Question: https://leetcode.com/problems/text-justification/

I feel this is a simulation based problem the main logic lies in understanding different justifications explained in the problem statement and implementing them.

Code:  
class Solution {

public List<String> fullJustify(String[] words, int maxWidth) {

int left = 0; List<String> result = new ArrayList<>();

while (left < words.length) {

int right = findRight(left, words, maxWidth);

result.add(justify(left, right, words, maxWidth));

left = right + 1;

}

return result;

}

private int findRight(int left, String[] words, int maxWidth) {

int right = left;

int sum = words[right++].length();

while (right < words.length && (sum + 1 + words[right].length()) <= maxWidth)

sum += 1 + words[right++].length();

return right - 1;

}

private String justify(int left, int right, String[] words, int maxWidth) {

if (right - left == 0) return padResult(words[left], maxWidth);

boolean isLastLine = right == words.length - 1;

int numSpaces = right - left;

int totalSpace = maxWidth - wordsLength(left, right, words);

String space = isLastLine ? " " : blank(totalSpace / numSpaces);

int remainder = isLastLine ? 0 : totalSpace % numSpaces;

StringBuilder result = new StringBuilder();

for (int i = left; i <= right; i++)

result.append(words[i])

.append(space)

.append(remainder-- > 0 ? " " : "");

return padResult(result.toString().trim(), maxWidth);

}

private int wordsLength(int left, int right, String[] words) {

int wordsLength = 0;

for (int i = left; i <= right; i++) wordsLength += words[i].length();

return wordsLength;

}

private String padResult(String result, int maxWidth) {

return result + blank(maxWidth - result.length());

}

private String blank(int length) {

return new String(new char[length]).replace('\0', ' ');

}

}

Github Link :<https://lnkd.in/ecwtJeaz>