I have provided sufficient explanation in code. I will refrain from elaborating on the screen messages…

TEST SCENARIO 1 (OLD CODE)

```
//THESE ARE ALL TEST CASES......SELECT ONE ONLY!
string text = "The quick brown fox jumps over the lazy dog";
//String text="";
//String text = " This is a test but making it a bit longer!";
//String text = "My";
//String text = "My name is Amit Amlani. This is a sligthly longer test to see if the text can be spanr
//String text = "This will be 16.";
//String text = "Thiswillbetesting  16 testing."
```

*****OUTPUT*****

```
Welcome to Online IDE!! Happy Coding :)
Your line getting bigger:The(3 chars inc white space)
Words after truncation: 1
Your line getting bigger:The·quick(9 chars inc white space)
Words after truncation: 2
Your line getting bigger:The·quick·brown(15 chars inc white space)
Words after truncation: 3
Your line getting bigger:The·quick·brown·fox(19 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):The·quick·brown
Length rolled back line: 15
This word getting truncated:fox
The current buffer is: 1
Number words to accomodate for: 3(The·quick·brown)
Last word in tokenizer should be same as truncated word: fox
What is wordcount here: 4
What is truncated word count: 3(The·quick·brown)  15 chars
The current buffer is: 1
The·quick·brown   =>qualifies for 1 padding at front since it has:3 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·The·quick·brown
This word will be carried over to next line:fox

Your line getting bigger:fox·jumps(9 chars inc white space)
Words after truncation: 2
Your line getting bigger:fox·jumps·over(14 chars inc white space)
Words after truncation: 3
Your line getting bigger:fox·jumps·over·the(18 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):fox·jumps·over
Length rolled back line: 14
This word getting truncated:the
The current buffer is: 2
Number words to accomodate for: 3(fox·jumps·over)
Last word in tokenizer should be same as truncated word: the
What is wordcount here: 7
What is truncated word count: 3(fox·jumps·over)  14 chars
The current buffer is: 2
CURRENT LENGTH of line: 14
```

Qualified for extra padding
1 extra padding between:fox·jumps·over
It will now process truncated string with extra 1 padding between the words:fox·jumps·over
Your line getting bigger:fox(3 chars inc white space)
Your line getting bigger:fox··jumps(10 chars inc white space)
Your line getting bigger:fox··jumps··over(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:fox··jumps··over
This word will be carried over to next line:the

Your line getting bigger:the·lazy(8 chars inc white space)
Words after truncation: 2
Your line getting bigger:the·lazy·dog(12 chars inc white space)
Words after truncation: 3
Finally completed the last line:the·lazy·dog
Left over StringJoiner: 3 word(s)=> the·lazy·dog (12 chars inc white space)
Total running words: 9
Buffer is: 4
It will now process the string with extra: 2 padding between words:the·lazy·dog
your line getting bigger:the(3 chars inc white space will NOT exceed 16)
your line getting bigger:the···lazy(10 chars inc white space will NOT exceed 16)
your line getting bigger:the···lazy···dog(16 chars inc white space will NOT exceed 16)
completed line:the···lazy···dog
NEW LENGTH of line:16

****THIS WILL PRINT ENTIRE TEXT**********

•The·quick·brown

fox••jumps••over

the•••lazy•••dog

****

** Process exited - Return Code: 0 **

********EXPECTED OUTCOME********

```
["the  quick brown", # 1 extra space on the left
"fox  jumps  over", # 2 extra spaces distributed evenly
"the   lazy   dog"] # 4 extra spaces distributed evenly
```

N.B There is considered to be an error in the question given…
Placing 4 extra spaces between the words will give 18 characters including white spaces….

the••••lazy••••dog

TEST SCENARIO 2 AN EMPTY STRING (OLD CODE)

```
//THESE ARE ALL TEST CASES......SELECT ONE ONLY!
//String text = "The quick brown fox jumps over the lazy dog";
String text="";
//String text = " This is a test but making it a bit longer!";
//String text = "My";
//String text = "My name is Amit Amlani. This is a sligthly longer test to see if the text can be spanr
//String text = "This will be 16.";
//String text = "Thiswillbetesting  16 testing."
```

*****OUTPUT*****

Welcome to Online IDE!! Happy Coding :)
Finally completed the last line:
Left over StringJoiner: 0 word(s)=> (0 chars inc white space)
Total running words: 0
Buffer is: 16
completed line:
NEW LENGTH of line:0

****THIS WILL PRINT ENTIRE TEXT**********

****

TEST SCENARIO 3 – A STRING WITH PADDING ALREADY AT FRONT. UP TO THE COMMA IT IS EXACTLY LENGTH OF K=16. IT SHOULD FIT EXACT UP TO HERE… (OLD CODE)

```
//THESE ARE ALL TEST CASES......SELECT ONE ONLY!
//String text = "The quick brown fox jumps over the lazy dog";
//String text="";
String text = "This is a test, but making it a bit longer!";
//String text = "My";
//String text = "My name is Amit Amlani. This is a sligthly longer test to see if the text can be span
//String text = "This will be 16.";
//String text = "Thiswillbetesting  16 testing."
```

*****OUTPUT*****

Welcome to Online IDE!! Happy Coding :)
Your line getting bigger:This(4 chars inc white space)
Words after truncation: 1
Your line getting bigger:This·is(7 chars inc white space)
Words after truncation: 2
Your line getting bigger:This·is·a(9 chars inc white space)
Words after truncation: 3
Your line getting bigger:This·is·a·test,(15 chars inc white space)
Words after truncation: 4
Your line getting bigger:This·is·a·test,·but(19 chars inc white space)
Words after truncation: 5

Rolled back to(due to exceeding 16):This·is·a·test,
Length rolled back line: 15
This word getting truncated:but
The current buffer is: 1
Number words to accomodate for: 4(This·is·a·test,)
Last word in tokenizer should be same as truncated word: but
What is wordcount here: 5
What is truncated word count: 4(This·is·a·test,) 15 chars
The current buffer is: 1
This·is·a·test,   =>qualifies for 1 padding at front since it has:4 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·This·is·a·test,
This word will be carried over to next line:but

Your line getting bigger:but·making(10 chars inc white space)
Words after truncation: 2
Your line getting bigger:but·making·it(13 chars inc white space)
Words after truncation: 3
Your line getting bigger:but·making·it·a(15 chars inc white space)
Words after truncation: 4
Your line getting bigger:but·making·it·a·bit(19 chars inc white space)
Words after truncation: 5
Rolled back to(due to exceeding 16):but·making·it·a
Length rolled back line: 15
This word getting truncated:bit
The current buffer is: 1
Number words to accomodate for: 4(but·making·it·a)
Last word in tokenizer should be same as truncated word: bit
What is wordcount here: 9
What is truncated word count: 4(but·making·it·a) 15 chars
The current buffer is: 1
but·making·it·a   =>qualifies for 1 padding at front since it has:4 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·but·making·it·a
This word will be carried over to next line:bit

Your line getting bigger:bit·longer!(11 chars inc white space)
Words after truncation: 2
Finally completed the last line:bit·longer!
Left over StringJoiner: 2 word(s)=>   bit·longer! (11 chars inc white space)
Total running words: 10
Buffer is: 5
It will now process the string with extra: 5 padding between words:bit·longer!
your line getting bigger:bit(3 chars inc white space will NOT exceed 16)
your line getting bigger:bit·····longer!(16 chars inc white space will NOT exceed 16)
completed line:bit·····longer!
NEW LENGTH of line:16

****THIS WILL PRINT ENTIRE TEXT**********

•This•is•a•test, it still maintains padding on left of first line as expected.

•but•making•it•a it has now incorporated frontal padding since longer than 1 word…

bit••••••longer! 3 characters + 6 padding + 7 characters = 16

****

** Process exited - Return Code: 0 **

TEST SCENARIO 4: 1 WORD
I HAD TO TWEAK THE CODE HERE, BUT IT INVOLVED AN EXTRA IF LOOP, THIS WAS FORTUNATELY REALISED DURING TESTING PHASE…. (OLD CODE)

```
//THESE ARE ALL TEST CASES......SELECT ONE ONLY!
//String text = "The quick brown fox jumps over the lazy dog";
//String text="";
//String text = " This is a test, but making it a bit longer!";
string text = "My";
//String text = "My name is Amit Amlani. This is a sligthly longer test to see if the text can be spanr
//String text = "This will be 16.";
//String text = "Thiswillbetesting  16 testing."
```

*****OUTPUT*****

Welcome to Online IDE!! Happy Coding :)
Your line getting bigger:My(2 chars inc white space)
Words after truncation: 1
Finally completed the last line:My
Left over StringJoiner: 1 word(s)=> My (2 chars inc white space)
Total running words: 1
Buffer is: 14
Entering here most likely due to having one or no words in the line
So the StringBuilder should be empty:
My =>qualifies for 14 padding at end since it has:1 word(s)
completed line:My··············
NEW LENGTH of line:16

****THIS WILL PRINT ENTIRE TEXT**********

My•••••••••••••• 14 padding at end

****

** Process exited - Return Code: 0 **

TEST SCENARIO 5: AN EXTRA LONG SENTENCE, BUT ALSO TO SEE DISTINCTION BETWEEN THE PADDING AND FULL STOP (OLD CODE)

```
//THESE ARE ALL TEST CASES......SELECT ONE ONLY!
//String text = "The quick brown fox jumps over the lazy dog";
//String text="";
//String text = " This is a test, but making it a bit longer!";
//String text = "My";
String text = "My name is Amit Amlani. This is a sligthly longer test to see if the text can be spanned across multiple lines.";
//String text = "This will be 16.";
//String text = "Thiswillbetesting  16 testing."
```

*****OUTPUT*****

Welcome to Online IDE!! Happy Coding :)
Your line getting bigger:My(2 chars inc white space)
Words after truncation: 1
Your line getting bigger:My·name(7 chars inc white space)
Words after truncation: 2
Your line getting bigger:My·name·is(10 chars inc white space)
Words after truncation: 3
Your line getting bigger:My·name·is·Amit(15 chars inc white space)
Words after truncation: 4
Your line getting bigger:My·name·is·Amit·Amlani.(23 chars inc white space)
Words after truncation: 5
Rolled back to(due to exceeding 16):My·name·is·Amit
Length rolled back line: 15
This word getting truncated:Amlani.
The current buffer is: 1
Number words to accomodate for: 4(My·name·is·Amit)
Last word in tokenizer should be same as truncated word: Amlani.
What is wordcount here: 5
What is truncated word count: 4(My·name·is·Amit) 15 chars
The current buffer is: 1
My·name·is·Amit =>qualifies for 1 padding at front since it has:4 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·My·name·is·Amit
This word will be carried over to next line:Amlani.

