Algorytmy i Struktury Danych
Egzamin 2: Zadanie A (5.IX.2022)

Format rozwiązań

Rozwiązanie zadania musi się składać z krótkiego opisu algorytmu (wraz z uzasadnieniem poprawności) oraz jego implementacji. Zarówno opis algorytmu jak i implementacja powinny się znajdować w tym samym pliku Pythona (rozszerzenie `.py`). Opis powinien być na początku pliku w formie komentarza (w pierwszej linii w komentarzu powinno być imię i nazwisko studenta). Opis nie musi być długi—wystarczy kilka zdań, jasno opisujących ideę algorytmu. Implementacja musi być zgodna z szablonem kodu źródłowego dostarczonym wraz z zadaniem. Niedopuszczalne jest w szczególności:

1. zmienianie nazwy funkcji implementującej algorytm, listy jej argumentów, lub nazwy pliku z rozwiązaniem,
2. wypisywanie na ekranie jakichkolwiek napisów innych niż wypisywane przez dostarczony kod (ew. napisy dodane na potrzeby diagnozowania błędów należy usunąć przed wysłaniem zadania).

Dopuszczalne jest natomiast:

1. korzystanie z zaawansowanych struktur danych (np. słowników czy zbiorów),
2. korzystanie z następujących elementarnych struktur danych: krotka, lista, kolejka `collections.deque`, kolejka priorytetowa (`queue.PriorityQueue` lub `heapq`),
3. korzystanie ze struktur danych dostarczonych razem z zadaniem (jeśli takie są),
4. korzystanie z wbudowanych funkcji sortujących (można założyć, że mają złożoność $O(n \log n)$).

Wszystkie inne algorytmy lub struktury danych wymagają implementacji przez studenta. Dopuszczalne jest oczywiście implementowanie dodatkowych funkcji pomocniczych w pliku z szablonem rozwiązania.

Zadania niezgodne z powyższymi ograniczeniami otrzymają ocenę 0 punktów. Rozwiązania w innych formatach (np. `.PDF`, `.DOC`, `.PNG`, `.JPG`) z definicji nie będą sprawdzane i otrzymają ocenę 0 punktów, nawet jeśli będą poprawne.

Testowanie rozwiązań

Żeby przetestować rozwiązanie zadania należy wykonać polecenie: `python egz2b.py`

Szablon rozwiązania:	`egz2b.py`
Złożoność akceptowalna (1.5pkt):	$O(n^2)$, gdzie n to komnat.
Złożoność wzorcowa (+2.5pkt):	$O(n)$, gdzie n to liczba komnat.

Magiczny Wojownik obudził się w komnacie 0 pewnej tajemniczej jaskini, mając w głowie jedynie instrukcje, jakie otrzymał od Złego Maga. Wie, że komnaty są ponumerowane od 0 do $n-1$ i w każdej komnacie znajduje się troje drzwi, z których każde pozwala przejść do komnaty o wyższym numerze (cofnięcie się do komnaty o niższym numerze grozi śmiercią Wojownika; co więcej, niektóre drzwi są zablokowane) oraz skrzynia z pewną liczbą sztabek złota. Wstępnie wszystkie drzwi są zamknięte, ale jeśli w skrzyni zostanie umieszczona odpowiednia liczba sztabek złota, to drzwi się otwierają i można nimi przejść. Z każdej skrzyni można zabrać najwyżej 10 sztabek złota, ale można też w niej zostawić dowolnie wiele sztabek. Na początku Wojownik nie ma ani jednej sztabki a jego celem (na zlecenie Złego Maga) jest dojść do komnaty $n-1$ mając jak najwięcej sztabek.

Zadanie polega na zaimplementowaniu funkcji:

```
def magic( C )
```

która otrzymuje na wejściu tablicę C opisującą jaskinię ($n = |C|$) i zwraca największą liczbę sztabek złota, z którymi Wojownik może dojść do komnaty $n-1$, lub -1 jeśli dotarcie do tej komnaty jest niemożliwe. Opis jaskini jest postaci $C = [R_0, \ldots, R_{n-1}]$, gdzie każde R_i to opis komnaty postaci:

$$[G, [K_0, W_0], [K_1, W_1], [K_2, W_2]]$$

gdzie G to liczba sztabek złota w skrzyni a każda para $[K_i, W_i]$ składa się z liczby K_i sztabek słota potrzebnych do otwarcia drzwi numer i prowadzących do komnaty W_i (gdzie $W_i > i$ lub $W_i = -1$ jeśli za tymi drzwiami jest lita skała i nie da się nimi przejść nawet jeśli się je otworzy). Funkcja powinna być możliwie jak najszybsza.

Przykład. Rozważmy następującą jaskinię:

```
C = [ [8,  [ 6, 3], [ 4, 2], [7, 1]],   # 0
      [22, [12, 2], [21, 3], [0,-1]],   # 1
      [9,  [11, 3], [ 0,-1], [7,-1]],   # 2
      [15, [ 0,-1], [ 1,-1], [0,-1]] ]  # 3
```

Optymalna trasa wojownika to:

1. Wziąć 1 sztabkę złota w komnacie 0 i przejść do komnaty 1.
2. Wziąć 10 sztabek złota w komnacie 1 i przejść do komnaty 2.
3. Zostawić 2 sztabki złota w komnacie 2 i przejść do komnaty 3.

Dzięki temu na koniec wędrówki Wojownik ma 9 sztabek złota.