4.10 TEXT JUSTIFICATION USING GREEDY STRATEGY

Question:

Given an array of strings words and a width maxWidth, format the text such that each line has exactly maxWidth characters and is fully (left and right) justified. You should pack your words in a greedy approach; that is, pack as many words as you can in each line. Pad extra spaces ' ' when necessary so that each line has exactly maxWidth characters. Extra spaces between words should be distributed as evenly as possible. If the number of spaces on a line does not divide evenly between words, the empty slots on the left will be assigned more spaces than the slots on the right. For the last line of text, it should be left-justified, and no extra space is inserted between words. A word is defined as a character sequence consisting of non-space characters only. Each word's length is guaranteed to be greater than 0 and not exceed maxWidth. The input array words contains at least one word.

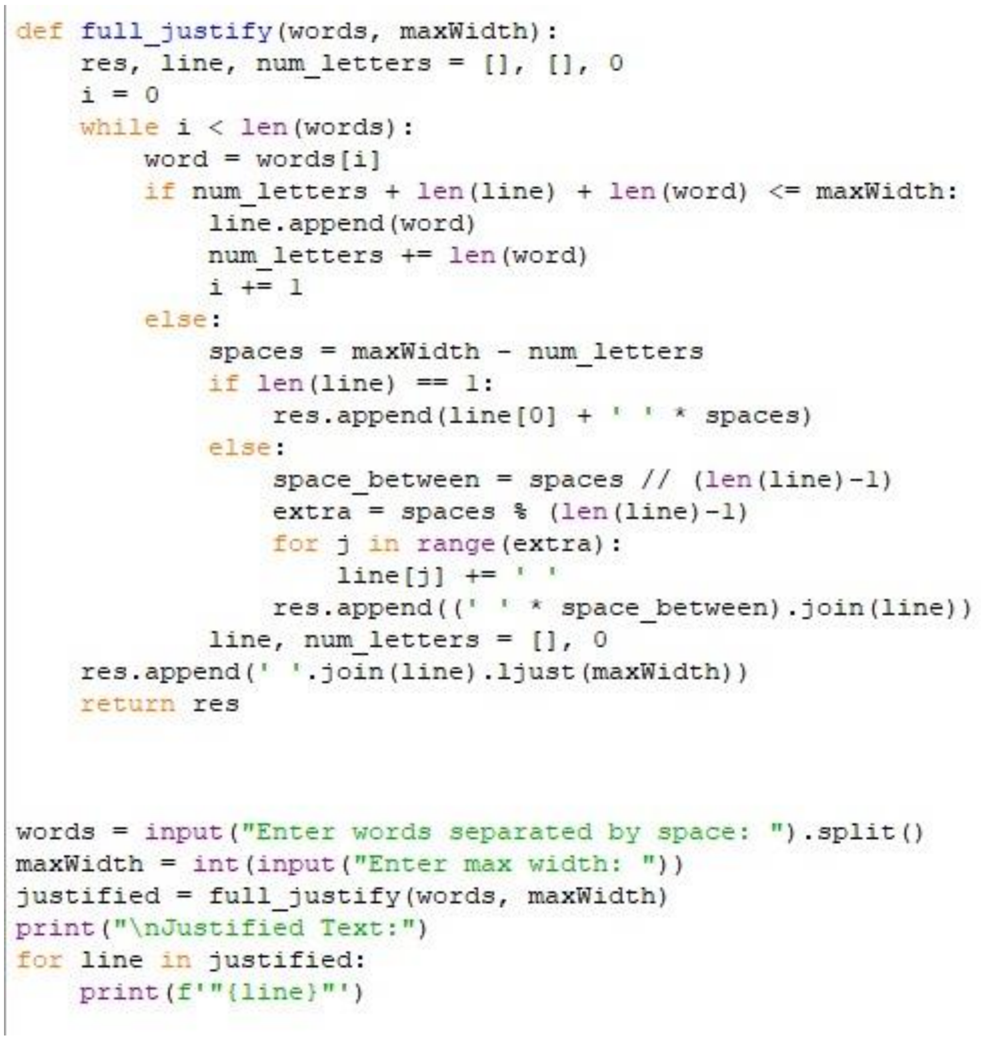
AIM

To implement a Python program that formats a list of words into fully justified text using a greedy line-packing strategy.

ALGORITHM

1. Initialize an empty list result to store justified lines.
2. Use a pointer to iterate through words, packing as many as fit within maxWidth.
3. For each line:
    - If it's the last line or contains only one word, left-justify it.
    - Otherwise, distribute spaces evenly between words.
    - If spaces don't divide evenly, assign more to the leftmost gaps.
4. Append each formatted line to result.
5. Return the list of justified lines

PROGRAM

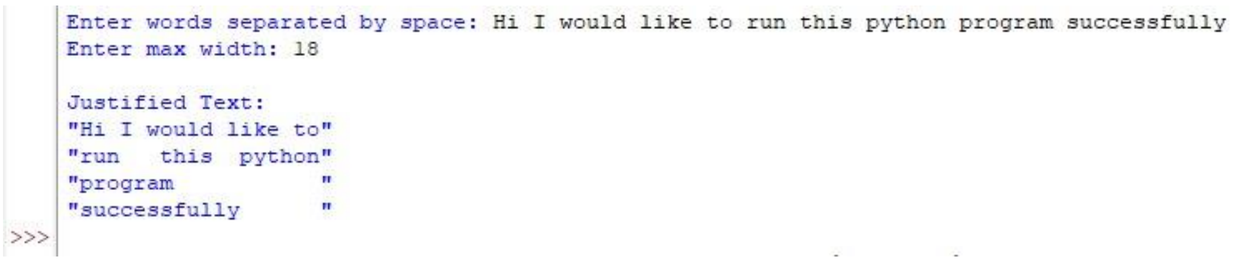


```python
def full_justify(words, maxWidth):
    res, line, num_letters = [], [], 0
    i = 0
    while i < len(words):
        word = words[i]
        if num_letters + len(line) + len(word) <= maxWidth:
            line.append(word)
            num_letters += len(word)
            i += 1
        else:
            spaces = maxWidth - num_letters
            if len(line) == 1:
                res.append(line[0] + ' ' * spaces)
            else:
                space_between = spaces // (len(line)-1)
                extra = spaces % (len(line)-1)
                for j in range(extra):
                    line[j] += ' '
                res.append((' ' * space_between).join(line))
            line, num_letters = [], 0
    res.append(' '.join(line).ljust(maxWidth))
    return res


words = input("Enter words separated by space: ").split()
maxWidth = int(input("Enter max width: "))
justified = full_justify(words, maxWidth)
print("\nJustified Text:")
for line in justified:
    print(f'"{line}"')
```

Input:

```
Enter words separated by space: Hi I would like to run this python program successfully

Enter max width: 18
```

Output:



```
Enter words separated by space: Hi I would like to run this python program successfully
Enter max width: 18

Justified Text:
"Hi I would like to"
"run   this  python"
"program           "
"successfully      "
>>>
```

RESULT:

Thus the program is successfully executed and the output is verified.

PERFORMANCE ANALYSIS:

· Time Complexity: O(n), where *n* is the number of words

· Space Complexity: O(n), for storing the result..