Your line getting bigger:Amlani.·This(12 chars inc white space)
Words after truncation: 2
Your line getting bigger:Amlani.·This·is(15 chars inc white space)
Words after truncation: 3
Your line getting bigger:Amlani.·This·is·a(17 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):Amlani.·This·is
Length rolled back line: 15

This word getting truncated:a
The current buffer is: 1
Number words to accomodate for: 3(Amlani.·This·is)
Last word in tokenizer should be same as truncated word: a
What is wordcount here: 8
What is truncated word count: 3(Amlani.·This·is) 15 chars
The current buffer is: 1
Amlani.·This·is =>qualifies for 1 padding at front since it has:3 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·Amlani.·This·is
This word will be carried over to next line:a

Your line getting bigger:a·sligthly(10 chars inc white space)
Words after truncation: 2
Your line getting bigger:a·sligthly·longer(17 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):a·sligthly
Length rolled back line: 10
This word getting truncated:longer
The current buffer is: 6
Number words to accomodate for: 2(a·sligthly)
Last word in tokenizer should be same as truncated word: longer
What is wordcount here: 10
What is truncated word count: 2(a·sligthly) 10 chars
The current buffer is: 6
CURRENT LENGTH of line: 10
Qualified for extra padding
6 extra padding between:a·sligthly
It will now process truncated string with extra 6 padding between the words:a·sligthly
Your line getting bigger:a(1 chars inc white space)
Your line getting bigger:a·····sligthly(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:a·····sligthly
This word will be carried over to next line:longer

Your line getting bigger:longer·test(11 chars inc white space)
Words after truncation: 2
Your line getting bigger:longer·test·to(14 chars inc white space)
Words after truncation: 3
Your line getting bigger:longer·test·to·see(18 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):longer·test·to
Length rolled back line: 14
This word getting truncated:see
The current buffer is: 2
Number words to accomodate for: 3(longer·test·to)
Last word in tokenizer should be same as truncated word: see
What is wordcount here: 13
What is truncated word count: 3(longer·test·to) 14 chars
The current buffer is: 2
CURRENT LENGTH of line: 14
Qualified for extra padding
1 extra padding between:longer·test·to
It will now process truncated string with extra 1 padding between the words:longer·test·to
Your line getting bigger:longer(6 chars inc white space)
Your line getting bigger:longer··test(12 chars inc white space)

Your line getting bigger:longer··test··to(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:longer··test··to
This word will be carried over to next line:see

Your line getting bigger:see·if(6 chars inc white space)
Words after truncation: 2
Your line getting bigger:see·if·the(10 chars inc white space)
Words after truncation: 3
Your line getting bigger:see·if·the·text(15 chars inc white space)
Words after truncation: 4
Your line getting bigger:see·if·the·text·can(19 chars inc white space)
Words after truncation: 5
Rolled back to(due to exceeding 16):see·if·the·text
Length rolled back line: 15
This word getting truncated:can
The current buffer is: 1
Number words to accomodate for: 4(see·if·the·text)
Last word in tokenizer should be same as truncated word: can
What is wordcount here: 17
What is truncated word count: 4(see·if·the·text) 15 chars
The current buffer is: 1
see·if·the·text   =>qualifies for 1 padding at front since it has:4 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:··see·if·the·text
This word will be carried over to next line:can

Your line getting bigger:can·be(6 chars inc white space)
Words after truncation: 2
Your line getting bigger:can·be·spanned(14 chars inc white space)
Words after truncation: 3
Your line getting bigger:can·be·spanned·across(21 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):can·be·spanned
Length rolled back line: 14
This word getting truncated:across
The current buffer is: 2
Number words to accomodate for: 3(can·be·spanned)
Last word in tokenizer should be same as truncated word: across
What is wordcount here: 20
What is truncated word count: 3(can·be·spanned) 14 chars
The current buffer is: 2
CURRENT LENGTH of line: 14
Qualified for extra padding
1 extra padding between:can·be·spanned
It will now process truncated string with extra 1 padding between the words:can·be·spanned
Your line getting bigger:can(3 chars inc white space)
Your line getting bigger:can··be(7 chars inc white space)
Your line getting bigger:can··be··spanned(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:can··be··spanned
This word will be carried over to next line:across

Your line getting bigger:across·multiple(15 chars inc white space)
Words after truncation: 2
Your line getting bigger:across·multiple·lines.(22 chars inc white space)

Words after truncation: 3
Rolled back to(due to exceeding 16):across·multiple
Length rolled back line: 15
This word getting truncated:lines.
The current buffer is: 1
Number words to accomodate for: 2(across·multiple)
Last word in tokenizer should be same as truncated word: lines.
What is wordcount here: 22
What is truncated word count: 2(across·multiple) 15 chars
The current buffer is: 1
CURRENT LENGTH of line: 15
Qualified for extra padding
1 extra padding between:across·multiple
It will now process truncated string with extra 1 padding between the words:across·multiple
Your line getting bigger:across(6 chars inc white space)
Your line getting bigger:across··multiple(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:across··multiple
This word will be carried over to next line:lines.

Finally completed the last line:lines.
Left over StringJoiner: 1 word(s)=> lines. (6 chars inc white space)
Total running words: 22
Buffer is: 10
Entering here most likely due to having one or no words in the line
So the StringBuilder should be empty:
lines. =>qualifies for 10 padding at end since it has:1 word(s)
completed line:lines.··········
NEW LENGTH of line:16

****THIS WILL PRINT ENTIRE TEXT**********

·My·name·is·Amit

·Amlani.·This·is height of padding distinguishable to full stop

a·····sligthly

longer··test··to

·see·if·the·text

can··be··spanned

across··multiple

lines.•••••••••• height of padding distinguishable to full stop

****

** Process exited - Return Code: 0 **

TEST SCENARIO 6: EXACTLY K CHARS AND WHITESPACE (OLD CODE)

```
//THESE ARE ALL TEST CASES......SELECT ONE ONLY!
//String text = "The quick brown fox jumps over the lazy dog";
//String text="";
//String text = " This is a test, but making it a bit longer!";
//String text = "My";
//String text = "My name is Amit Amlani. This is a sligthly longer test to see if the text can be spanned acr
String text = "This will be 16.";
//String text = "Thiswillbetesting  16 testing."
```

*****OUTPUT*****

Welcome to Online IDE!! Happy Coding :)
Your line getting bigger:This(4 chars inc white space)
Words after truncation: 1
Your line getting bigger:This·will(9 chars inc white space)
Words after truncation: 2
Your line getting bigger:This·will·be(12 chars inc white space)
Words after truncation: 3
Your line getting bigger:This·will·be·16.(16 chars inc white space)
Words after truncation: 4
Finally completed the last line:This·will·be·16.
Left over StringJoiner: 4 word(s)=> This·will·be·16. (16 chars inc white space)
Total running words: 4
Buffer is: 0
It will now process the string with extra: 0 padding between words:This·will·be·16.
your line getting bigger:This(4 chars inc white space will NOT exceed 16)
your line getting bigger:This·will(9 chars inc white space will NOT exceed 16)
your line getting bigger:This·will·be(12 chars inc white space will NOT exceed 16)
your line getting bigger:This·will·be·16.(16 chars inc white space will NOT exceed 16)
completed line:This·will·be·16.
NEW LENGTH of line:16

****THIS WILL PRINT ENTIRE TEXT**********

This•will•be•16. Remans exactly same

****

** Process exited - Return Code: 0 **

TEST SCENARIO 7: FIRST LINE EXCEEDS K LIMIT (OLD CODE)

```
//THESE ARE ALL TEST CASES......SELECT ONE ONLY!
//String text = "The quick brown fox jumps over the lazy dog";
//String text="";
//String text = " This is a test, but making it a bit longer!";
//String text = "My";
//String text = "My name is Amit Amlani. This is a sligthly longer test to see if the text can be spanned acr
//String text = "This will be 16.";
string text = "Thiswillbetesting  16 testing."
```

*****OUTPUT*****

Welcome to Online IDE!! Happy Coding :)

