**Colin Powers**

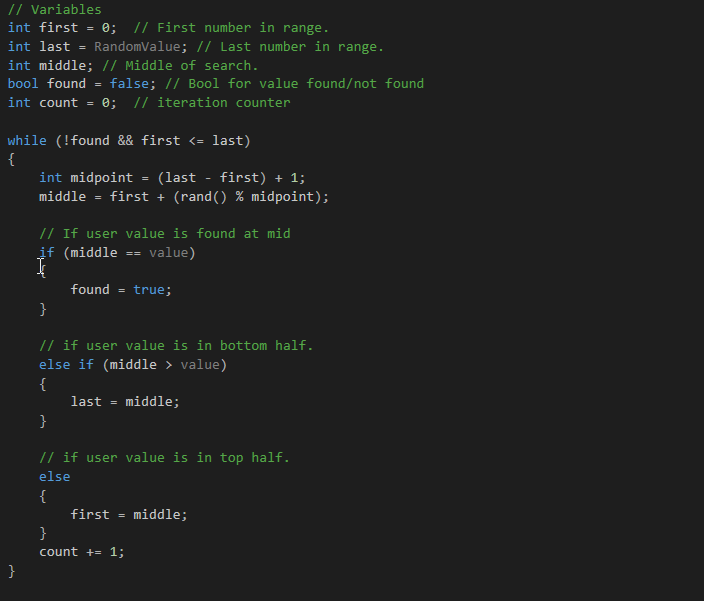
**Sherwood Report**

## Executive Summary

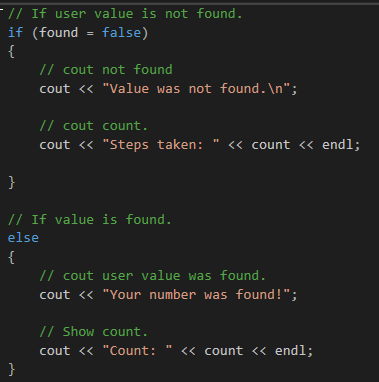
The first step I did was build a Binary search program and a Sherwood Search Program. As they are fairly simple, I built them into one program and just created a looping menu to choose which one the user wants to run. The Sherwood will randomly pick a spot to begin where as the Binary search will start straight down the middle every time.

I expect the Sherwood algorithm to move fast because it will basically discard more information that is NOT correct fast. For example, the Binary search will break it down 50/50, and one side of that 50% will be the correct side to search. Where as the Sherwood algorithm will pick something like 75/25, and possibly discard that 75%. It will break down larger chunks of information faster. The Binary search will continue to break down to 50/50 until the answer is reached which will make it run slower.   
  
I believe there is a possibility that the Sherwood algorithm could pick the wrong random start point and be slower than a binary search. This is because it could accidentally chose to discard the 25% and still have 75% of the data to search through, whereas again a Binary search will always begin at 50%. I personally believe that the Sherwood algorithm will beat out 9/10 times on the Binary because even if it does pick the wrong half, it will still have a chance to pick the better half to search through and thus speed up the process.  
  
As predicted, the sherwood algorithm usually picked a good random number to start and ran with faster times than that of the Binary search. This is because the random start point lessens the chance of a worst case scenario, in which it runs through a lot of data very slowly.

## Experiment

The Sherwood algorithm is as follows:  


The while loop first searches if the user input is found and if the first number is greater than that of the random number selected. Then it begins searching if the value is found at the middle, top, or bottom.  
“int last = RandomValue” is the integer created to create a random start point. This is created in C++ By using the CSTDLib and CTime libraries with the srand(time(0)); command. This created a random number and then this is passed on to the Sherwood algorithm function.



After it has finished searching through the data it tells you whether the number the user input was found and displays the count or steps taken to get there.

## Results

After running the test a total of 10 times, the result is that the sherwood algorithm won 4 out of ten on timing, tied twice, and lost 4. Although I am sure that if I were to conduct the experiment 100 times, I would achieve slightly different results. This goes to prove that the Sherwood algorithm’s randomness can provide a certain amount of speed maybe 75% of the time, but also because of its randomness it can be slower.

## Conclusion

In conclusion this was a fun program to work on. The random nature of the sherwood algorithm, while minimizing the negative aspects of binary search helps, sometimes you get a slightly slower program. The results were very interesting as I thought the Sherwood algorithm was going to win every time and maybe lose once or twice. I had not thought about it tieing at all with the Binary search. In the end, the Sherwood does either run equal to or faster than a binary search.

Appendix 1

