Roman Numeral Conversion

Logic Step	Time taken to Understand	Time taken to implement
Roll around the floor in pain trying to understand why Cobol does the things Cobol does	∞	∞
Break down code into function files	30 min	20 min
Learn to hate spaghetti code	∞	∞
Learn basic Cobol syntax/ comments	30 min	Entire Project
Reinvent the structure of the parser to not use GOTO's	N/A	30 min
Modify program to mirror the functionality of the given code	N/A	15 min
Looping program to accept new input	5 min	15 min
Reading in Files	60 min	120 min
Learning how to use functions in Cobol	20 min	30 min
Total:	~3 Hours	~4 Hours

Would it have been easier to re-write the program from scratch in a language such as C?

It would’ve been much easier to write this program in C, it would've been much easier to implement functions to make the code less redundant and shorter. There wasn't anything overly difficult about how the program needed to be designed, other than there is very little information about problems you are facing. I think the most difficult part of the entire project was implementing dynamic file name reading, and after I found a question based around the same concept it was fairly easy to implement in my code.

Is your program shorter or longer? Why?

My program is longer than the original code, I decided to write a new parsing function instead of reimplementing the original code. I chose to go with an IF-ELSE block of statements, which added to the projects length. I cut down on code by putting the entire parser and output in a function file, so I could call it easier when reading a file as well.

Is there a better way of writing the program?

The better way of writing the program would have definitely been to create a function for subtracting the previous values for the sum. I had originally planned it this way, but after not getting it to work I focused on manually adding clauses to the addition of numbers. It increased the length of the program slightly, but I feel it doesn't cause much slow down. Although, I would like to improve on that if I had to write it in a different language again.

What were the greatest problems faced during the re-engineering process?

The greatest problem I had re-engineering the code was the input file. That being said, it wasn’t very hard after I found a few examples of it online. It required a lot of refactoring of my code to make it work cleanly and efficiently but in the long run having to do that improved the overall readability of my program. There weren't a lot of other difficult problems that couldn't be solved by simply searching for what I thought the C equivalent should be. Sorting through the mainframe forums vs the new Cobol standard was a bit tricky though.

Is there anything you'd like to change in your program?

I would’ve liked to incorporate more functions into my program, but I didn't feel like adding more files for a relatively simple small amount of code. I feel like adding a 60 line additional file would've made it more complex than the added 60 lines in my other function to accomplish the same thing.

I would have also liked to experiment more with how string lengths and printing them to the terminal could have been effected more. From my design it limited how long the roman numeral could be, but I feel at 30 characters there aren't many combinations that could break my program. Plus, theres an added benefit of having everything line up on output because the strings are all padded with blank spaces after the characters