Szablon rozwiązania:	`kol3b.py`
Złożoność akceptowalna (1.5pkt):	$O(n^2)$, gdzie n to liczba lotnisk.
Złożoność wzorcowa (+2.5pkt):	$O(m \log n)$, gdzie m to liczba dróg (tj. krawędzie w G) a n to liczba lotnisk.

Mapa lotnisk w Bitlandii ma postać grafu nieskierowanego $G = (V, E)$, gdzie wierzchołki to lotniska a E to drogi między lotniskami. Każda droga ma pewien koszt wyrażony w złotówkach (trzeba zapłacić za paliwo). Dodatkowo między dowolnymi dwoma lotniskami można przelecieć szybowcem. Przy takim przelocie nie trzeba płacić za paliwo, ale każde lotnisko ma swoje opłaty (takie same za start i lądowanie).
W ramach zadania należy zaimplementować funkcję:

```
def airports( G, A, s, t )
```

która oblicza najniższy możliwy koszt dotarcia z lotniska s do lotniska t przy następujących założeniach:

1. G zawiera graf reprezentowany listowo (czyli dla każdego lotniska u, `G[u]` to lista par postaci `(v,c)`, oznaczających że mamy krawędź z u do v o koszcie przejazdu c; jeśli istnieje krawędź z u do v o koszcie c, to graf zawiera też krawędź z v do u o tym samym koszcie).
2. A to lista opłat lotniskowych (czyli `A[u]` to opłata na lotnisku u, ponoszona tylko wtedy gdy z tego lotniska startujemy szybowcem lub na nim lądujemy szybowcem).
3. *Wierzchołek s to lotnisko startowe a wierzchołek t to lotnisko docelowe.*

Rozważmy następujące dane:

```
G = [ [(1,3), (3,2)],              # 0
      [(0,3), (2,20)],             # 1
      [(1,20), (5,1), (3,6)],      # 2
      [(0,2), (2,6), (4,1)],       # 3
      [(3,1), (5,7)],              # 4
      [(4,7), (2,1)] ]             # 5

#     0    1  2   3  4   5
A = [50, 100, 1, 20, 2, 70 ]
```

Wywołanie `airports( G, A, 0, 5 )` powinno zwrócić wynik 7: z 0 jedziemy do 3, stamtąd do 4, wsiadamy do szybowca i lecimy do 2, po czym jedziemy do 5. Koszt tej trasy to $2 + 1 + (2 + 1) + 1$ (w nawiasie zawarty jest koszt przelotu).

Podpowiedź. Ile lotów maksymalnie i minimalnie warto rozważać?