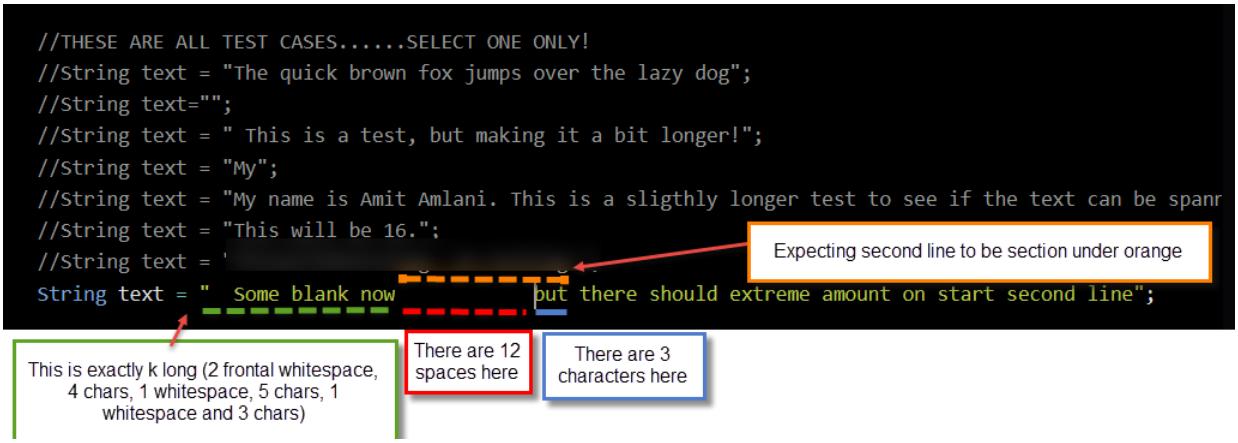
The following word exceeds line limit of k(16):
Thiswillbetesting

** Process exited - Return Code: 0 **

*** CODE **

SEE ATTACHMENT

TEST SCENARIO 8: THIS WAS ONLY INTRODUCED SINCE IT WAS REALISED A SCENARIO WAS NOT TESTED AS SUCH… THE FIRST LINE WILL START WITH BLANK SPACES AND THE WORDS WILL END PERFECTLY AT K CONSTRAINT, BUT THE NEXT LINE ALSO HAS NATURAL WHITESPACES IN THE STRING (OLD CODE)



*****OUTPUT*****

UNFORTUNATELY THIS TIME, IT HAS FAILED SEVERELY!

****THIS WILL PRINT ENTIRE TEXT**********

Some··blank··now it has lost leading padding and hence kept padding inter words

but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second·line a severe mess in line 2 and line 3

****

My initial logic tells me it is something related to having customized paddingTest character… But it is totally unrelated…
It was later figured out…
It is quite upsetting that this test case was not explored….
This would be great chance to examine the screen outputs. I am hoping its not too detrimental since it is quite upsetting to find this at end of code.

At first instance, the only possible way to sort this is to keeping running total of characters (white space included) of the text (up to point of end of token outputted).
Need to be careful since a token might be repeated multiple times, so how can exact location be found without further errors?
Only possible if the text is stripped off at intervals of the token……
So the initial text has to be stored in a StringBuilder….

ALSO:
keep running total of the length of the tokens…
Anything in between (presumed to be whitespace) has to be placed back in as natural padding…
This has to be done before additional padding since it will compromise k…..

MY CODE WAS INCREASINGLY LOOKING LIKE IT WILL TAKE LOTS UNPRECEDENTED CHANGES…

*****OUTPUT*****

Welcome to Online IDE!! Happy Coding :)
Your line getting bigger:Some(4 chars inc white space) it has failed straight away and lost frontal spaces.. This tells me that it is related to the String Tokenizer.. Until now, my assumption was that delimiter is inter-words…
So need to find a technique to keep the blank spaces…
This would have to be done at start of each line……
Words after truncation: 1
Your line getting bigger:Some·blank(10 chars inc white space)
Words after truncation: 2
Your line getting bigger:Some·blank·now(14 chars inc white space)
Words after truncation: 3
Your line getting bigger:Some·blank·now·but(18 chars inc white space)

Words after truncation: 4
Rolled back to(due to exceeding 16):Some·blank·now
Length rolled back line: 14
This word getting truncated:but
The current buffer is: 2
Number words to accomodate for: 3(Some·blank·now)
Last word in tokenizer should be same as truncated word: but
What is wordcount here: 4
What is truncated word count: 3(Some·blank·now) 14 chars
The current buffer is: 2
CURRENT LENGTH of line: 14
Qualified for extra padding
1 extra padding between:Some·blank·now
It will now process truncated string with extra 1 padding between the words:Some·blank·now
Your line getting bigger:Some(4 chars inc white space)
Your line getting bigger:Some··blank(11 chars inc white space)
Your line getting bigger:Some··blank··now(16 chars inc white space)
NEW LENGTH of line after formatting: 0
Completed line:Some··blank··now
This word will be carried over to next line:but

Your line getting bigger:but·there(9 chars inc white space)
Words after truncation: 2
Your line getting bigger:but·there·should(16 chars inc white space)
Words after truncation: 3
Your line getting bigger:but·there·should·extreme(24 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):but·there·should
Length rolled back line: 16
This word getting truncated:extreme
The current buffer is: 0
Number words to accomodate for: 3(but·there·should)
Last word in tokenizer should be same as truncated word: extreme
What is wordcount here: 7
What is truncated word count: 3(but·there·should) 16 chars
The current buffer is: 0
Your line getting bigger:but·there·should·extreme·extreme·amount(39 chars inc white space)
Words after truncation: 5
Rolled back to(due to exceeding 16):but·there·should·extreme·extreme
Length rolled back line: 32
This word getting truncated:amount
The current buffer is: -16
Number words to accomodate for: 4(but·there·should·extreme·extreme)
Last word in tokenizer should be same as truncated word: amount
What is wordcount here: 8
What is truncated word count: 4(but·there·should·extreme·extreme) 32 chars
The current buffer is: -16
Your line getting bigger:but·there·should·extreme·extreme·amount·amount·on(49 chars inc white space)
Words after truncation: 6
Rolled back to(due to exceeding 16):but·there·should·extreme·extreme·amount·amount
Length rolled back line: 46
This word getting truncated:on
The current buffer is: -30
Number words to accomodate for: 5(but·there·should·extreme·extreme·amount·amount)
Last word in tokenizer should be same as truncated word: on
What is wordcount here: 9
What is truncated word count: 5(but·there·should·extreme·extreme·amount·amount) 46 chars
The current buffer is: -30
Your line getting bigger:but·there·should·extreme·extreme·amount·amount·on·on·start(58 chars inc white space)
Words after truncation: 7
Rolled back to(due to exceeding 16):but·there·should·extreme·extreme·amount·amount·on·on
Length rolled back line: 52

This word getting truncated:start
The current buffer is: -36
Number words to accomodate for: 6(but·there·should·extreme·extreme·amount·amount·on·on)
Last word in tokenizer should be same as truncated word: start
What is wordcount here: 10
What is truncated word count: 6(but·there·should·extreme·extreme·amount·amount·on·on) 52 chars
The current buffer is: -36
Your line getting bigger:but·there·should·extreme·extreme·amount·amount·on·on·start·start·second(71 chars inc white space)
Words after truncation: 8
Rolled back to(due to exceeding 16):but·there·should·extreme·extreme·amount·amount·on·on·start·start
Length rolled back line: 64
This word getting truncated:second
The current buffer is: -48
Number words to accomodate for: 7(but·there·should·extreme·extreme·amount·amount·on·on·start·start)
Last word in tokenizer should be same as truncated word: second
What is wordcount here: 11
What is truncated word count: 7(but·there·should·extreme·extreme·amount·amount·on·on·start·start) 64 chars
The current buffer is: -48
Your line getting bigger:but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second·line(83 chars inc white space)
Words after truncation: 9
Rolled back to(due to exceeding 16):but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second
Length rolled back line: 78
This word getting truncated:line
The current buffer is: -62
Number words to accomodate for: 8(but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second)
Last word in tokenizer should be same as truncated word: line
What is wordcount here: 12
What is truncated word count: 8(but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second) 78 chars
The current buffer is: -62
Finally completed the last line:but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second·line
Left over StringJoiner: 8 word(s)=>
but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second·line (83 chars inc white space)
Total running words: 12
Buffer is: -67
There is one word in last line:line
CURRENT LENGTH of line: 4
but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second·line =>qualifies for -67 padding at end since it has:8 word(s)
completed line:but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second·line
NEW LENGTH of line:83

****THIS WILL PRINT ENTIRE TEXT**********
Some··blank··now
but·there·should·extreme·extreme·amount·amount·on·on·start·start·second·second·line
****

** Process exited - Return Code: 0 **

I have remediated this situation… I think it would also begin to add lots of unnecessary code in a way which would not suit the design that I chose to use.
So, the code would be less readable and also perhaps lose its professionalism.
But I had to complete the action at minimum for front existingFrontalPadding since requirements did not state to remove this…
Also, in real world for example, if end user has chosen to deliberately including front space such as in headings, this will be compromised…
So a fix was required…

Retrospectively, the exercise did promote uniform space inter words..
So it would be a contradiction trying to force issue and further implement beyond the frontal padding.

Due to all the changes, I created another coded version since it effectively changed the pseudo code and code arrangement of the first..
If there is ever any need to maintain existing inter-word padding, it is explained how it can be achieved using the figure overleaf.

This has been included in st.hasMoreTokens() to support execution...

If there is ever need to fit existing inter-word padding back, then it has to be considered by using similar logic (as bottom right image), creating a variable such as existingInterWordPadding for wordcount>0 (since this will apply at token level) and following same processs

```
if (wordCount==0)
{
    System.out.println("currently inside String Builder:" + sb.toString());
    System.out.println("GGGGG");
    sb.append( existingFrontalPadding );
    sb.append(tokentoString);
    System.out.println("existing frontal padding detected:" + ( existingFrontalPadding );

    alternateFirstToken= extraFrontalPadding + tokentoString;
```

```
nline IDE!! Happy Coding :)
h value:0
h value:1
h value:2
currently inside String Builder:
GGGGG
existing frontal padding detected:**
This is the token here:Some
currently inside StringBuilder again:Some
ALTERNATE FIRST TOKEN WITH EXISTING FRONTAL PADDING:**Some
option 2
INSIDE StringJoiner:**·Some
unformmated line here so far:**·Some
Your line getting bigger:**·Some(7 chars inc white space)
Words after truncation: 1
This is the token here:blank
currently inside StringBuilder again:blank
ALTERNATE FIRST TOKEN WITH EXISTING FRONTAL PADDING:**Some
option 1
INSIDE StringJoiner:**·Some·blank
unformmated line here so far:**·Some·blank
Your line getting bigger:**·Some·blank(13 chars inc white space)
Words after truncation: 2
```

This is first point of extracting value from the text.. Tokenizer has also stripped the first word.. This was something I never would have expected and clearly taking this temp was not appropriate

As expected, the StringJoiner will also be influenced...

