Szablon rozwiązania	`egzP8b.py`
Złożoność akceptowalna (1.5pkt):	$O(n^3 \log n)$
Złożoność wzorcowa (+2.5pkt):	$O(n^3)$, gdzie **n** to liczba punktów na hali

Pewna duża amerykańska firma technologiczna postanowiła zorganizować wyścigi autonomicznych robotów. Wyścigi odbyły się na potężnej prostokątnej hali, która dla tego celu została w pełni opróżniona. Na podłodze zostały umieszczone naklejki w dwóch kolorach. Białe oznaczają punkty zmiany kierunku, a szare to punkty kontrolne (które oczywiście jednocześnie pełnią również role punktów białych). Roboty wyposażone są w kamery oraz dobre oświetlenie, co pozwala im rozpoznać, że znajdują się na punkcie danego koloru. Ze względu na zastosowane technologie, autonomiczne roboty po najechaniu na punkt, wybierają kolejny punkt swojej trasy, a następnie zmieniają swój kierunek oraz decydują o odległości, którą przebędą (tj. jeżeli określą następny punkt, to wszystkie punkty na trasie do niego zostaną zignorowane). Między dwoma punktami zawsze poruszają się w linii prostej. W trakcie trasy muszą odwiedzić one wszystkie punkty kontrolne w określonej przez organizatora kolejności, przebywając łącznie jak najkrótszy dystans. Rozwiązanie zadania byłoby oczywiste, gdyby nie to, że na hali zostały rozmieszczone przeszkody. Do pamięci robota została wgrana przygotowana mapa w postaci grafu ważonego i musi zdecydować on o kolejności odwiedzenia punktów.

Zadanie polega na zaimplementowaniu funkcji:

```
def robot( G, P )
```

która oblicza dystans, który pokona robot, przy następujących założeniach.

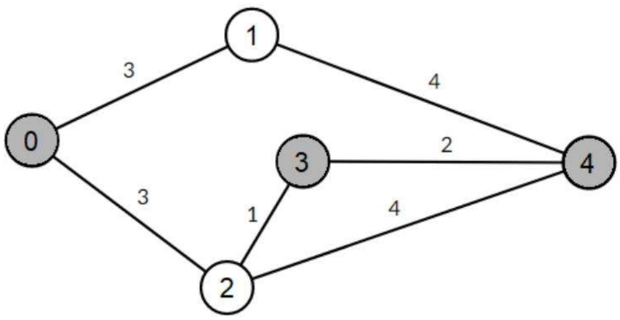
1. Graf ważony **G** jest wyrażony jako lista sąsiedztwa, dla każdych dwóch punktów **u** oraz **v**, pomiędzy którymi nie znajduje się żadna przeszkoda, oddalonych od siebie o **x cm**, lista **G[u]** będzie zawierała krotkę ***(v, x)*** oraz lista G[v] będzie zawierała krotkę ***(u, x)***

2. Tablica **P** zawiera listę szarych punktów, ich numeracja odpowiada wierzchołkom z grafu **G**, w kolejności, w której muszą zostać odwiedzone. Robot zaczyna swoją trasę w pierwszym punkcie, a kończy w ostatnim. Wierzchołki w tablicy **P** są unikalne.

3. Można założyć, że przeszkody zostały umieszczone w taki sposób, że dana przeszkoda wyklucza poruszanie się tylko między jedną parą punktów na hali, a także, że ich ilość jest marginalna względem ilości wszystkich możliwych par punktów.

Rozważmy następujące dane:

```
G = [
  [(1, 3), (2, 3)],
  [(0, 3), (4, 4)],
  [(0, 3), (3, 1), (4, 4)],
  [(2, 1), (4, 2)],
  [(1, 4), (2, 4), (3, 2)]
]
P = [0, 3, 4]
```



Wywołanie `robot( G, P )` powinno zwrócić wynik **6** (Odwiedzamy punkty 0 – 2 – 3 – 4)