StringCalculator

This tool is designed to take any string as an input, extract numbers and return their sum. It ignores any letters and symbols, ignores any numbers greater than 1000 and return an exception if negatives were provided.

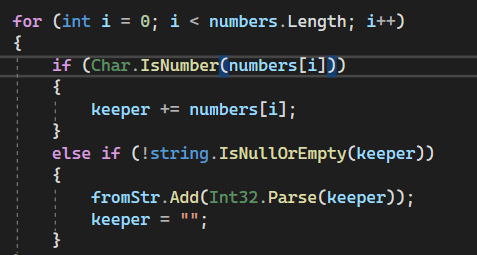
Breakdown:

Main:

It meets user with the message to enter a string. User can enter anything they wish. On typing “exit” the program will finish executing.

Add:

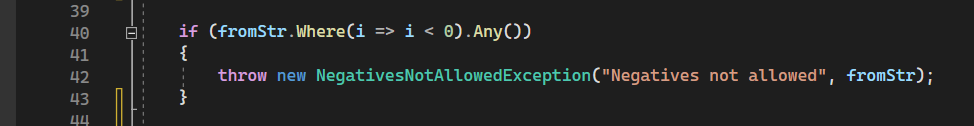
I’ve started with the simple check to return 0 if string is null or empty by using string.IsNullOrEmpty(numbers). I’ve then created the “for” loop to go through all symbols. As I wanted to cover numbers greater than 9, I’ve created the string “keeper” to store numbers until symbol is encountered. In case it happens, “keeper” will be converted into the number, stored in the list and “keeper” dropped to “” to pick new numbers:



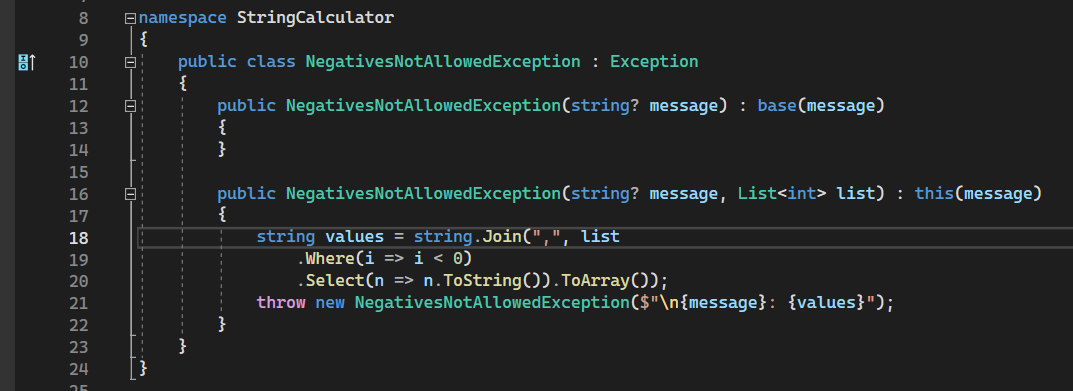
This method allowed me to ignore all signs, new liners and delimiters. At this stage, however, the calculator was ignoring “-“ as well, meaning that negative numbers would become positives, and it is supposed to create an exception. I’ve resolved it by adding OR to the for loop ((numbers[i].Equals('-') && Char.IsNumber(numbers[i+1]))). It would add negatives to the list, but if dash was put as a last symbol in the “numbers” (eg: “-1,2,-3-) the application would crash due to IndexOutOfRangeException. To fix this I’ve decided to add an extra symbol to the “numbers”:



As for negative numbers I’ve created an Exception, which would be called after List is populated with all numbers:



The exception would get all values and with the help of Linq extract negative values and convert them to string and would end up throwing an exception:



For ignoring numbers greater than 1000, I’ve used method in Lists Removing matching numbers and put it before the check for negative numbers (to exclude them from the exception (in case input will contain negative 1000s):



During the implementation, I’ve been writing up Unit tests and wrote 14 to cover as many cases as possible.