1. An obstacle I first faced was when I began to write my code was figuring out how to separate the order string so that I could test separate conditions, like checking if the state code was valid. By building my program incrementally, I was able to overcome this for one state order in the string but encountered the same problem when trying to expand it to have multiple state orders. When trying to incorporate multiple state orders, it was hard to keep one state order’s from replacing the other; for example, one state order would be invalid but the next one would be valid, and the function as a whole would return true, which was incorrect.
2. Valid syntax function

Check if empty string

Loop through order string

Separate state code

Track position after state code

Check state code with given function

Check numbers after state code

Track position after number of cases

Check status

Track position after status

If all tests pass

Return true

Return false

Count cases function

Check if order string is valid

Check if status is valid

Check if empty order string

Loop through orders

Separate state code

Separate case number

Add to empty string

Convert string to int

Check if case number is zero

Check if status matches

If it does, add to total number of cases

Reset string for case number to empty for next state order

Return 0

1. Valid syntax function
   1. (“TX38-“):

to test first increment of function, single case

* 1. (“”):

to test empty string

* 1. (“TX38-CA132+”):

to test multiple state orders

* 1. ("MX38-CA132+"):

to catch incorrect state code

* 1. (“CA132+MX38-“):

to make sure the first valid case is overwritten by the invalid one

* 1. (“TX38CA132+”):

to test lack of status

* 1. (“tx38-“):

to test uppercase function

* 1. (“TX-CA132+”):

to test lack of numbers after state code

* 1. (“132+”)

to test lack of state code

* 1. (“CA00+”)

to test zero cases

Count cases function

* 1. (“TX38-“, ‘-‘, -999):

to test single order and check if case number is replaced

* 1. (“TX38-“, ‘&‘, -999):

to test return for invalid status and that case number is not replaced

* 1. (“CA-“, ‘-‘, -999):

to test return for invalid syntax and that case number is not replaced

* 1. ("TX38-CA132+Ms6-nY290-UT006+MS8+CA15+", '+', cases):

to test multiple case and addition of the right cases

* 1. (“Ms6+ CA000+nY290- “, ‘+‘, -999):

to test return for 0 cases and that case number is not replaced

* 1. (“”, ‘-‘,-999):

to test return for empty string and replacement of case number