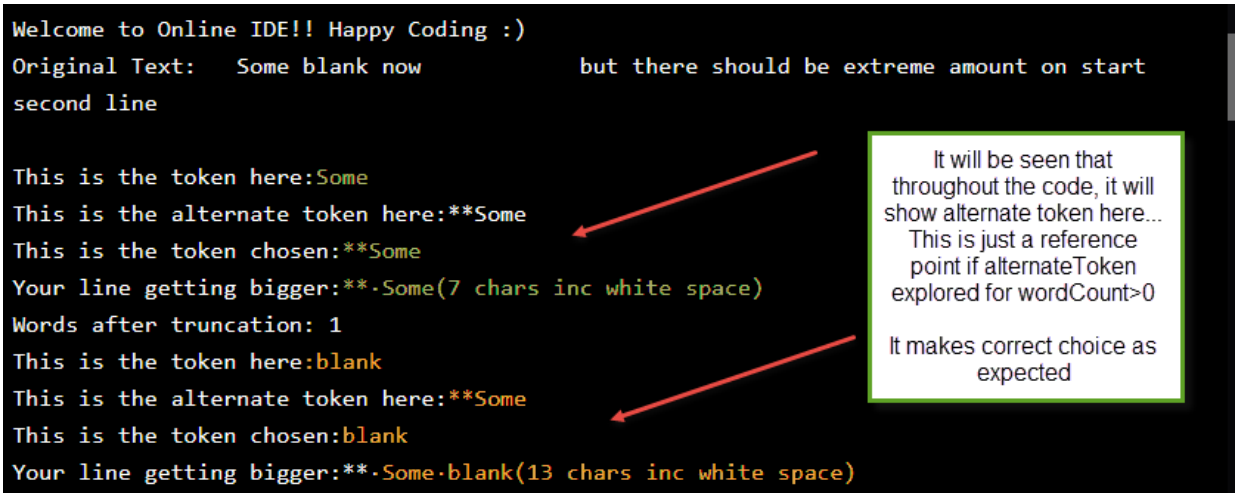
This is now correct and can be seen yellow highlights that its all resolved...

```
if ( existingFrontalPadding =="")
{
    System.out.println("option 1");
sj.add(sb);  //  StringBuilder added into s
}
else
{
    System.out.println("option 2");
    sj.add( existingFrontalPadding );
    sj.add(tokentoString);
}

existingFrontalPadding ="";
```

I wanted to stay away from this iteractive

```
for (int h=newposText; h<text.length(); h++)
{
    System.out.println("h value:" + h);
    if (text.charAt(h)==' ')
    {
        existingFrontalPadding = existingFrontalPadding + existingFrontalPadding;
        existingFrontalPaddingCount++;
    }
    else
    {
        h=newposText;
        //System.out.println("Extra:" + existingFrontalPadding );
        break;
```



I have finally completed the code.. I did lots of testing.. And this was required given my coding approach of just not iteratively traversing text.
I consider this to be toughest challenge to date given the approach undertaken…

TEST SCENARIO 1 (NEW CODE) - example given…..

```
String text = "The quick brown fox jumps over the lazy dog";  // no issues.....
```

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

*·The·quick·brown

fox··jumps··over

the···lazy···dog

****

TEST SCENARIO 2 (NEW CODE) - no text

```
String text="";
```

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

****

TEST SCENARIO 3 (NEW CODE) - as described

```
String text = " This is a test, but making it a bit longer!";
```

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

*This·is·a·test,
·but·making·it·a
bit·····longer!

****

This is first occurrence of existing natural padding on left hand side.. It forms part of the first word….

TEST SCENARIO 4 (NEW CODE) - single word

```
String text = "My";
```

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

My·····

****

TEST SCENARIO 5 (NEW CODE) - longer line and extended frontal padding

****THIS WILL PRINT ENTIRE TEXT**********

```
string text = "      My name is Amit Amlani. This is a sligthly longer test to see if the text can be spanned across multiple lines.";
```

KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

******My·name·is
Amit·····Amlani.
·······This·is·a
sligthly··longer
··test·to·see·if
·the·text·can·be
spanned···across
multiple··lines.

****

TEST SCENARIO 6 (NEW CODE) - longer line and heavy intermittent padding - FAIL

```
string text = "  Some blank now            but there should be extreme amount on start second line";
```

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

**Some·blank·now but·there·should be·····extreme ·amount·on·start second·····line
****

Although there is large space, it is not consistent with text.. It is as explained, Tokenizer does not maintain extra whitespaces in original text.. Apart from that, it is all formatted as expected…

TEST SCENARIO 7 (NEW CODE) - exact fit with k variable

```
String text = "This will be 16.";   //no issues.
```

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces.. BUT expect user to have single whitespace for the justification to look tidy

This·will·be·16.
****

TEST SCENARIO 8 (NEW CODE) - first line in exact k boundary. Second line with word Amit

```
string text = "This will be 16. Amit";  //no issues....
```

This is scenario I did not test with old code.. And it would have failed…

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit

·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

This·will·be·16.
Amit·····
****

TEST SCENARIO 9 (NEW CODE) - word exceeds k limit

```
string text = "Thiswillbetesting  16 testing.";
```

The following word exceeds line limit of k(16):Thiswillbetesting

TEST SCENARIO 10 (NEW CODE) - pushing the limit

```
Welcome to Online IDE!! Happy Coding :)
Original Text: Java is a high-level, class-based2222222, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA),[16] meaning that compiled Java code can run on all platforms that support Java without the need to recompile.[17] Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages.
```

The following word exceeds line limit of k(16):class-based2222222,

TEST SCENARIO 11 (NEW CODE) - resolving woed limit issue in line, executing again with full ***OUTPUT**

Welcome to Online IDE!! Happy Coding :)
Original Text: Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA),[16] meaning that compiled Java code can run on all platforms that support Java without the need to recompile.[17] Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages.

Your line getting bigger:Java(4 chars inc white space)
Words after truncation: 1
Your line getting bigger:Java·is(7 chars inc white space)
Words after truncation: 2
Your line getting bigger:Java·is·a(9 chars inc white space)
Words after truncation: 3
Your line getting bigger:Java·is·a·high-level,(21 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):Java·is·a
Length rolled back line: 9
This word getting truncated:high-level,
The current buffer is: 7
Number words to accomodate for: 3(Java·is·a)
Last word in tokenizer should be same as truncated word: high-level,
What is wordcount here: 4
What is truncated word count: 3(Java·is·a) 9 chars
Value of temp:high-level,
Java·is·a =>qualifies for 7 padding at front since it has:3 words
CURRENT LENGTH of line: 9
NEW LENGTH of line after formatting: 16
Completed line:·····Java·is·a
This word will be carried over to next line:high-level,

Your line getting bigger:high-level,·class-based,(24 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):high-level,
Length rolled back line: 11
This word getting truncated:class-based,
The current buffer is: 5
Number words to accomodate for: 1(high-level,)
Last word in tokenizer should be same as truncated word: class-based,
What is wordcount here: 5
What is truncated word count: 1(high-level,) 11 chars
Value of temp:class-based,
high-level, =>qualifies for 5 padding at end since it has:1 words
CURRENT LENGTH of line: 11
NEW LENGTH of line after formatting: 16
Completed line:high-level,·····
This word will be carried over to next line:class-based,

Your line getting bigger:class-based,·object-oriented(28 chars inc white space)

Words after truncation: 2
Rolled back to(due to exceeding 16):class-based,
Length rolled back line: 12
This word getting truncated:object-oriented
The current buffer is: 4
Number words to accomodate for: 1(class-based,)
Last word in tokenizer should be same as truncated word: object-oriented
What is wordcount here: 6
What is truncated word count: 1(class-based,) 12 chars
Value of temp:object-oriented
class-based,   =>qualifies for 4 padding at end since it has:1 words
CURRENT LENGTH of line: 12
NEW LENGTH of line after formatting: 16
Completed line:class-based,····
This word will be carried over to next line:object-oriented

Your line getting bigger:object-oriented·programming(27 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):object-oriented
Length rolled back line: 15
This word getting truncated:programming
The current buffer is: 1
Number words to accomodate for: 1(object-oriented)
Last word in tokenizer should be same as truncated word: programming
What is wordcount here: 7
What is truncated word count: 1(object-oriented) 15 chars
Value of temp:programming
object-oriented   =>qualifies for 1 padding at end since it has:1 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:object-oriented·
This word will be carried over to next line:programming

Your line getting bigger:programming·language(20 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):programming
Length rolled back line: 11
This word getting truncated:language
The current buffer is: 5
Number words to accomodate for: 1(programming)
Last word in tokenizer should be same as truncated word: language
What is wordcount here: 8
What is truncated word count: 1(programming) 11 chars
Value of temp:language
programming   =>qualifies for 5 padding at end since it has:1 words
CURRENT LENGTH of line: 11
NEW LENGTH of line after formatting: 16
Completed line:programming·····
This word will be carried over to next line:language

Your line getting bigger:language·that(13 chars inc white space)
Words after truncation: 2
Your line getting bigger:language·that·is(16 chars inc white space)
Words after truncation: 3
Completed line:language·that·is
NEW LENGTH of line:16
Your line getting bigger:designed(8 chars inc white space)

