به نام خدا

علی افشار دگرسی – 9826653

پروژه 2 – پاراگراف ها

ایده اصلی این است که بیاییم و از دو ارایه dp و ans استفاده کنیم ، به این صورت که dp[i] برابر مینیمم هزینه وقتی است که از i امین کلمه شروع به چینش کنیم و ans[i] نیز برابر شماره اخرین کلمه ای در خطی میشود که i امین کلمه اولین کلمه آن خط است.

فرض کنید برای هر خط l اولین کلمه در آن خط دراندکس i در words باشد. حداقل هزینه آن خط در dp [i] ذخیره می شود. آخرین کلمه در آن خط در ایندکس j در words است ، و j می تواند از i تا n متفاوت باشد. تمام مقادیر j را بررسی می کنیم و تعداد کاراکترهای اضافه شده در خط l را در یک متغیر ذخیره می کنیم. اگر تعداد کاراکترها کمتر از m باشد ، با این تعداد کاراکتر هزینه خط l را پیدا می کنیم. این هزینه را با حداقل هزینه ای که تاکنون برای این خط پیدا کرده ایم یعنی dp [i] مقایسه می کنیم و dp [i] و ans [i] را به ترتیب به روز می کنیم. روش بالا را برای هر مقدار i تکرار کرده و در نهایت کلمات شروع و پایان هر سطر در ایندکس i و ایندس ans [i] آرایه words ذخیره میشود.

پیچیدگی زمانی برنامه از O(n^2) و پیچیدگی حافظه آن O(n) است.

```java
public static void wordingWrap(int[] wordsLength, int m, String[] words) {
    int currentLenUsage;
    int cost;
    int[] dp = new int[wordsLength.length];
    int[] ans = new int[wordsLength.length];
    dp[wordsLength.length - 1] = 0;
    ans[wordsLength.length - 1] = wordsLength.length - 1;
    for (int i = wordsLength.length - 2; i >= 0; i--) {
        currentLenUsage = -1;
        dp[i] = Integer.MAX_VALUE;
        for (int j = i; j < wordsLength.length; j++) {
            currentLenUsage += (wordsLength[j] + 1);
            if (currentLenUsage > m) {
                break;
            }
            if (j == wordsLength.length - 1) {
                cost = 0;
            } else {
                cost = (m - currentLenUsage) * (m - currentLenUsage)  * (m - currentLenUsage)  +
                        dp[j + 1];
            }
            if (cost < dp[i]) {
                dp[i] = cost;
                ans[i] = j;
            }
        }
    }
    int indexOfWord = 0;
    int indexOfLine = 1;
    while (indexOfWord < wordsLength.length) {
        System.out.println("Line " + indexOfLine++ + " : "
                + Arrays.toString(Arrays.copyOfRange(words, indexOfWord, to: ans[indexOfWord] + 1)));
        indexOfWord = ans[indexOfWord] + 1;
    }
    System.out.println("TOTAL COST : " + dp[0]);
}
```

```
enter number of words u want to enter :
5
ali ba ma rafte ast
enter words :
enter M :
6
Line 1 : [ali]
Line 2 : [ba, ma]
Line 3 : [rafte]
Line 4 : [ast]
TOTAL COST : 29

Process finished with exit code 0
```