Zane Koch

404984674

CS 31

Stahl

2/10/18

Report

There were a lot of obstacles for me in this project. This arose from starting it without enough proper planning, so I dug myself into a whole of rather inefficient code that took a lot of coding for each bit of progress made. After the first function I went about it in a better way which led to less obstacles by writing out on paper what each function would do and, using a flowchart, how I would write code to do the task. The most difficult thing was figuring out how to write code that would compare the listing price to the summation of bids. This was hard because there are not man easy ways of writing a program which identifies which numbers in the string correspond to which type of value. I eventually solved this problem by iterating over the string looking for ‘B’s because the numbers should always follow a B. The +’s also posed a problem that I overcame by knowing that the howMuch function would only be called when a string was valid, so it could be assumed that after the first B there would be no more +’s.

The general form of my program is as follows. The isValid function is built using many switch and if statements which are iterated over by for loops. These loops compare the number and sequence of the characters in the string. First it is checked if any bad characters are in the string and if not, it makes sure all characters have characters before and after them that are logical and allowed. Next the listingSold function is built out of two int functions: listingPrice and bidSummation. ListingPrice returns what the listing price was by looking at the number after the first L using a for loop and switch statement. bidSummation sums up all the bids by looking at the numbers following each B through a large for loop and a bool statement that tells wether or not it is the first B (this matters because only the first B has a +). The listingSold function then compares the output of these two functions and if the bidSummation is greater than the listing price it returns true and otherwise false. The howMuch function depends on the previous 3 functions. If listingSold is true it then returns the number given by the bidSummation function, otherwise it returns either 0 or -1 depending on if the string is valid. Lastly, the watchers function is very simple, it iterates over the whole string looping for any w’s or u’s and either subtracts or adds to the number of watchers depending on what it encounters.

**Tet Data:**

**Test 1:** “L50”

Good test because sometimes a string might be processed incorrectly because the program depends on all expected characters being there.

**Test 2:** “L100B+50WWUB50”

Good test because it uses all types of possible characters and also test if the program handles correctly the case where the bids are equal to the listing price

**Test 3:** “2hqkj fdehfq0”

This good be any sort of gibberish string, the point is that the program must be able to identify invalid strings.

**Test 4:** “l100b+50”

This is a good test because it makes sure the program treats uppercase and lowercase letters in the same way.

**Test 5:** “L100WUwB+1000”

This is a good test because it test both if the proram thinks the string is valid with only one bid and if it keeps track of watches and un-watches appropriately.

**Test 6:** “L50L60”

This is a good test because it makes sure the program knows an item can only be listed once and the price cannot be changed.

**Test 7:** “B+50L50”

This is a good test because while it contains all valid characters it is in fact invalid because the string does not start with a listing.

**Test 8:** “L50UWU”

This tests if the program knows watching must happen before unwatching and if the number of unwatchers can exceed the number of watchers.

**Test 9:** “L50.5B+10”

This test if the program knows that floating point values cannot be used.

**Test 10:** “L100B+50BW50”

Watching cannot happen mid bid.