Words after truncation: 1
Your line getting bigger:designed·to(11 chars inc white space)
Words after truncation: 2
Your line getting bigger:designed·to·have(16 chars inc white space)
Words after truncation: 3
Completed line:designed·to·have
NEW LENGTH of line:16
Your line getting bigger:as(2 chars inc white space)
Words after truncation: 1
Your line getting bigger:as·few(6 chars inc white space)
Words after truncation: 2
Your line getting bigger:as·few·implementation(21 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):as·few
Length rolled back line: 6
This word getting truncated:implementation
The current buffer is: 10
Number words to accomodate for: 2(as·few)
Last word in tokenizer should be same as truncated word: implementation
What is wordcount here: 16
What is truncated word count: 2(as·few) 6 chars
Value of temp:implementation
CURRENT LENGTH of line: 6
Qualified for extra padding
10 extra padding between:as·few
It will now process truncated string with extra 10 padding between the words:as·few
Your line getting bigger:as(2 chars inc white space)
Your line getting bigger:as··········few(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:as··········few
This word will be carried over to next line:implementation

Your line getting bigger:implementation·dependencies(27 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):implementation
Length rolled back line: 14
This word getting truncated:dependencies
The current buffer is: 2
Number words to accomodate for: 1(implementation)
Last word in tokenizer should be same as truncated word: dependencies
What is wordcount here: 17
What is truncated word count: 1(implementation) 14 chars
Value of temp:dependencies
implementation =>qualifies for 2 padding at end since it has:1 words
CURRENT LENGTH of line: 14
NEW LENGTH of line after formatting: 16
Completed line:implementation··
This word will be carried over to next line:dependencies

Your line getting bigger:dependencies·as(15 chars inc white space)
Words after truncation: 2
Your line getting bigger:dependencies·as·possible.(25 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):dependencies·as
Length rolled back line: 15
This word getting truncated:possible.
The current buffer is: 1

Number words to accomodate for: 2(dependencies·as)
Last word in tokenizer should be same as truncated word: possible.
What is wordcount here: 19
What is truncated word count: 2(dependencies·as) 15 chars
Value of temp:possible.
CURRENT LENGTH of line: 15
Qualified for extra padding
1 extra padding between:dependencies·as
It will now process truncated string with extra 1 padding between the words:dependencies·as
Your line getting bigger:dependencies(12 chars inc white space)
Your line getting bigger:dependencies··as(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:dependencies··as
This word will be carried over to next line:possible.

Your line getting bigger:possible.·It(12 chars inc white space)
Words after truncation: 2
Your line getting bigger:possible.·It·is(15 chars inc white space)
Words after truncation: 3
Your line getting bigger:possible.·It·is·a(17 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):possible.·It·is
Length rolled back line: 15
This word getting truncated:a
The current buffer is: 1
Number words to accomodate for: 3(possible.·It·is)
Last word in tokenizer should be same as truncated word: a
What is wordcount here: 22
What is truncated word count: 3(possible.·It·is) 15 chars
Value of temp:a
possible.·It·is =>qualifies for 1 padding at front since it has:3 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·possible.·It·is
This word will be carried over to next line:a

Your line getting bigger:a·general-purpose(17 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):a
Length rolled back line: 1
This word getting truncated:general-purpose
The current buffer is: 15
Number words to accomodate for: 1(a)
Last word in tokenizer should be same as truncated word: general-purpose
What is wordcount here: 23
What is truncated word count: 1(a) 1 chars
Value of temp:general-purpose
a =>qualifies for 15 padding at end since it has:1 words
CURRENT LENGTH of line: 1
NEW LENGTH of line after formatting: 16
Completed line:a···············
This word will be carried over to next line:general-purpose

Your line getting bigger:general-purpose·programming(27 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):general-purpose
Length rolled back line: 15

This word getting truncated:programming
The current buffer is: 1
Number words to accomodate for: 1(general-purpose)
Last word in tokenizer should be same as truncated word: programming
What is wordcount here: 24
What is truncated word count: 1(general-purpose) 15 chars
Value of temp:programming
general-purpose   =>qualifies for 1 padding at end since it has:1 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:general-purpose·
This word will be carried over to next line:programming

Your line getting bigger:programming·language(20 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):programming
Length rolled back line: 11
This word getting truncated:language
The current buffer is: 5
Number words to accomodate for: 1(programming)
Last word in tokenizer should be same as truncated word: language
What is wordcount here: 25
What is truncated word count: 1(programming) 11 chars
Value of temp:language
programming   =>qualifies for 5 padding at end since it has:1 words
CURRENT LENGTH of line: 11
NEW LENGTH of line after formatting: 16
Completed line:programming·····
This word will be carried over to next line:language

Your line getting bigger:language·intended(17 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):language
Length rolled back line: 8
This word getting truncated:intended
The current buffer is: 8
Number words to accomodate for: 1(language)
Last word in tokenizer should be same as truncated word: intended
What is wordcount here: 26
What is truncated word count: 1(language) 8 chars
Value of temp:intended
language   =>qualifies for 8 padding at end since it has:1 words
CURRENT LENGTH of line: 8
NEW LENGTH of line after formatting: 16
Completed line:language·····
This word will be carried over to next line:intended

Your line getting bigger:intended·to(11 chars inc white space)
Words after truncation: 2
Your line getting bigger:intended·to·let(15 chars inc white space)
Words after truncation: 3
Your line getting bigger:intended·to·let·programmers(27 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):intended·to·let
Length rolled back line: 15
This word getting truncated:programmers
The current buffer is: 1

Number words to accomodate for: 3(intended·to·let)
Last word in tokenizer should be same as truncated word: programmers
What is wordcount here: 29
What is truncated word count: 3(intended·to·let) 15 chars
Value of temp:programmers
intended·to·let   =>qualifies for 1 padding at front since it has:3 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·intended·to·let
This word will be carried over to next line:programmers

Your line getting bigger:programmers·write(17 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):programmers
Length rolled back line: 11
This word getting truncated:write
The current buffer is: 5
Number words to accomodate for: 1(programmers)
Last word in tokenizer should be same as truncated word: write
What is wordcount here: 30
What is truncated word count: 1(programmers) 11 chars
Value of temp:write
programmers   =>qualifies for 5 padding at end since it has:1 words
CURRENT LENGTH of line: 11
NEW LENGTH of line after formatting: 16
Completed line:programmers·····
This word will be carried over to next line:write

Your line getting bigger:write·once,(11 chars inc white space)
Words after truncation: 2
Your line getting bigger:write·once,·run(15 chars inc white space)
Words after truncation: 3
Your line getting bigger:write·once,·run·anywhere(24 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):write·once,·run
Length rolled back line: 15
This word getting truncated:anywhere
The current buffer is: 1
Number words to accomodate for: 3(write·once,·run)
Last word in tokenizer should be same as truncated word: anywhere
What is wordcount here: 33
What is truncated word count: 3(write·once,·run) 15 chars
Value of temp:anywhere
write·once,·run   =>qualifies for 1 padding at front since it has:3 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·write·once,·run
This word will be carried over to next line:anywhere

Your line getting bigger:anywhere·(WORA),[16](20 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):anywhere
Length rolled back line: 8
This word getting truncated:(WORA),[16]
The current buffer is: 8
Number words to accomodate for: 1(anywhere)
Last word in tokenizer should be same as truncated word: (WORA),[16]

What is wordcount here: 34
What is truncated word count: 1(anywhere) 8 chars
Value of temp:(WORA),[16]
anywhere   =>qualifies for 8 padding at end since it has:1 words
CURRENT LENGTH of line: 8
NEW LENGTH of line after formatting: 16
Completed line:anywhere·····
This word will be carried over to next line:(WORA),[16]

Your line getting bigger:(WORA),[16]·meaning(19 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):(WORA),[16]
Length rolled back line: 11
This word getting truncated:meaning
The current buffer is: 5
Number words to accomodate for: 1((WORA),[16])
Last word in tokenizer should be same as truncated word: meaning
What is wordcount here: 35
What is truncated word count: 1((WORA),[16]) 11 chars
Value of temp:meaning
(WORA),[16]   =>qualifies for 5 padding at end since it has:1 words
CURRENT LENGTH of line: 11
NEW LENGTH of line after formatting: 16
Completed line:(WORA),[16]·····
This word will be carried over to next line:meaning

Your line getting bigger:meaning·that(12 chars inc white space)
Words after truncation: 2
Your line getting bigger:meaning·that·compiled(21 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):meaning·that
Length rolled back line: 12
This word getting truncated:compiled
The current buffer is: 4
Number words to accomodate for: 2(meaning·that)
Last word in tokenizer should be same as truncated word: compiled
What is wordcount here: 37
What is truncated word count: 2(meaning·that) 12 chars
Value of temp:compiled
CURRENT LENGTH of line: 12
Qualified for extra padding
4 extra padding between:meaning·that
It will now process truncated string with extra 4 padding between the words:meaning·that
Your line getting bigger:meaning(7 chars inc white space)
Your line getting bigger:meaning·····that(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:meaning·····that
This word will be carried over to next line:compiled

Your line getting bigger:compiled·Java(13 chars inc white space)
Words after truncation: 2
Your line getting bigger:compiled·Java·code(18 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):compiled·Java
Length rolled back line: 13
This word getting truncated:code
The current buffer is: 3

Number words to accomodate for: 2(compiled·Java)
Last word in tokenizer should be same as truncated word: code
What is wordcount here: 39
What is truncated word count: 2(compiled·Java) 13 chars
Value of temp:code
CURRENT LENGTH of line: 13
Qualified for extra padding
3 extra padding between:compiled·Java
It will now process truncated string with extra 3 padding between the words:compiled·Java
Your line getting bigger:compiled(8 chars inc white space)
Your line getting bigger:compiled····Java(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:compiled····Java
This word will be carried over to next line:code

