Szablon rozwiązania	`egzP7a.py`
Złożoność akceptowalna (3 pkt):	$O(n^4)$
Złożoność wzorcowa (+1 pkt):	$O(n^2)$, gdzie **n** to liczba pokojów w akademiku

W okresie wakacyjnym Miasteczko Studenckie AGH zainwestowało w remont jednego z domów studenckich. W wyniku tego remontu jego standard został podwyższony do Komfort++, gdyż wszystkie pokoje w nim są jednoosobowe. Ilość chętnych była dużo większa niż ilość dostępnych miejsc, więc o kwalifikacji zadecydowała odległość miejsca zamieszkania od uczelni. Studenci są dość wybredni i chcieliby mieszkać w konkretnych pokojach, dlatego administracja umożliwiła każdemu z nich wskazanie preferencji tj. podanie maksymalnie trzech numerów pokoi, które chcieliby zamieszkiwać. Część z nich nie podała żadnych preferencji, gdyż stwierdziła, że będzie zadowolona niezależnie od pokoju, do którego trafi. Oczywiście administracji zależy na tym, aby każdy był zadowolony, więc zrobili wszystko co w ich mocy, aby jak najwięcej studentów mieszkało w wybranych przez siebie w pokojach. Osoby których preferencje nie zostały wzięte pod uwagę w ramach zadośćuczynienia otrzymają darmowy karnet na basen Akademii Górniczo-Hutniczej. Ile karnetów zostanie wydanych studentom?

Zadanie polega na zaimplementowaniu funkcji:

```
akademik( T )
```

która zwróci ilość karnetów, która zostanie wydana studentom. Tablica **T** zawiera preferencje wszystkich studentów wyrażone w postaci krotek (p_1, p_2, p_3) gdzie p_i jest numerem pokoju. Przykładowo, dla studenta który chciałby mieszkać w pokoju 500 lub 501, krotka przyjmie wartość (500, 501, None). Numeracja pokoi zaczyna się od pokoju 0.

Rozważmy następujące dane:

```
T = [(2, 3, None), (0, 1, 3), (0, 2, None), (1, 3, 4), (2, 3, None)]
```

Wywołanie funkcji `akademik( T )` powinno zwrócić wynik **0**. W przypadku dopasowania pokoi jako kolejno: **2** – **1** – **0** – **4** – **3** wszyscy będą zadowoleni, więc nikt nie otrzyma karnetu.

Szablon rozwiązania	`egzP7b.py`
Złożoność akceptowalna (1.5 pkt):	$O(n^2)$
Złożoność wzorcowa (+2.5 pkt):	O(nlogn), gdzie **n** to liczba drzew w ogrodzie

Juliusz posadził w linii prostej n drzew owocowych, a każde drzewo jest jednego z m gatunków. Jego syn Brutus bardzo chciał zjeść jakieś owoce z sadu, jednakże wiedział, że jeżeli ojciec go na tym przyłapie, to dostanie szlaban na zadanka z Algebry, w związku z tym obmyślił plan. Plan był nie byle jaki, bo był sprytny. Brutus powiedział ojcu, że idzie susza i że trzeba jak najszybciej sprzedać co się da. Juliusz jako że nie chciał narażać się na straty, wsiadł jak najszybciej na traktor i stwierdził że zbierze ile się da gatunków owoców, ale albo wszystkie owoce muszą być z jednego drzewa albo nie zbiera ich w ogóle. Dodatkowo żeby szybko dotrzeć na targ, zbierze owoce tylko z spójnego fragmentu drzew. Brutus ucieszony tym, że ojca nie będzie w domu przez najbliższe kilka godzin zaczął zastanawiać się, ile pieniędzy przyniesie jego sabotaż.

Zadanie polega na zaimplementowaniu funkcji:

```
ogrod( S, V )
```

która zwróci przychód z sabotażu, przy następujących założeniach:

1. Tablica **S** określa gatunki kolejnych drzew i ma postać $[s_0, s_1, \ldots, s_{n-1}]$ gdzie s_i oznacza gatunek i-tego drzewa. Wartości w tablicy **S** zawierają się w przedziale domkniętym <1, m>

2. Tablica **V** określa wartości gatunków wyrażone w i ma postać $[v_0, v_1, \ldots, v_{m-1}]$ gdzie v_i oznacza wartość w monetach i-tego gatunku.

Rozważmy następujące dane:

```
S = [2, 3, 1, 1, 4, 1, 2, 4, 1]
V = [5, 3, 6, 6]
```

Wywołanie funkcji `ogrod( S, V )` powinno zwrócić wynik **15.** Interesujący Brutusa fragment to od drugiego do siódmego drzewa. Wtedy zbierze owoce z drzew gatunku 2, 3 oraz 4, a owoce z gatunku 1 pominie. Otrzyma w ten sposób **3** + **6** + **6** czyli **15** monet.