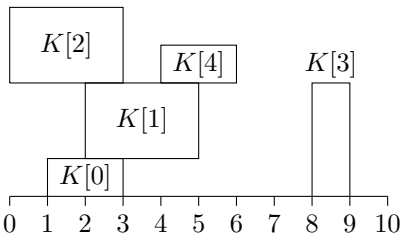
Warunki

1. W implementacjach można korzystać tylko z elementarnych konstrukcji Python'a (funkcje, instrukcje warunkowe, pętle, `range`, klasy użyte do definiowania struktur danych). **Nie wolno korzystać ze słowników i zbiorów, itp. Wolno korzystać z:**
   (a) wbudowanego sortowania,
   (b) wbudowanej kolejki (`deque` z biblioteki `collections`),
   (c) wbudowanej kolejki priorytetowej (`PriorityQueue` z biblioteki `queue`).
2. Rozwiązania muszą być efektywne obliczeniowo (także w zadaniach, w których nie podajemy wprost ograniczenia na złożoność obliczeniową). Zadania o zbyt wysokiej złożoności będą otrzymywały obniżone oceny (lub 0).
3. Rozwiązania zadań proszę umieszczać w załączonych plikach (`zadX.py`).

Zadanie 1 (spadające klocki)

Dana jest tablica klocków $K[0], \ldots, K[n-1]$. Każdy klocek $K[i]$ to krotka (a, b, c), która mówi że klocek zaczyna się na pozycji a, ciągnie się do pozycji b (wszystkie pozycje to nieujemne liczby naturalne) oraz ma wysokość c. Można założyć, że $a < b$ oraz $c \geqslant 1$, oraz że a, b, c to liczby naturalne. Klocki układane są po kolei. Jeśli klocek nachodzi na któryś z poprzednich, to jest przymocowywany na szczycie poprzedzającego klocka. Na przykład dla klocków opisanych przez $(1, 3, 1)$, $(2, 5, 2)$, $(0, 3, 2)$, $(8, 9, 3)$, $(4, 6, 1)$ powstaje konstrukcja o wysokości 5:



Proszę zaimplementować funkcję:

```
def blocks_height( K )
```

która zwraca wysokość konstrukcji. Funkcja powinna być możliwie jak najszybsza.