Your line getting bigger:code·can(8 chars inc white space)
Words after truncation: 2
Your line getting bigger:code·can·run(12 chars inc white space)
Words after truncation: 3
Your line getting bigger:code·can·run·on(15 chars inc white space)
Words after truncation: 4
Your line getting bigger:code·can·run·on·all(19 chars inc white space)
Words after truncation: 5
Rolled back to(due to exceeding 16):code·can·run·on
Length rolled back line: 15
This word getting truncated:all
The current buffer is: 1
Number words to accomodate for: 4(code·can·run·on)
Last word in tokenizer should be same as truncated word: all
What is wordcount here: 43
What is truncated word count: 4(code·can·run·on) 15 chars
Value of temp:all
code·can·run·on =>qualifies for 1 padding at front since it has:4 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·code·can·run·on
This word will be carried over to next line:all

Your line getting bigger:all·platforms(13 chars inc white space)
Words after truncation: 2
Your line getting bigger:all·platforms·that(18 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):all·platforms
Length rolled back line: 13
This word getting truncated:that
The current buffer is: 3
Number words to accomodate for: 2(all·platforms)
Last word in tokenizer should be same as truncated word: that
What is wordcount here: 45
What is truncated word count: 2(all·platforms) 13 chars
Value of temp:that
CURRENT LENGTH of line: 13
Qualified for extra padding
3 extra padding between:all·platforms
It will now process truncated string with extra 3 padding between the words:all·platforms
Your line getting bigger:all(3 chars inc white space)
Your line getting bigger:all····platforms(16 chars inc white space)

NEW LENGTH of line after formatting: 16
Completed line:all····platforms
This word will be carried over to next line:that

Your line getting bigger:that·support(12 chars inc white space)
Words after truncation: 2
Your line getting bigger:that·support·Java(17 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):that·support
Length rolled back line: 12
This word getting truncated:Java
The current buffer is: 4
Number words to accomodate for: 2(that·support)
Last word in tokenizer should be same as truncated word: Java
What is wordcount here: 47
What is truncated word count: 2(that·support) 12 chars
Value of temp:Java
CURRENT LENGTH of line: 12
Qualified for extra padding
4 extra padding between:that·support
It will now process truncated string with extra 4 padding between the words:that·support
Your line getting bigger:that(4 chars inc white space)
Your line getting bigger:that·····support(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:that·····support
This word will be carried over to next line:Java

Your line getting bigger:Java·without(12 chars inc white space)
Words after truncation: 2
Your line getting bigger:Java·without·the(16 chars inc white space)
Words after truncation: 3
Completed line:Java·without·the
NEW LENGTH of line:16
Your line getting bigger:need(4 chars inc white space)
Words after truncation: 1
Your line getting bigger:need·to(7 chars inc white space)
Words after truncation: 2
Your line getting bigger:need·to·recompile.[17](22 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):need·to
Length rolled back line: 7
This word getting truncated:recompile.[17]
The current buffer is: 9
Number words to accomodate for: 2(need·to)
Last word in tokenizer should be same as truncated word: recompile.[17]
What is wordcount here: 52
What is truncated word count: 2(need·to) 7 chars
Value of temp:recompile.[17]
CURRENT LENGTH of line: 7
Qualified for extra padding
9 extra padding between:need·to
It will now process truncated string with extra 9 padding between the words:need·to
Your line getting bigger:need(4 chars inc white space)
Your line getting bigger:need·····to(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:need·····to
This word will be carried over to next line:recompile.[17]

Your line getting bigger:recompile.[17]·Java(19 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):recompile.[17]
Length rolled back line: 14
This word getting truncated:Java
The current buffer is: 2
Number words to accomodate for: 1(recompile.[17])
Last word in tokenizer should be same as truncated word: Java
What is wordcount here: 53
What is truncated word count: 1(recompile.[17]) 14 chars
Value of temp:Java
recompile.[17] =>qualifies for 2 padding at end since it has:1 words
CURRENT LENGTH of line: 14
NEW LENGTH of line after formatting: 16
Completed line:recompile.[17]··
This word will be carried over to next line:Java

Your line getting bigger:Java·applications(17 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):Java
Length rolled back line: 4
This word getting truncated:applications
The current buffer is: 12
Number words to accomodate for: 1(Java)
Last word in tokenizer should be same as truncated word: applications
What is wordcount here: 54
What is truncated word count: 1(Java) 4 chars
Value of temp:applications
Java =>qualifies for 12 padding at end since it has:1 words
CURRENT LENGTH of line: 4
NEW LENGTH of line after formatting: 16
Completed line:Java····
This word will be carried over to next line:applications

Your line getting bigger:applications·are(16 chars inc white space)
Words after truncation: 2
Completed line:applications·are
NEW LENGTH of line:16
Your line getting bigger:typically(9 chars inc white space)
Words after truncation: 1
Your line getting bigger:typically·compiled(18 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):typically
Length rolled back line: 9
This word getting truncated:compiled
The current buffer is: 7
Number words to accomodate for: 1(typically)
Last word in tokenizer should be same as truncated word: compiled
What is wordcount here: 57
What is truncated word count: 1(typically) 9 chars
Value of temp:compiled
typically =>qualifies for 7 padding at end since it has:1 words
CURRENT LENGTH of line: 9
NEW LENGTH of line after formatting: 16
Completed line:typically····
This word will be carried over to next line:compiled

Your line getting bigger:compiled·to(11 chars inc white space)
Words after truncation: 2
Your line getting bigger:compiled·to·bytecode(20 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):compiled·to
Length rolled back line: 11
This word getting truncated:bytecode
The current buffer is: 5
Number words to accomodate for: 2(compiled·to)
Last word in tokenizer should be same as truncated word: bytecode
What is wordcount here: 59
What is truncated word count: 2(compiled·to) 11 chars
Value of temp:bytecode
CURRENT LENGTH of line: 11
Qualified for extra padding
5 extra padding between:compiled·to
It will now process truncated string with extra 5 padding between the words:compiled·to
Your line getting bigger:compiled(8 chars inc white space)
Your line getting bigger:compiled·····to(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:compiled·····to
This word will be carried over to next line:bytecode

Your line getting bigger:bytecode·that(13 chars inc white space)
Words after truncation: 2
Your line getting bigger:bytecode·that·can(17 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):bytecode·that
Length rolled back line: 13
This word getting truncated:can
The current buffer is: 3
Number words to accomodate for: 2(bytecode·that)
Last word in tokenizer should be same as truncated word: can
What is wordcount here: 61
What is truncated word count: 2(bytecode·that) 13 chars
Value of temp:can
CURRENT LENGTH of line: 13
Qualified for extra padding
3 extra padding between:bytecode·that
It will now process truncated string with extra 3 padding between the words:bytecode·that
Your line getting bigger:bytecode(8 chars inc white space)
Your line getting bigger:bytecode····that(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:bytecode····that
This word will be carried over to next line:can

Your line getting bigger:can·run(7 chars inc white space)
Words after truncation: 2
Your line getting bigger:can·run·on(10 chars inc white space)
Words after truncation: 3
Your line getting bigger:can·run·on·any(14 chars inc white space)
Words after truncation: 4
Your line getting bigger:can·run·on·any·Java(19 chars inc white space)
Words after truncation: 5
Rolled back to(due to exceeding 16):can·run·on·any
Length rolled back line: 14

This word getting truncated:Java
The current buffer is: 2
Number words to accomodate for: 4(can·run·on·any)
Last word in tokenizer should be same as truncated word: Java
What is wordcount here: 65
What is truncated word count: 4(can·run·on·any) 14 chars
Value of temp:Java
can·run·on·any   =>qualifies for 2 padding at front since it has:4 words
CURRENT LENGTH of line: 14
NEW LENGTH of line after formatting: 16
Completed line:··can·run·on·any
This word will be carried over to next line:Java

Your line getting bigger:Java·virtual(12 chars inc white space)
Words after truncation: 2
Your line getting bigger:Java·virtual·machine(20 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):Java·virtual
Length rolled back line: 12
This word getting truncated:machine
The current buffer is: 4
Number words to accomodate for: 2(Java·virtual)
Last word in tokenizer should be same as truncated word: machine
What is wordcount here: 67
What is truncated word count: 2(Java·virtual) 12 chars
Value of temp:machine
CURRENT LENGTH of line: 12
Qualified for extra padding
4 extra padding between:Java·virtual
It will now process truncated string with extra 4 padding between the words:Java·virtual
Your line getting bigger:Java(4 chars inc white space)
Your line getting bigger:Java····virtual(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:Java····virtual
This word will be carried over to next line:machine

Your line getting bigger:machine·(JVM)(13 chars inc white space)
Words after truncation: 2
Your line getting bigger:machine·(JVM)·regardless(24 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):machine·(JVM)
Length rolled back line: 13
This word getting truncated:regardless
The current buffer is: 3
Number words to accomodate for: 2(machine·(JVM))
Last word in tokenizer should be same as truncated word: regardless
What is wordcount here: 69
What is truncated word count: 2(machine·(JVM)) 13 chars
Value of temp:regardless
CURRENT LENGTH of line: 13
Qualified for extra padding
3 extra padding between:machine·(JVM)
It will now process truncated string with extra 3 padding between the words:machine·(JVM)
Your line getting bigger:machine(7 chars inc white space)
Your line getting bigger:machine···(JVM)(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:machine···(JVM)

This word will be carried over to next line:regardless

Your line getting bigger:regardless·of(13 chars inc white space)
Words after truncation: 2
Your line getting bigger:regardless·of·the(17 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):regardless·of
Length rolled back line: 13
This word getting truncated:the
The current buffer is: 3
Number words to accomodate for: 2(regardless·of)
Last word in tokenizer should be same as truncated word: the
What is wordcount here: 71
What is truncated word count: 2(regardless·of) 13 chars
Value of temp:the
CURRENT LENGTH of line: 13
Qualified for extra padding
3 extra padding between:regardless·of
It will now process truncated string with extra 3 padding between the words:regardless·of
Your line getting bigger:regardless(10 chars inc white space)
Your line getting bigger:regardless····of(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:regardless····of
This word will be carried over to next line:the

Your line getting bigger:the·underlying(14 chars inc white space)
Words after truncation: 2
Your line getting bigger:the·underlying·computer(23 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):the·underlying
Length rolled back line: 14
This word getting truncated:computer
The current buffer is: 2
Number words to accomodate for: 2(the·underlying)
Last word in tokenizer should be same as truncated word: computer
What is wordcount here: 73
What is truncated word count: 2(the·underlying) 14 chars
Value of temp:computer
CURRENT LENGTH of line: 14
Qualified for extra padding
2 extra padding between:the·underlying
It will now process truncated string with extra 2 padding between the words:the·underlying
Your line getting bigger:the(3 chars inc white space)
Your line getting bigger:the···underlying(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:the···underlying
This word will be carried over to next line:computer

Your line getting bigger:computer·architecture.(22 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):computer
Length rolled back line: 8
This word getting truncated:architecture.
The current buffer is: 8
Number words to accomodate for: 1(computer)
Last word in tokenizer should be same as truncated word: architecture.
What is wordcount here: 74

What is truncated word count: 1(computer) 8 chars
Value of temp:architecture.
computer   =>qualifies for 8 padding at end since it has:1 words
CURRENT LENGTH of line: 8
NEW LENGTH of line after formatting: 16
Completed line:computer·····
This word will be carried over to next line:architecture.

