Zadanie: FIL
Ścieżki

polish

BOI 2015, dzień 2. Dostępna pamięć: 256 MB. *1.05.2015*

Bajtazar uwielbia życie na krawędzi: zamiast łatać dziury bezpieczeństwa swoich systemów, blokuje IP hakerów; wysyła rozwiązania zadań konkursowych bez testowania ich na przykładowych danych; a nade wszystko lubi, aby każdy plik na jego komputerze miał tak długą ścieżkę katalogową, na jaką pozwala system operacyjny (w systemie Linux, na przykład, limit długości ścieżki wynosi 4095 znaków).

Kiedy Bajtazar pracuje na komputerze należącym do kogoś innego, zazwyczaj nie wszystkie pliki spełniają powyższy warunek. Wtedy zaczyna on tworzyć dowiązania symboliczne (*symlinki*) i odwoływać się do plików za ich pośrednictwem. Znając nazwy i położenie plików w systemie, sprawdź dla każdego z nich, czy Bajtazar może wygenerować jednego symlinka (o ustalonej wcześniej długości) takiego, żeby do tego pliku dało się odwołać poprzez ścieżkę o długości równej k.

Dla katalogu `kat1`, który zawiera katalog `kat2`, który zawiera katalog `kat3` itd. aż do katalogu `katN`, w którym znajduje się plik o nazwie `plik`, *ścieżka bezwzględna* do tego pliku to `/kat1/kat2/.../katN/plik`. Katalog główny systemu plików oznaczony jest przez `/`. Dla każdego pliku zawartego w katalogu głównym jego ścieżka bezwzględna ma postać `/plik`.

Dowiązanie symboliczne (czyli symlink) to nazwany skrót do katalogu, który może być umieszczony w dowolnym katalogu w systemie plików. **W tym zadaniu symlinki do plików są niedozwolone**. Używając symlinków możemy wyznaczać *alternatywne* ścieżki plikowe. Na przykład, jeśli w katalogu `/` umieścimy symlink nazwany `hello` prowadzący do `/`, wtedy ścieżki `/kat/plik`, `/hello/kat/plik` i `/hello/hello/kat/plik` odnoszą się do tego samego pliku, ale mają różne długości.

Jeśli natomiast umieścimy symlink nazwany `hi` prowadzący do `/` w katalogu `/kat`, to ścieżki `/kat/plik`, `/kat/hi/kat/plik` i `/kat/hi/kat/hi/kat/plik` odnoszą się do tego samego pliku.

Symlinki mogą prowadzić zarówno w górę i w dół w hierarchii katalogów, a także w bok (do innego poddrzewa systemu plików). W szczególności, symlink może prowadzić do katalogu, w którym się znajduje. Na potrzeby tego zadania, ścieżki zawierające odwołania do bieżącego katalogu poprzez `./`, ścieżki zawierające odwołania do katalogu nadrzędnego poprzez `../` oraz ścieżki zawierające `//` nie są dozwolone.

Wejście

Pierwszy wiersz wejścia zawiera trzy dodatnie liczby całkowite: n (liczba katalogów w systemie plików, bez uwzględnienia katalogu głównego), m (liczba plików) oraz k (żądana długość ścieżki katalogowej). Katalog główny reprezentowany jest liczbą 0, pozostałe katalogi numerowane są od 1 do n. Pliki są ponumerowane od 1 do m. Drugi wiersz wejścia zawiera liczbę s, oznaczającą długość nazwy symlinka, który generuje Bajtazar. Nie przejmujemy się samą nazwą tego symlinka – można założyć, że da się go nazwać tak, aby nie kolidował z żadną inną nazwą używaną w systemie plików.

W kolejnych n wierszach wejścia znajdują się opisy katalogów (innych, niż katalog główny) w systemie plików. W i-tym z tych wierszy znajdują się dwie liczby całkowite p_i oraz l_i, oznaczające (odpowiednio) że katalog o numerze i znajduje się w katalogu o numerze p_i oraz że nazwa tego katalogu składa się z l_i znaków. Dla każdego i zachodzi $p_i < i$.

W kolejnych m wierszach znajdują się opisy plików. W j-tym z tych wierszy znajdują się dwie liczby całkowite p_j oraz l_j, oznaczające (odpowiednio), że plik o numerze j znajduje się w katalogu o numerze p_j oraz że nazwa tego pliku składa się z l_j znaków.

Wszystkie pliki i katalogi mają nazwy dodatniej długości, a ich ścieżki bezwzględne nie przekraczają k-znakowego limitu na długość ścieżki katalogowej w systemie plików.

Wyjście

Twój program powinien wypisać m wierszy, po jednym dla każdego pliku. W j-tym z tych wierszy należy wypisać `YES` jeśli jest możliwe wygenerowanie symlinka o nazwie długości s takiego, żeby do j-tego pliku można było utworzyć ścieżkę katalogową o długości dokładnie k, a w przeciwnym przypadku należy wypisać `NO`.

Przykłady

Dla danych wejściowych:	poprawnym wynikiem jest:

`2 4 22`
`2`
`0 1`
`1 5`
`2 13`
`2 10`
`1 4`
`0 7`
	`YES`
	`YES`
	`YES`
	`NO`

Wyjaśnienie do przykładu: Przyjmijmy, że (odpowiednio) nazwa symlinka to `LL`, nazwy katalogów to `a` i `bbbbb`, natomiast nazwy plików to `ccccccccccccc`, `ddddddddddd`, `eeee` i `fffffff`. Katalog główny zawiera katalog `a` i plik `fffffff`. Katalog `a` zawiera katalog `bbbbb` i plik `eeee`. Katalog `bbbbb` zawiera pliki `ccccccccccccc` i `ddddddddddd`.

```
/
|-- a
|   |-- bbbbb
|   |   |-- ccccccccccccc
|   |   +-- ddddddddddd
|   +-- eeee
+-- fffffff
```

W pierwszym przypadku ścieżka bezwzględna `/a/bbbbb/ccccccccccccc` jest już maksymalnej długości, więc nie potrzebujemy symlinka. W drugim przypadku możemy wygenerować dowiązanie `/a/LL -> /a`, co daje możliwą ścieżkę `/a/LL/bbbbb/ddddddddddd`. W trzecim przypadku symlink generujemy jako `/a/LL -> /` i odwołujemy się przez ścieżkę `/a/LL/a/LL/a/LL/a/eeee`. W czwartym przypadku nie da się tak utworzyć symlinka, żeby odwołanie do pliku `fffffff` było wymaganej długości.

Ocenianie

We wszystkich podzadaniach $1 \leq k, s \leq 1\,000\,000$.

Podzadanie	Ograniczenia	Punkty
1	$n, m \leq 500$	33
2	$n, m \leq 3000$, dla każdego pliku, dla którego odpowiedź brzmi `YES` da się utworzyć symlink w taki sposób, że wystarczy nim przejść co najwyżej jeden raz	33
3	$n, m \leq 3000$	34