Algorytmy i Struktury Danych
Zadanie offline 9 (17.VI.2024)

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. korzystanie z wbudowanych funkcji sortujących,
2. korzystanie z zaawansowanych struktur danych (np. słowników czy zbiorów),
3. zmienianie nazwy funkcji implementującej algorytm, listy jej argumentów, lub nazwy pliku z rozwiązaniem,
4. modyfikowanie testów dostarczonych wraz z szablonem,
5. 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 następujących elementarnych struktur danych: krotka, lista, kolejka `collections.deque`, kolejka priorytetowa (`queue.PriorityQueue, heapq`),
2. korzystanie ze struktur danych dostarczonych razem z zadaniem (jeśli takie są).
3. 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: `python3 zad8.py`

Zadanie offline 8.

Szablon rozwiązania: `zad8.py`

Mapa Gór Algorytmicznych to macierz o wymiarach $m \times n$, której każde pole to liczba naturalna stanowiąca wysokość danego obszaru (co ciekawe, każdy obszar ma inną wysokość). Student chce się wybrać na wycieczkę po tym paśmie górskim, ale stawia kilka warunków: Po pierwsze, będąc na danym polu może przejść wyłącznie na inne pole, które dzieli z nim jedną współrzędną (czyli może się poruszać o jedno pole na prawo, lewo, w górą, lub w dół). Po drugie, może przejść wyłącznie na pole o wyższej wysokości (wycieczka ma być wyzwaniem). Po trzecie, jego trasa ma być jak najdłuższa, czyli ma przejść jak najwięcej pól, wliczając w to pole startowe (w końcu im więcej czasu spędzie się w Górach Algorytmiki, tym lepiej). Student może wyruszyć z dowolnego wybranego przez siebie pola (po prostu jakoś się tam znajdzie w sobotę rano; nikt nie wie jak to się dzieje).

Zadanie polega na implementacji funkcji:

```
trip( M )
```

która na wejściu otrzymuje mapę Gór Algorytmicznych `M` i zwraca największą liczbę pól, które może odwiedzić student.

Przykład. Dla wejścia:

```
M = [ [7,6,5,12],
      [8,3,4,11],
      [9,1,2,10] ]
```

wynikiem jest 8 (jedną z tras o tej długości jest $1 \to 3 \to 4 \to 5 \to 6 \to 7 \to 8 \to 9$).

Podpowiedź. To zadanie łączy w sobie materiał z wszystkich części przedmiotu ASD (skojarzenie z acykliczynmi grafami skierowanymi nie jest niezbędne, ale pomaga w zrozumieniu poprawności rozwiązania).