1. A file named **report.docx** (in Microsoft Word format), or **report.txt** (an ordinary text file) that contains:
   1. A brief description of notable obstacles you overcame.
   2. A description of the design of your project. Use [pseudocode](https://web.cs.ucla.edu/classes/fall23/cs31/pseudocode.html) where it clarifies the presentation. Someone reading your description should be able to determine what the responsibilities of the functions you wrote are. If someone had to modify your code later, could they from your description readily find in your program the approximate location of the code that, for example, handles a word portion not fitting on a line?
   3. A list of the test data that could be used to thoroughly test your function, along with the reason for each test case (e.g. "word broken at hyphen" or "two spaces after period"). You must note which test cases your program does not handle correctly. (This could happen if you didn't have time to write a complete solution, or if you ran out of time while still debugging a supposedly complete solution.)
2. Some obstacles I overcame were overcomplicating the code and then struggling to debug. I ended up having to restart because I tried to do the whole thing at once instead of doing it step-by-step. In the future, I will break the project down into several parts. Additionally, I also struggled with figuring out where and when to put spaces and paragraph/line breaks.
3. Pseudocode
   1. Get word function
      1. Loop until no leading spaces
         1. Increment position
      2. Repeat until space or end:
         1. Get word
         2. Increment position
         3. If last char in word is hyphen
            1. set return value to 1
            2. Break
      3. Return 1 if end in hyphen, 0 if not
   2. Skip paragraph function
   3. Check line function
      1. Repeat until end of line
         1. Call get word function
         2. break if end of line
         3. Check if paragraph
            1. Call skip paragraph function
            2. Make new paragraph
            3. Set character count to zero
            4. Continue loop
         4. Check how many spaces to add
            1. Add 1 to spaces
            2. Add 1 more if last character of previous word is punctuation
         5. If word + charCount + spaces <= lineLength
            1. Output spaces
            2. Output word
            3. Adjust character count
         6. Else
            1. Check if first word

If not, new line

* + - * 1. if word less than lineLength

Output word

Adjust character count

* + - * 1. Else

Repeatedly

Repeatedly

Output each character in word

Break if ‘\0’

Increment position

If ‘\0’

Set character count

Else

Output new line

* + - 1. Set last character, previous check, and first no longer true
  1. Render function
     1. Set result to 0, outputNothing to false
     2. If lineLength invalid
        1. Return 2
     3. Repeatedly get line from input
        1. If first line
           1. If skip paragraphs returns false

Break

* + - * 1. Change first to false
      1. Check the line
      2. If the check line function returns 1, set result to 1
    1. If first is false
       1. Output new line
    2. Return result

1. Test cases
   1. lineLength = -1 (lineLength is invalid, should return 2)
   2. lineLength = 10, text.txt = “ blah” (leading spaces)
   3. lineLength = 10, text.txt = “blah ” (trailing spaces)
   4. lineLength = 10, text.txt = “@P@ blah” (leading paragraph break)
   5. lineLength = 10, text.txt = “blah @P@” (trailing paragraph break)
   6. lineLength = 10, text.txt = “blah @P@ @P@ blah blah blah” (multiple paragraph breaks in a row)
   7. lineLength = 10, text.txt = “blah \n blah blah blah” (line break)
   8. lineLength = 10, text.txt = “blah \n \n blah blah blah” (multiple line breaks)
   9. lineLength = 10, text.txt = “blah. \n blah blah blah” (line break after punctuation)
   10. lineLength = 10, text.txt = “blah. \n \n blah blah blah” (multiple line breaks after punctuation)
   11. lineLength = 10, text.txt = “blahblahblahblah blah” (word longer than lineLength)
   12. lineLength = 10, text.txt = “blah blahblahblahblah blah” (word longer than lineLength, not in first position)
   13. lineLength = 10, text.txt = blah. blah blah blah” (with a period, not at the end of output line)
   14. lineLength = 10, text.txt = blah: blah blah blah” (with a colon, not at the end of output line)
   15. lineLength = 10, text.txt = blah? blah blah blah” (with a question mark, not at the end of output line)
   16. lineLength = 10, text.txt = blah! blah blah blah” (with an exclamation, not at the end of output line)
   17. lineLength = 10, text.txt = “blah blah. blah” (with a period, at the end of output line)
   18. lineLength = 10, text.txt = “blah blah: blah” (with a colon, at the end of output line)
   19. lineLength = 10, text.txt = “blah blah? blah” (with a question mark, at the end of output line)
   20. lineLength = 10, text.txt = “blah blah! blah” (with an exclamation, at the end of output line)
   21. lineLength = 251, text.txt = “blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah blahblahblah” (lineLength > 250)
   22. lineLength = 10, text.txt = “blah-blah blah blah” (hyphen not split)
   23. lineLength = 10, text.txt = “blah blah-blah blah” (hyphen split)
   24. lineLength = 10, text.txt = “blah blah blah blah” (multiple spaces between words)
   25. lineLength = 10, text.txt = “blah@P@ blah” (word with @P@ that is not by itself)
   26. lineLength = 10, text.txt = “blah blah!blah--blah” (two hyphens in a row)
   27. lineLength = 10, text.txt = “bl!ah” (punctuation in the middle of a word)
   28. lineLength = 10, text.txt = “@P@ @P@“ (when @P@ is the only word in input)
   29. lineLength = 10, text.txt = “ “ (when only spaces)
   30. lineLength = 10, text.txt = “” (empty text)
   31. lineLength = 8, " This is a test \n" (spaces right before the \n at the end)
   32. lineLength = “blah blah blah \n blah blah blah” (something with a new line)
   33. lineLength = “\n \n \n” (only new lines)