Your line getting bigger:architecture.·The(17 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):architecture.
Length rolled back line: 13
This word getting truncated:The
The current buffer is: 3
Number words to accomodate for: 1(architecture.)
Last word in tokenizer should be same as truncated word: The
What is wordcount here: 75
What is truncated word count: 1(architecture.) 13 chars
Value of temp:The
architecture.   =>qualifies for 3 padding at end since it has:1 words
CURRENT LENGTH of line: 13
NEW LENGTH of line after formatting: 16
Completed line:architecture.···
This word will be carried over to next line:The

Your line getting bigger:The·syntax(10 chars inc white space)
Words after truncation: 2
Your line getting bigger:The·syntax·of(13 chars inc white space)
Words after truncation: 3
Your line getting bigger:The·syntax·of·Java(18 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):The·syntax·of
Length rolled back line: 13
This word getting truncated:Java
The current buffer is: 3
Number words to accomodate for: 3(The·syntax·of)
Last word in tokenizer should be same as truncated word: Java
What is wordcount here: 78
What is truncated word count: 3(The·syntax·of) 13 chars
Value of temp:Java
The·syntax·of   =>qualifies for 3 padding at front since it has:3 words
CURRENT LENGTH of line: 13
NEW LENGTH of line after formatting: 16
Completed line:···The·syntax·of
This word will be carried over to next line:Java

Your line getting bigger:Java·is(7 chars inc white space)
Words after truncation: 2
Your line getting bigger:Java·is·similar(15 chars inc white space)
Words after truncation: 3
Your line getting bigger:Java·is·similar·to(18 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):Java·is·similar
Length rolled back line: 15
This word getting truncated:to
The current buffer is: 1
Number words to accomodate for: 3(Java·is·similar)

Last word in tokenizer should be same as truncated word: to
What is wordcount here: 81
What is truncated word count: 3(Java·is·similar) 15 chars
Value of temp:to
Java·is·similar =>qualifies for 1 padding at front since it has:3 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·Java·is·similar
This word will be carried over to next line:to

Your line getting bigger:to·C(4 chars inc white space)
Words after truncation: 2
Your line getting bigger:to·C·and(8 chars inc white space)
Words after truncation: 3
Your line getting bigger:to·C·and·C++,(13 chars inc white space)
Words after truncation: 4
Your line getting bigger:to·C·and·C++,·but(17 chars inc white space)
Words after truncation: 5
Rolled back to(due to exceeding 16):to·C·and·C++,
Length rolled back line: 13
This word getting truncated:but
The current buffer is: 3
Number words to accomodate for: 4(to·C·and·C++,)
Last word in tokenizer should be same as truncated word: but
What is wordcount here: 85
What is truncated word count: 4(to·C·and·C++,) 13 chars
Value of temp:but
CURRENT LENGTH of line: 13
Qualified for extra padding
1 extra padding between:to·C·and·C++,
It will now process truncated string with extra 1 padding between the words:to·C·and·C++,
Your line getting bigger:to(2 chars inc white space)
Your line getting bigger:to··C(5 chars inc white space)
Your line getting bigger:to··C··and(10 chars inc white space)
Your line getting bigger:to··C··and··C++,(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:to··C··and··C++,
This word will be carried over to next line:but

Your line getting bigger:but·has(7 chars inc white space)
Words after truncation: 2
Your line getting bigger:but·has·fewer(13 chars inc white space)
Words after truncation: 3
Your line getting bigger:but·has·fewer·low-level(23 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):but·has·fewer
Length rolled back line: 13
This word getting truncated:low-level
The current buffer is: 3
Number words to accomodate for: 3(but·has·fewer)
Last word in tokenizer should be same as truncated word: low-level
What is wordcount here: 88
What is truncated word count: 3(but·has·fewer) 13 chars
Value of temp:low-level
but·has·fewer =>qualifies for 3 padding at front since it has:3 words
CURRENT LENGTH of line: 13
NEW LENGTH of line after formatting: 16

Completed line:···but·has·fewer
This word will be carried over to next line:low-level

Your line getting bigger:low-level·facilities(20 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):low-level
Length rolled back line: 9
This word getting truncated:facilities
The current buffer is: 7
Number words to accomodate for: 1(low-level)
Last word in tokenizer should be same as truncated word: facilities
What is wordcount here: 89
What is truncated word count: 1(low-level) 9 chars
Value of temp:facilities
low-level =>qualifies for 7 padding at end since it has:1 words
CURRENT LENGTH of line: 9
NEW LENGTH of line after formatting: 16
Completed line:low-level·····
This word will be carried over to next line:facilities

Your line getting bigger:facilities·than(15 chars inc white space)
Words after truncation: 2
Your line getting bigger:facilities·than·either(22 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):facilities·than
Length rolled back line: 15
This word getting truncated:either
The current buffer is: 1
Number words to accomodate for: 2(facilities·than)
Last word in tokenizer should be same as truncated word: either
What is wordcount here: 91
What is truncated word count: 2(facilities·than) 15 chars
Value of temp:either
CURRENT LENGTH of line: 15
Qualified for extra padding
1 extra padding between:facilities·than
It will now process truncated string with extra 1 padding between the words:facilities·than
Your line getting bigger:facilities(10 chars inc white space)
Your line getting bigger:facilities··than(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:facilities··than
This word will be carried over to next line:either

Your line getting bigger:either·of(9 chars inc white space)
Words after truncation: 2
Your line getting bigger:either·of·them.(15 chars inc white space)
Words after truncation: 3
Your line getting bigger:either·of·them.·The(19 chars inc white space)
Words after truncation: 4
Rolled back to(due to exceeding 16):either·of·them.
Length rolled back line: 15
This word getting truncated:The
The current buffer is: 1
Number words to accomodate for: 3(either·of·them.)
Last word in tokenizer should be same as truncated word: The
What is wordcount here: 94
What is truncated word count: 3(either·of·them.) 15 chars

Value of temp:The
either·of·them. =>qualifies for 1 padding at front since it has:3 words
CURRENT LENGTH of line: 15
NEW LENGTH of line after formatting: 16
Completed line:·either·of·them.
This word will be carried over to next line:The

Your line getting bigger:The·Java(8 chars inc white space)
Words after truncation: 2
Your line getting bigger:The·Java·runtime(16 chars inc white space)
Words after truncation: 3
Completed line:The·Java·runtime
NEW LENGTH of line:16
Your line getting bigger:provides(8 chars inc white space)
Words after truncation: 1
Your line getting bigger:provides·dynamic(16 chars inc white space)
Words after truncation: 2
Completed line:provides·dynamic
NEW LENGTH of line:16
Your line getting bigger:capabilities(12 chars inc white space)
Words after truncation: 1
Your line getting bigger:capabilities·(such(18 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):capabilities
Length rolled back line: 12
This word getting truncated:(such
The current buffer is: 4
Number words to accomodate for: 1(capabilities)
Last word in tokenizer should be same as truncated word: (such
What is wordcount here: 100
What is truncated word count: 1(capabilities) 12 chars
Value of temp:(such
capabilities =>qualifies for 4 padding at end since it has:1 words
CURRENT LENGTH of line: 12
NEW LENGTH of line after formatting: 16
Completed line:capabilities····
This word will be carried over to next line:(such

Your line getting bigger:(such·as(8 chars inc white space)
Words after truncation: 2
Your line getting bigger:(such·as·reflection(19 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):(such·as
Length rolled back line: 8
This word getting truncated:reflection
The current buffer is: 8
Number words to accomodate for: 2((such·as)
Last word in tokenizer should be same as truncated word: reflection
What is wordcount here: 102
What is truncated word count: 2((such·as) 8 chars
Value of temp:reflection
CURRENT LENGTH of line: 8
Qualified for extra padding
8 extra padding between:(such·as
It will now process truncated string with extra 8 padding between the words:(such·as
Your line getting bigger:(such(5 chars inc white space)
Your line getting bigger:(such·····as(16 chars inc white space)

NEW LENGTH of line after formatting: 16
Completed line:(such·····as
This word will be carried over to next line:reflection

Your line getting bigger:reflection·and(14 chars inc white space)
Words after truncation: 2
Your line getting bigger:reflection·and·runtime(22 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):reflection·and
Length rolled back line: 14
This word getting truncated:runtime
The current buffer is: 2
Number words to accomodate for: 2(reflection·and)
Last word in tokenizer should be same as truncated word: runtime
What is wordcount here: 104
What is truncated word count: 2(reflection·and) 14 chars
Value of temp:runtime
CURRENT LENGTH of line: 14
Qualified for extra padding
2 extra padding between:reflection·and
It will now process truncated string with extra 2 padding between the words:reflection·and
Your line getting bigger:reflection(10 chars inc white space)
Your line getting bigger:reflection···and(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:reflection···and
This word will be carried over to next line:runtime

