1. Some of the obstacles I faced was trying to analyze the string. The task seemed simple, yet hard to approach initially. I was trying to think of a loop that would scan through some sort of pattern of 2 letters, and then n numbers of integers, and then what follows must be specifically a symbol of either ‘+’ or ‘-‘ which brainstorming this code as a whole was daunting. The isValidUppercaseStateCode.txt was actually helpful in jumpstarting me for scanning letters. I tweaked this code to encompass the other combinations of capitalizations for the state codes. Then, I incrementally added to the loops to then pass through n numbers and scan a + or – subsequently. After having a start to creating a string to test specifically two chars of the string, I slowly cracked a solution (likewise for countCases).

bool hasValidSyntax(string orders)

if “orders” is empty, return true

Repeatedly until all chars of the string are checked:

defines a string of the two chars (starting with first two chars of string)

check if the chars are valid (i.e. letters)

check if the subsequent char is a digit (if not, return false)

while the subsequent char are digits, increment the char.

If the char following the digits is a “+” or “-“, increment the char. Otherwise, return false.

If chars aren’t valid, return false.

Return true if function hasn’t returned anything yet and cycled through flawlessly

int countCases(string orders, char status, int& caseCount)

if “orders” isn’t an order string, return 1

Repeatedly until all chars of the string are checked:

If there’s a digit that is a 0 alone, return 2;

Increment to the next char to check

If status isn’t “+” or “-“, return 3

Start “caseCount” at 0 to calculate cases only for that order string

Repeatedly until all chars of the string are checked:

Scan for consecutive digits and save it into a string.

If status is “+” and the char following the consecutive digits is “+”, add the consecutive digits to “caseCount”

If status is “-” and the char following the consecutive digits is “-”, add the consecutive digits to “caseCount”

If status symbol DOESN’T match the char following the consecutive digits is “+”, reset the string.

If countCases hasn’t returned anything yet, Return 0

bool isValidCode(string stateCode)

Initialize string containing all variations of capitalization for state codes/abbreviations.

Returns if true if “stateCode” matches with any of the variations in the initialized string above. Returns false otherwise.

1. Test Data
2. cA32-CA32+Wy45-nc3- //checks for all variations in capitalization for state codes
3. //entering nothing shouldn’t change caseCount and should be counted as a string
4. GA0000- //entering empty strings with 0 should return 2 from countCases
5. int cc; Cout << countCases(“mi09-“,’#’, cc); //should return 3 due to the status not being “+” or “–“
6. La3 //checks for absence of status in the order string which should not be valid string according to hasValidSyntax
7. Int c=-90; int b = countCases(“nV9-Va7-“,’-’, c); //make sure the initialized c doesn’t have values carried over to countCases function