Pembahasan Ayo Cari SOFITA!

Deskripsi Singkat

Terdapat sebuah barisan huruf kapital sepanjang N dan X_i adalah huruf ke-i pada barisan tersebut. Berapa jumlah huruf yang perlu dicoret Sofita agar ia mendapatkan *subsequence* 'S O F I T A' sebanyak mungkin?

Ide

1. Untuk mencari jumlah huruf yang perlu dicoret dengan efisien, program harus mencari jumlah *subsequence* 'S O F I T A' yang dapat terbentuk terlebih dahulu.

2. Untuk membuat program dengan kompleksitas sesederhana mungkin, maka program tidak boleh memiliki terlalu banyak *loop.*

3. Untuk membuat program dengan 1 *loop* saja, idenya adalah dengan menyimpan jumlah huruf 'S', 'O', 'F', 'I', 'T', 'A' yang ditemukan ke dalam variabel yang sesuai. Ketentuan yang perlu diperhatikan pada tahap ini adalah keenam karakter pada subsequence 'S O F I T A' harus ditemukan secara berurutan.

4. Buatlah iterasi dari 1 sampai N dan lakukan pengecekan pada setiap iterasi yang dijalankan.

5. Perhatikan bahwa index pada barisan akan naik. Oleh karena itu, ketika X_i merupakan salah satu huruf anggota 'S O F I T A' dan X_i dimisalkan sebagai Y_j dengan Y = ['S', 'O', 'F', 'I', 'T', 'A'], maka kita hanya perlu memeriksa apakah masih tersedia huruf Y_{j-1} yang belum menjadi bagian dari *subsequence* 'S O F I T A' yang telah atau sedang dibentuk.

6. Misalnya, huruf pada barisan index ke-i ($X_i = Y_j$) adalah 'T', maka kita perlu memeriksa jumlah huruf 'I' (Y_{j-1}) pada index 1 hingga i-1 barisan yang belum terpakai pada *subsequence* 'S O F I T A' yang lain.

7. Jika X_i bukan merupakan anggota *subsequence* 'S O F I T A', maka kita abaikan.

8. Total huruf yang perlu dicoret adalah N - (jumlah **maksimum** *subsequence* 'S O F I T A' yang dapat dibentuk dikali 6).

Berikut ini adalah potongan kode *method* `getTotalDeletedLetters()` dengan kompleksitas $O(N)$.

```
static int getTotalDeletedLetters(int N, char[] x) {
   int S, O, F, I, T, A;
   S = O = F = I = T = A = 0;

   for (int i = 0; i < N; ++i) {
       if (x[i] == 'S') {
           S++;
       } else if (x[i] == 'O' && S != 0) {
           S--;
           O++;
       } else if (x[i] == 'F' && O != 0) {
           O--;
           F++;
       } else if (x[i] == 'I' && F != 0) {
           F--;
           I++;
       } else if (x[i] == 'T' && I != 0) {
           I--;
           T++;
       } else if (x[i] == 'A' && T != 0) {
           T--;
           A++;
       }
   }

   int total = N - A * 6;
   return total;
}
```