Your line getting bigger:runtime·code(12 chars inc white space)
Words after truncation: 2
Your line getting bigger:runtime·code·modification)(26 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):runtime·code
Length rolled back line: 12
This word getting truncated:modification)
The current buffer is: 4
Number words to accomodate for: 2(runtime·code)
Last word in tokenizer should be same as truncated word: modification)
What is wordcount here: 106
What is truncated word count: 2(runtime·code) 12 chars
Value of temp:modification)
CURRENT LENGTH of line: 12
Qualified for extra padding
4 extra padding between:runtime·code
It will now process truncated string with extra 4 padding between the words:runtime·code
Your line getting bigger:runtime(7 chars inc white space)
Your line getting bigger:runtime·····code(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:runtime·····code
This word will be carried over to next line:modification)

Your line getting bigger:modification)·that(18 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):modification)
Length rolled back line: 13
This word getting truncated:that
The current buffer is: 3
Number words to accomodate for: 1(modification))

Last word in tokenizer should be same as truncated word: that
What is wordcount here: 107
What is truncated word count: 1(modification)) 13 chars
Value of temp:that
modification)   =>qualifies for 3 padding at end since it has:1 words
CURRENT LENGTH of line: 13
NEW LENGTH of line after formatting: 16
Completed line:modification)···
This word will be carried over to next line:that

Your line getting bigger:that·are(8 chars inc white space)
Words after truncation: 2
Your line getting bigger:that·are·typically(18 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):that·are
Length rolled back line: 8
This word getting truncated:typically
The current buffer is: 8
Number words to accomodate for: 2(that·are)
Last word in tokenizer should be same as truncated word: typically
What is wordcount here: 109
What is truncated word count: 2(that·are) 8 chars
Value of temp:typically
CURRENT LENGTH of line: 8
Qualified for extra padding
8 extra padding between:that·are
It will now process truncated string with extra 8 padding between the words:that·are
Your line getting bigger:that(4 chars inc white space)
Your line getting bigger:that·····are(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:that·····are
This word will be carried over to next line:typically

Your line getting bigger:typically·not(13 chars inc white space)
Words after truncation: 2
Your line getting bigger:typically·not·available(23 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):typically·not
Length rolled back line: 13
This word getting truncated:available
The current buffer is: 3
Number words to accomodate for: 2(typically·not)
Last word in tokenizer should be same as truncated word: available
What is wordcount here: 111
What is truncated word count: 2(typically·not) 13 chars
Value of temp:available
CURRENT LENGTH of line: 13
Qualified for extra padding
3 extra padding between:typically·not
It will now process truncated string with extra 3 padding between the words:typically·not
Your line getting bigger:typically(9 chars inc white space)
Your line getting bigger:typically····not(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:typically····not
This word will be carried over to next line:available

Your line getting bigger:available·in(12 chars inc white space)

Words after truncation: 2
Your line getting bigger:available·in·traditional(24 chars inc white space)
Words after truncation: 3
Rolled back to(due to exceeding 16):available·in
Length rolled back line: 12
This word getting truncated:traditional
The current buffer is: 4
Number words to accomodate for: 2(available·in)
Last word in tokenizer should be same as truncated word: traditional
What is wordcount here: 113
What is truncated word count: 2(available·in) 12 chars
Value of temp:traditional
CURRENT LENGTH of line: 12
Qualified for extra padding
4 extra padding between:available·in
It will now process truncated string with extra 4 padding between the words:available·in
Your line getting bigger:available(9 chars inc white space)
Your line getting bigger:available·····in(16 chars inc white space)
NEW LENGTH of line after formatting: 16
Completed line:available·····in
This word will be carried over to next line:traditional

Your line getting bigger:traditional·compiled(20 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):traditional
Length rolled back line: 11
This word getting truncated:compiled
The current buffer is: 5
Number words to accomodate for: 1(traditional)
Last word in tokenizer should be same as truncated word: compiled
What is wordcount here: 114
What is truncated word count: 1(traditional) 11 chars
Value of temp:compiled
traditional   =>qualifies for 5 padding at end since it has:1 words
CURRENT LENGTH of line: 11
NEW LENGTH of line after formatting: 16
Completed line:traditional·····
This word will be carried over to next line:compiled

Your line getting bigger:compiled·languages.(19 chars inc white space)
Words after truncation: 2
Rolled back to(due to exceeding 16):compiled
Length rolled back line: 8
This word getting truncated:languages.
The current buffer is: 8
Number words to accomodate for: 1(compiled)
Last word in tokenizer should be same as truncated word: languages.
What is wordcount here: 115
What is truncated word count: 1(compiled) 8 chars
Value of temp:languages.
compiled   =>qualifies for 8 padding at end since it has:1 words
CURRENT LENGTH of line: 8
NEW LENGTH of line after formatting: 16
Completed line:compiled·····
This word will be carried over to next line:languages.

Last token processed:languages.

******Currently in last line:
****
Finally completing the last line:languages.
Left over StringJoiner: 1 word(s)=> languages. (10 chars inc white space)
Total running words: 115
Buffer is: 6
Entering here most likely due to having one word in the line
So the StringBuilder should be empty:
languages. =>qualifies for 6 padding at end since it has:1 word(s)
completed line:languages.·····
NEW LENGTH of line:16

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 16 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

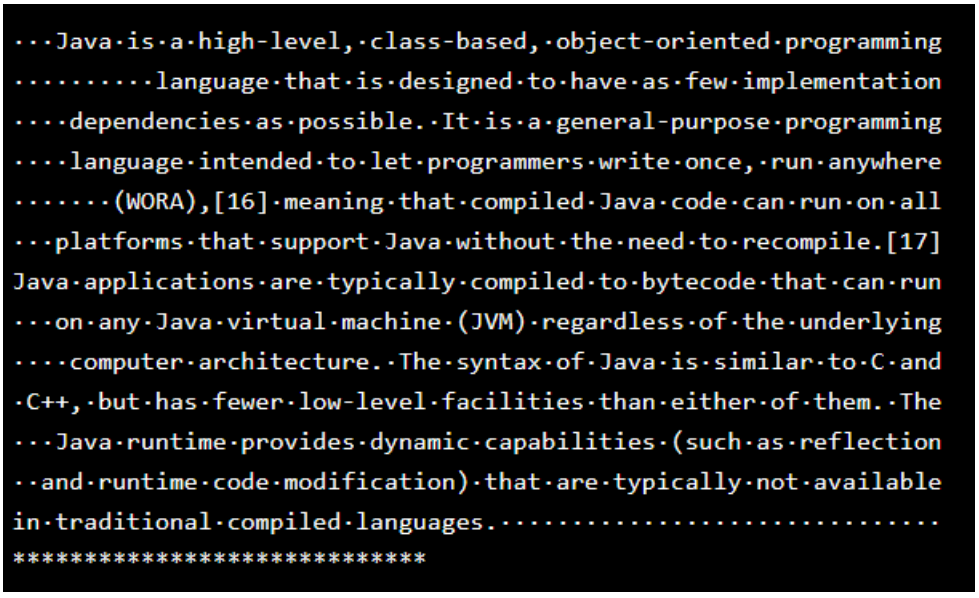
·····Java·is·a
high-level,·····
class-based,····
object-oriented·
programming·····
language·that·is
designed·to·have
as··········few
implementation··
dependencies··as
·possible.·It·is
a··············
general-purpose·
programming·····
language·····
·intended·to·let
programmers·····
·write·once,·run
anywhere·····
(WORA),[16]·····
meaning·····that
compiled····Java
·code·can·run·on
all····platforms
that·····support

Java·without·the
need·····to
recompile.[17]··
Java·····
applications·are
typically·····
compiled·····to
bytecode····that
··can·run·on·any
Java·····virtual
machine····(JVM)
regardless····of
the···underlying
computer·····
architecture.···
···The·syntax·of
·Java·is·similar
to··C··and··C++,
···but·has·fewer
low-level·····
facilities··than
·either·of·them.
The·Java·runtime
provides·dynamic
capabilities····
(such·····as
reflection···and
runtime·····code
modification)···
that·····are
typically····not
available·····in
traditional·····
compiled·····
languages.·····
****

TEST SCENARIO 12 (NEW CODE) - using wider k=65 limit…

****THIS WILL PRINT ENTIRE TEXT**********
KEY: 65 line limit
·=Inter-word padding as per requirements
*=existing frontal padding
In future, can provision for existing inter-word padding for instance multiple whitespaces..
BUT expect user to have single whitespace for the justification to look tidy

Screenshot shows its got correct formatting..



As expected with other font, it can not keep proportion but it can be seen characters are same each line.

···Java·is·a·high-level,·class-based,·object-oriented·programming
··········language·that·is·designed·to·have·as·few·implementation
····dependencies·as·possible.·It·is·a·general-purpose·programming
····language·intended·to·let·programmers·write·once,·run·anywhere

·····(WORA),[16]·meaning·that·compiled·Java·code·can·run·on·all
···platforms·that·support·Java·without·the·need·to·recompile.[17]
Java·applications·are·typically·compiled·to·bytecode·that·can·run
···on·any·Java·virtual·machine·(JVM)·regardless·of·the·underlying
····computer·architecture.·The·syntax·of·Java·is·similar·to·C·and
·C++,·but·has·fewer·low-level·facilities·than·either·of·them.·The
···Java·runtime·provides·dynamic·capabilities·(such·as·reflection
··and·runtime·code·modification)·that·are·typically·not·available
in·traditional·compiled·languages.·····
****