HTML2LaTeX Assignment

For this assignment, the goal was to create a a program that would convert a subset of HTML code into the equivalent LaTeX code using the Lex lexical analyzer generator. Breaking down the HTML tags into smaller tokens allowed for the conversion of these elements, along with the text and potential nested elements within. The program itself requires support for most common HTML tags, including comments, headings, pre-formatted paragraphs, regular paragraphs, ordered and unordered lists, and list items. Additionally, the program was required to support the nesting of other tags within list items and paragraphs, such as small, large, bold, italic, and many others.

My approach involved creating multiple rules for the different HTML elements. These would match the corresponding HTML tag and use regular expressions to match either text or other elements nested within. For the more simpler tags, such as headings and pre-formatted paragraphs, I simply matched the beginning and ending HTML tag, along with all the text within. Once matched, I would then strip off the ending HTML tag and print out the LaTex equivalent, containing the text that was found. In addition to printing this text, I would also ensure to print everything following the beginning of the HTML tag to ensure that all that was being inserted into the LaTeX element was the text. For the nested elements within paragraphs, I defined rules for each that would effectively do the same thing as previously described. The difference was that, when encountering a paragraph, I would use the `<PARA>` definition to branch off of. Then, following this `<PARA>` definition would be the nested elements. This would help ensure that these elements were actually inside of the paragraph and convert them as such. In addition, a similar approach was followed for the list elements within ordered and unordered lists. The primary challenge here was determining which kind of list this list element lived in. Since each can contain multiple list elements, you would need to return to the parent list on the completion of a list element. To do this, I used the `list_type` variable definition to track this. For example, when an ordered list element was found, and it contained a list item, I would set this variable to 1. Then, within these list items, after matching the nested elements and fully creating the

LaTeX equivalent, I would then check which list it belonged to through this variable. If it was 1, I would simply `BEGIN OL` to jump back to the parent element and continue. This allowed for all of the list items to be properly placed within their respective parent lists and be converted correctly to LaTeX.

While writing my solution for this assignment, I made sure to rigorously test each individual tag conversion to ensure the correct functionality. For example, when I wrote the nested elements for paragraphs, I made sure to test each individual one as I went by creating various HTML files. I would also test the functionality on another HTML file that contained all of the nested elements within a paragraph to ensure they all worked seamlessly with one another. In addition to this, when testing, I would also run the solution file on the same HTML files and view the differences between both my produced LaTeX output and the expected. I would also run the `diff` command to truly see what exactly was different between my output and the expected output, whether that be whitespace differences or actual errors. Running the solution program like this helped ensure that I was, in fact, doing the assignment correctly which I appreciated. Another debugging and testing technique I used was simply printing. As I implemented the different tags, I wanted to ensure that I was stripping off the correct piece of text and not accidentally deleting part of the text that I wanted to keep within the LaTeX element. This proved to be immensely helpful in simplifying the assignment and breaking down the steps needed in each conversion.

Overall, I really enjoyed implementing this assignment. The only issue I really ran into while completing this was finding the whitespace differences between my output and the expected LaTeX output. While this wasn't a huge issue to solve, it was slightly confusing at first and did require some meticulous newline placement within my Lex code. Besides that, I didn't have any other issues and I enjoyed getting to learn and work with Lex!