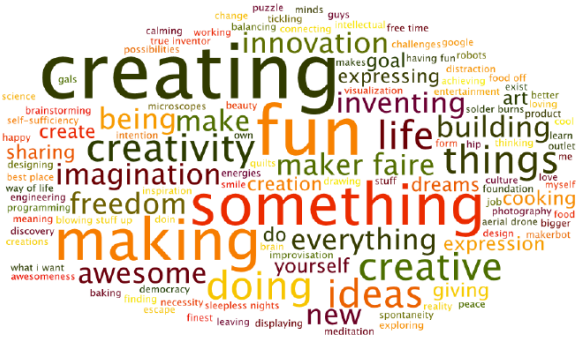

24-780 B—Engineering Computation

Assigned: Wed. Nov. 10, 2021
Due: Tues. Nov. 16, 2021, 11:59pm
(but 80% of work done on Nov. 10)

Problem Set 8: Binary Search Tree

Binary search trees (BST) are used to efficiently store data in a quickly searchable structure. In this assignment, we will create a generalized BST and then use it to perform a word count on a text document. I've also added a simple word cloud in OpenGL using the data stored in the BST.

Since the deadline is tight, start by downloading the incomplete project code attached to this assignment. Note that the C++ compiler needs to have access to the function definitions at compile time and thus it is best if the whole entire template class (declarations and definitions) is contained within the .h file. The project contains the start of the BST code and a main() file that does a lot of the work of reading a file and handling graphics.

Once downloaded, fill in the code needed for **BST::insertItem()**, **BST::printInOrder()**, **BST::findItem()**, and **BST::retrieveItemCount()**.

Note that the tree is implemented as a long vector, using the array representation of a BST, where the left child of a tree node at index p is always located at index $2*p + 1$, and the right child of the same node is located at index $2*p + 2$. If the vector you are using is not big enough to create a new tree node (i.e., vector size is less than the tree node index you need), simply resize the vector to the new index + 1.

Here is the sample text file included in the project that you may use for testing (The Gettysburg Address*[1]):

```
Four score and seven years ago our fathers brought forth on this continent, a new nation,
conceived in Liberty, and dedicated to the proposition that all men are created equal.

Now we are engaged in a great civil war, testing whether that nation, or any nation so
conceived and so dedicated, can long endure. We are met on a great battlefield of that
war. We have come to dedicate a portion of that field, as a final resting place for those
who here gave their lives that that nation might live. It is altogether fitting and proper
that we should do this.

But, in a larger sense, we cannot dedicate, we cannot consecrate, we cannot hallow, this
ground. The brave men, living and dead, who struggled here, have consecrated it, far above
our poor power to add or detract. The world will little note, nor long remember what we
say here, but it can never forget what they did here. It is for us the living, rather, to
be dedicated here to the unfinished work which they who fought here have thus far so nobly
advanced. It is rather for us to be here dedicated to the great task remaining before us,
that from these honored dead we take increased devotion to that cause for which they gave
the last full measure of devotion, that we here highly resolve that these dead shall not
have died in vain, that this nation, under God, shall have a new birth of freedom, and
that government of the people, by the people, for the people, shall not perish from the
earth.

Abraham Lincoln
```

[1] The Battle of Gettysburg was one of the deadliest in the American Civil War and is considered the turning point for the eventual victory of the United States over the Confederacy. Lincoln's short address commemorating the battlefield is held as one of the most powerful speeches in American History.

Word Cloud

I used the word frequency developed using the BST to create a ***very, very simple*** word cloud, making use of the GraphicFont library I developed (which is included in the zipped project file).

Note that the word cloud used in the title above requires that the placement of each word be perfectly located based on pixel reasoning so that the cloud can fill in tiny spaces between words. I did not accomplish this level of sophistication for this assignment. Instead, I created a word cloud using the following simple rules:

- Only words whose count is greater than 1 will be displayed (you may also choose to omit some common words like "the", "a", "and", etc.)
- Whatever size (*S)* is selected for words with a count of 2, the size is increased by a factor 1.5 for each increase (e.g., 1.5*S*, 3.0*S*, 4.5*S*, 6.0*S*, 7.5*S*, and 9.0*S* for word counts of 3, 4, 5, 6, 7, and 8)
- The location of each word is randomly selected for each word (i.e., random X, random Y).
- Set the transparency of each word to about 0.4 so that you can see words that are covered up.

Feel free to play around with the graphics so you get to understand how it all works.

Deliverables

1 zip file, containing your whole project. I advise that you include your name in comments at the top of each of the files. I expect your project will at least include the following:

wordCloud_main_andrewID.cpp << contains *main()* and some other code

BST_andrewID.h << contains your new code as well as mine

Upload the zip file to the class Canvas page before the deadline (Tuesday, Nov.16, 11:59pm), although I expect that you'll finish this assignment rather quickly so you can work on the Team Project.

Learning Objectives

Template classes in C++

Using Binary Search Trees.

Reading and writing to files.

Use OpenGL primitives along with code written by others.

Searching references (online or textbook) for C++ library functions.