Algorytmy i struktury danych

Lista nr 3

Waga listy: 2

(termin oddania: 4 laboratorium - ostatnie własne zajęcia przed 10 maja)

Zadanie 1 (1 pkt) Napisz program, który implementuje działanie kolejki priorytetowej. Struktura powinna umożliwiać przechowywanie wartości typu `int` wraz z ich priorytetem, będącym liczbą naturalną, przy czym najwyższym priorytetem jest wartość 0. Program nie powinien wymagać żadnych parametrów uruchomienia, a na standardowym wejściu przyjmuje dodatnią liczbę całkowitą m (liczba operacji), a następnie (w liniach 2–$(m+1)$) jedną z operacji:

- insert x p - wstaw do struktury wartość x o priorytecie p,
- empty - wypisz wartość *true* (1) dla pustej struktury, wartość *false* (0) w p.p.
- top - wypisz wartość o najwyższym priorytecie lub pustą linię w przypadku braku elementów w strukturze,
- pop - wypisz wartość o najwyższym priorytecie a następnie usuń ją ze struktury (wypisz pustą linię w przypadku braku elementów w strukturze),
- priority x p - dla każdego elementu o wartości x obecnego w strukturze ustawia priorytet p, jeśli jest on wyższy od aktualnego priorytetu danego elementu,
- print - wypisuje w jednej linii zawartość struktury w postaci (x_i, p_i), gdzie x_i to kolejne wartości przechowywane w kolejce a p_i odpowiadające im priorytety.

Dla liczby elementów w kolejce n, koszt operacji musi wynosić $O(\log n)$, dla wszystkich poleceń z pominięciem print.

Zadanie 2 (1 pkt) Korzystając ze struktury zaimplementowanej w **Zadaniu 1** zaimplementuj program realizujący algorytm Djikstry, dla podanego grafu skierowanego $G = (V, E)$, znajdujący najkrótsze ścieżki z wybranego wierzchołka $v \in V$ do każdego $\tilde{v} \in V$.

Program nie powinien wymagać żadnych parametrów uruchomienia. Po uruchomieniu programu, na standardowym wejściu, podajemy definicję grafu G oraz wierzchołek startowy v. Kolejno wczytywane są:

- liczba wierzchołków $n = |V|$ (przyjmujmy, że wierzchołki są etykietowane kolejnymi liczbami naturalnymi $\{1, \ldots, n\}$)
- liczba krawędzi $m = |E|$ (krawędzie są postaci (u, v, w), gdzie u i v są wierzchołkami a w wagą krawędzi – zakładamy, że wagi są nieujemnymi liczbami rzeczywistymi, ale niekoniecznie spełniona jest nierówność trójkąta, ponadto przyjmujemy iż ścieżka z u do u zawsze istnieje i ma koszt 0
- kolejno m definicji krawędzi w postaci `u v w`
- etykieta wierzchołka startowego.

Na standardowym wyjściu powinno zostać n linii, w formacie `id_celu waga_drogi`, natomiast na standardowym wyjściu błędów powinny być wypisane dokładne ścieżki (tzn. wierzchołki pośrednie i wagi) do każdego z wierzchołków docelowych oraz czas działania programu w milisekundach.

Zadanie 3 (2 pkt) Zaimplementuj algorytmy znajdujące dla podanego grafu nieskierowanego $G = (V, E)$ minimalne drzewa rozpinające, wykorzystując algorytm *Union-Find* i kopiec.

Program powinien umożliwiać wykonanie algorytmu Prima (parametr uruchomienia `-p`) oraz algorytmu Kruskala (parametr uruchomienia `-k`). Niezależnie od parametru uruchomienia, dane wejściowe przyjmują postać:

- liczba wierzchołków $n = |V|$ (przyjmujmy, że wierzchołki są etykietowane kolejnymi liczbami naturalnymi $\{1, \ldots, n\}$)
- liczba krawędzi $m = |E|$ (krawędzie są postaci (u, v, w), gdzie $u \in V$ i $v \in V$ są połączonymi wierzchołkami a w wagą krawędzi – zakładamy, że wagi są nieujemnymi liczbami rzeczywistymi, ale niekoniecznie spełniona jest nierówność trójkąta, naturalnie krawędź z u do u zawsze istnieje i ma koszt 0, ponadto przyjmujemy, że graf jest spójny)
- kolejno m definicji krawędzi w postaci `u v w`

Na standardowym wyjściu powinny zostać wypisane, w kolejnych liniach, krawędzie `u v w` użyte w drzewie rozpinającym (przyjmujemy, że $u < v$) oraz łączna waga drzewa rozpinającego.

Zadanie 4 (1pkt) Zaimplementuj algorytm znajdujący dla podanego grafu skierowanego składowe silnie spójne i wypisujący je w wyliczonym porządku topologicznym.

Na standardowym wejściu podawane są kolejno:

- liczba wierzchołków $n = |V|$ (przyjmujmy, że wierzchołki są etykietowane kolejnymi liczbami naturalnymi $\{1, \ldots, n\}$),
- liczba krawędzi $m = |E|$,
- kolejno m definicji krawędzi w postaci `u v` ((u, v) jest krawędzią skierowaną z u do v).

Na standardowym wyjściu w kolejnych k liniach (k - liczba znalezionych składowych silnie spójnych) powinny zostać wypisane numery wierzchołków z tych składowych, a kolejność wypisanych linii odpowiadać porządkowi topologicznemu w otrzymanym DAG, natomiast na standardowym wyjściu błędów powinien być wypisany czas działania programu w milisekundach.