زهرا امان ۹۹۳۱۰۰۶

۱- آرایه A را به صورت $A=[۷,۱۰,۲۵,۱۵,۲۵]$ در نظر می‌گیریم.

$dp[i][j]$ یعنی کمترین تعداد عملیات لازم برای ضرب ماتریس های شماره‌ی i ام تا شماره‌ی j ام. در نتیجه در مثال ما $dp[1][4]$ جواب مسأله است.

$$dp[i][j] = \min\Big(dp[i][k] + dp[k+1][j] + A[i-1] \times A[k] \times A[j]$$

for k in i to j-1 $\Big)$

	۱	۲	۳	۴
۱	۰	۱۷۵۰	۴۳۷۵	۷۰۰۰
۲	-۱	۰	۳۷۵	۷۵۰۰
۳	-۱	-۱	۰	۹۳۷۵
۴	-۱	-۱	-۱	۰

→ ۷۰۰۰ جواب مسأله

۲- بزرگترین زیردنباله مشترک صعودی را LCIS می‌نامیم.

$dp[j]$ یعنی طول LCIS که با المان j ام آرایه B تمام می‌شود.

ما دو آرایه A و B را در ابتدا داریم.

```
for i in n:
    now = 0
    for j in m:
        if A[i] = B[j] and now+1 > dp[j]:
            dp[j] = now+1
        if A[i] > B[j] and dp[j] > now:
            now = dp[j]
```

ans = max dp[j] for j in m

فرهاد امان ۹۹۴۱۰۰۶

۲ - ب) همانطور که در الگوریتم dp برای حل مسأله LCS می بینیم

نیاز داریم که یک جدول n×m را پر کنیم. و جواب

نهایی dp[m][n] است size(A)=m size(B)=n

اگر به رابطه ی بازگشتی دقت کنیم هر بار تنها نیاز به سطر فعلی یا سطر

قبلی جدول داریم.

```
if A[i] = B[j]:
    dp[i][j] = dp[i-1][j-1] + 1
else:
    dp[i][j] = max(dp[i][j-1], dp[i-1][j])
```

با توجه به اینکه در هر مرحله تنها نیاز به ۲ سطر از جدول داریم می توانیم سطرهای مورد نیاز را در همان ۲ سطر ذخیره کرده و باقی سطرها را دور بریزیم.

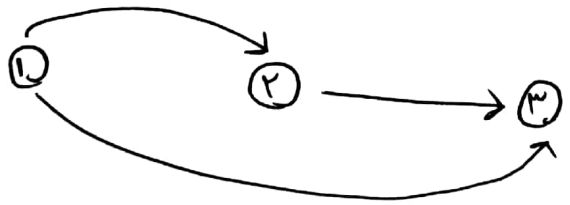
فرهاد امان ۹۹۲۱۰۰۶

۳ - چون یک DAG داریم [redacted] می‌توان یک topological sort برای گراف پیدا کرد حالا در ابتدای کار فاصله‌ی [redacted] هر رأس را برابر ∞ - قرار می‌دهیم و تنها فاصله s را ۰ قرار می‌دهیم.

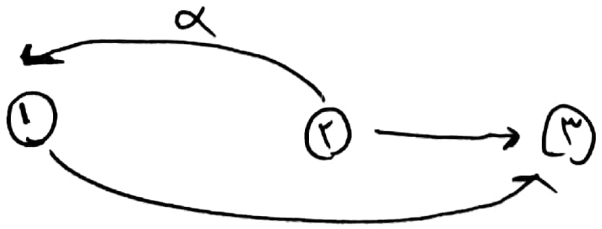
```
for u in topol order:
    for v in adj u:
        if 'dis[v] < dis[u] + w[u][v]:
            dis[v] = dis[u] + w[u][v]:
```

بعد از انجام این عملیات طول بلندترین مسیر از رأس s به هر رأس گراف را داریم.

تعریف topological sort : نوعی قرار دادن رأس های گراف است به طوری که جهت تمام یال های خروجی از هر رأس به سمت راست باشد.



✓



✗

فرهاد اسلامی ۹۹۳۱۰۶

۴- $dp[i][j]$ را اینگونه تعریف می‌کنیم که کمترین هزینه مورد نیاز برای تبدیل زیررشته $A[0 \text{ to } i]$ به $B[0 \text{ to } j]$ باشد $dp[i][j] = \max(i, j)$

واضح است که اگر یکی از نایان برابر ۰ باشد

اگر $A[i] = B[j]$ باشد کافیست $A[0 \text{ to } i-1]$ را به $B[0 \text{ to } j-1]$ برسانیم پس در این حالت $dp[i][j] = dp[i-1][j-1]$

اگر $A[i] \neq B[j]$ یک راه این است که $A[0 \text{ to } i-1]$ به $A[0 \text{ to } i]$ تبدیل شود و عنصر آخر A حذف شود یا اینکه $A[0 \text{ to } i]$ به $B[0 \text{ to } j]$ رسانده شود و عنصر آخر B به آن اضافه شود و درنهایت اینکه $A[0 \text{ to } i-1]$ به $B[0 \text{ to } j-1]$ رسانده شود و عنصر آخر A با عنصر آخر B تعویض شود پس در این حالت $A[0 \text{ to } i-1]$ به $B[0 \text{ to } j-1]$ رسانده شده و عنصر آخر

$$dp[i][j] = 1 + \min\left(dp[i-1][j], dp[i][j-1], dp[i-1][j-1]\right)$$

فرهاد امان ۹۹۴۱۰۰۶

۵ - dp[i][j] برابر طول بلندترین برجی است که در ساخت آن از مکعب نگین تر استفاده شده و رنگ بالای آن ن است.

جواب مسأله برابر max(dp[n][j] for j in all colors)

ابتدا قرار می دهیم dp[i][j] = dp[i-1][j] for j in colors است.

پس قرار می دهیم

```
for side in cube sides:
    dp[i][side] = max(dp[i-1][side], dp[i-1][opposite]+1)
```

opposite در واقع بیت مقابل side در مکعب ماست.