به نام خدا

گزارش پروژه دوم (پاراگراف ها) غزل پوراسفندیار بروجنی (9820453)

بناست پاراگرافی را به صورت رشته و ماکزیمم تعداد کاراکتر که ممکن است در هر خط بیاید را دریافت کنیم و طوری کلمات را در خطوط قرار دهیم که میزان فضای خالی کمترین مقدار باشد. برای این کار با رویکرد برنامه نویسی پویا، با ذکر مثالی پیش می رویم.فرض کنیم رشته ورودی و طول هر خط به صورت زیر باشد:

input String : “she is my friend”

maximum width of line = 6

آرایه ای دو بعدی n*n (که n به تعداد کلمات موجود در پاراگراف می باشد) با نام extras را در نظر میگریم و آن را مطابق زیر پر میکنیم: (در کد برای راحتی کار و اینکه با ایندکس 0 کار نکنیم، سایز آرایه را (n+1)*(n+1) در نظر گرفتیم)(منظور از ”she“ – 6 در واقع ”she“.length() – 6 می باشد.)

6 – “she”	6 – “she is”	6 – “she is my”	6 – “she is my friend”
	6 – “is”	6 – “is my”	6 – “is my friend”
		6 – “my”	6 – “my friend”
			6 – “friend”

باز نویسی آرایه به این صورت خواهد شد:

3	0	-3	-10
	4	1	-6
		4	-3
			0

درایه های غیر منفی آرایه را به توان دو می رسانیم تا آرایه costOfLine تشکیل شود:

9	0	∞	∞
	16	1	∞
		16	∞
			0

برای مثال درایه costOfLint[i][j] هزینه خطی شامل کلمات i تا j را شامل می شود.

آرایه تک بعدی totalCost را تشکیل میدهیم به طوری که مطابق فرمول زیر پر شود:

totalCost[j] = totalCost[i-1] + costOfLine[i][j]

(البته به شرطی که totalCost[i-1] و costOfLine[i][j] مقدار بی نهایت نداشته باشند و totalCost[i-1] + costOfLine[i][j] < totalCost[j] باشد.)

پیچیدگی زمانی الگوریتم:

پیچیدگی زمانی این الگوریتم از مرتبه $O(n^2)$ می باشد.(پیمایش ها روی آرایه دو بعدی است.)

مثالی از ورودی برنامه :

```
Input String : she is my friend
Maximum width of each line : 6
```

مثالی از خروجی برنامه :

```
Line number 1: From word no. 1 to 1
Line number 2: From word no. 2 to 3
Line number 3: From word no. 4 to 4
```

که به این معناست در خط شماره 1 کلمه اول، در خط شماره 2 کلمه دوم و سوم و در خط شماره 3 کلمه چهارم قرار خواهد گرفت. یعنی به این صورت خواهد بود:

```
she
is my
friend
```