse. This takes advantage of the doubly linked list capabilities of your class. See sample output for the formatting.

Finally, you also need a global function called **readData** that can open a file called **pronouns.txt** and insert each string there into the linked list. Note that you should stop reading from the file when you see the following string: “Source:”. There is no science here just use >> to read in.

Here is a copy of the main function for your program:

```
int main(){
    DoublyLinkedList<string> DLL;
    DLL.insert("ge"); DLL.insert("ge"); DLL.insert("gg");
    DLL.print(); //should do ge -> ge -> gg
    DLL.searchDelete("ge"); DLL.searchDelete("gg");
    //deletes all instances so both ge are gone, also gg
    DLL.print(); //empty
    DLL.backwardsPrint(); //empty
    readData(DLL); //read in pronouns.txt
    DLL.print(); //should print pronouns sorted alphabetically a->z
    cout << endl << endl;
    DLL.backwardsPrint(); //z->a reverse alphabetical
    cout << "\nPride is for everyone." << endl; // <3
    return 0;
}
```

Refer to sample output to see what it should print.

Additionally, here is a copy of an alternate main function you can use to test some edge cases of your program. See the second sample output to see its output. This is what I used personally to make sure my code was catching all edge cases, but don’t trust me and do your own testing too!

```
int main(){
    DoublyLinkedList<int> DLL;
    DLL.insert(5);
    DLL.insert(2);
    DLL.insert(8);
    DLL.insert(7);
    DLL.insert(7);
    DLL.insert(8);
    DLL.insert(9);
    DLL.print();
    DLL.searchDelete(2);
    DLL.searchDelete(9);
    DLL.searchDelete(7);
    DLL.insert(7);
    DLL.searchDelete(7);
    DLL.searchDelete(5);
    DLL.searchDelete(8);
    DLL.insert(7);
    DLL.print();
```

```
    DLL.backwardsPrint();
    return 0;
}
```

Notes:
-Make sure your code passes all Codegrade tests if you want to receive full points.
-We may run your code with a different main so we will catch you if you just hardcode output!
-Make sure your output is EXACTLY formatted like the given sample outputs or Codegrade will eat your grade.
-Comment your source code appropriately according to the stipulations on the rubric.
-Disclaimer: I tried to find all pronouns I know they are all different type but I wanted to make the list as large as possible for the assignment, also I may have incorrect/out of date data, please don't cancel me! On the contraire I hope this assignment allows you to explore more about the wonderful world out there. Remember be Proud and love everyone!

Sample Terminal Output (running valgrind):

```
sjf@AST01$ g++ main.cpp
sjf@AST01$ valgrind ./a.out
==41415== Memcheck, a memory error detector
==41415== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==41415== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==41415== Command: ./a.out
==41415==
ge -> ge -> gg -> .
.
.
ce -> cir -> cir -> cirs -> cirself -> co -> co -> cos -> cos -> coself -> cy ->
cyr -> cyr -> cyrs -> cyrself -> eir -> eirs -> em -> emself -> ey -> he -> heir ->
heirs -> hem -> hemself -> her -> her -> hers -> herself -> hey -> him -> himself -
> hir -> hir -> hir -> hir -> hirs -> hirs -> hirself -> hirself -> his -> his ->
ne -> nem -> nemself -> nir -> nirs -> quem -> quemself -> qui -> quis -> quis ->
she -> sie -> teir -> teirs -> tem -> temself -> tey -> their -> theirs -> them ->
themself -> they -> ve -> ver -> ver -> verself -> vis -> xe -> xem -> xemself ->
xie -> xyr -> xyrs -> yo -> yo -> yos -> yos -> yoself -> ze -> zir -> zir -> zirs
-> zirself -> .


zirself -> zirs -> zir -> zir -> ze -> yoself -> yos -> yos -> yo -> yo -> xyrs ->
xyr -> xie -> xemself -> xem -> xe -> vis -> verself -> ver -> ver -> ve -> they ->
themself -> them -> theirs -> their -> tey -> temself -> tem -> teirs -> teir ->
sie -> she -> quis -> quis -> qui -> quemself -> quem -> nirs -> nir -> nemself ->
nem -> ne -> his -> his -> hirself -> hirself -> hirs -> hirs -> hir -> hir -> hir
-> hir -> himself -> him -> hey -> herself -> hers -> her -> her -> hemself -> hem
-> heirs -> heir -> he -> ey -> emself -> em -> eirs -> eir -> cyrself -> cyrs ->
cyr -> cyr -> cy -> coself -> cos -> cos -> co -> co -> cirself -> cirs -> cir ->
cir -> ce -> .

Pride is for everyone.
==41415==
==41415== HEAP SUMMARY:
==41415==     in use at exit: 0 bytes in 0 blocks
==41415==   total heap usage: 92 allocs, 92 frees, 86,616 bytes allocated
==41415==
==41415== All heap blocks were freed -- no leaks are possible
==41415==
==41415== For lists of detected and suppressed errors, rerun with: -s
```

```
==41415== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
sjf@AST01$
```

Sample Terminal Output (alternate main):

```
sjf@AST01$ ./a.out
2 -> 5 -> 7 -> 7 -> 8 -> 8 -> 9 -> .
7 -> .
7 -> .
sjf@AST01$
```