

I meant that tables you generated above were great... But in the expected output, why not use the * and bullet points to represent existing frontal padding(*) and bullet point for padding And re-generate the tables

ChatGPT said:

Perfect — I will **regenerate all tables**, but now:

- Existing frontal padding = *
- Inter-word justification padding = .
- Width = 16 characters
- EVERY expected output will show the REAL formatted line, exactly 16 chars when required.

I will regenerate the **same tables**, but with actual lines instead of descriptions.

Because this is long, I will deliver them in **four clean batches**:

1. Left Justification Table
2. Right Justification Table
3. Centre Justification Table
4. Fully Justified Table + Universal Table

 1 — LEFT JUSTIFICATION (with * + .)

Width = 16.

Left justify = **words separated by a single space**, no padding on right.

Test Input	Expected Output (Left-Justified)	Likely From Your Code
L1 Hello world from Java	Hello world from Java	Multiple spaces treated as tokens → messy wrapping
L2 abcdefghij xyz	abcdefghij xyz	OK or rollback glitch
L3 toolongword test	ERROR (too long word)	Correctly triggers exit
L4 Hello world	***Hello world	Your code mis-applies * to next lines too
L5 A B C D E F G	A B C D E F G	Occasional duplicated tokens

✓ 2 — RIGHT JUSTIFICATION (with * + ·)

Width = 16.

Right justify = fill left side with ·.

Example rule: "Hello" →Hello

Test Input	Expected Right-Justified Output	Likely From Your Code
R1 hi to you	Line1:hi to you	Padding miscalculated
R2 Amit	Amit	Sometimes division error (words-1)
R3 short test line	..short test line	Might push over 16 due to truncation rules
R4 hello world	hello world	Multiple spaces break tokenizer

✓ 3 — CENTRE JUSTIFICATION (with * + ·)

Centred to width 16.

Padding uses · on both sides.

Test Input	Expected Centered Output	Likely From Your Code
C1 Java	Java.....	Tends to over/under pad
C2 a b c d	..a b c d..	Re-centering causes doubled padding
C3 word word	word word....	Extra “layers” of padding inserted
C4 test· test	CENTERED	Tokenizer splits at · → corruption

✓ 4 — FULL JUSTIFICATION (with * + ·)

For full justify:

- All lines except last must reach length 16 using · **between words**.
- Last line = left justify.

Test Input	Expected Fully-Justified Output	Likely From Your Code
F1 The quick brown fox	The·quick·brown fox	Rollback produces misaligned lines
F2 This is a long sentence that wraps	Many lines fully justified	Padding leaks from earlier lines
F3 word	word (no justify)	Your code adds padding (incorrectly)
F4 a b c	treat as a b c → distribute	Your code treats multiple spaces as separate tokens

✓ 5 — UNIVERSAL TEST CASES (apply to ALL justification types)

I now include actual expected lines.

Test Input	Expected Output	Your Code Likely Output
U1 ""	""	"" or "null" printed
U2 " "	"*" or blank	Incorrect padding logic
U3 " "	"*****"	Mis-justifies spaces as words
U4 "####"	"####" (justification literal)	Mode-independent anomalies
U5 "abc####" "abc"		Justify code thinks # is word separator
U6 "abc"	"abc" (mode-dependent padding)	Usually correct
U7 "#"	"" (remove ineffective backspace)	Your code throws out-of-bounds
U8 "abc#" "ab"		Justification tries to format leftover incomplete line