Контролна работа №2 по ДАА, Информатика, 04.06.2011

Име:..... ФN:..... Група:....

Задача	1	2	3	4	5	Общо
получени точки						
от максимално	*20*	*10*	*10*	*10*	*10*	*60*

Зад. 1 Докажете или опровергайте всяко от следните пет твърдения. Дайте кратка аргументация на отговорите си. Отговори без никаква аргументация не се оценяват.

а) Следната имплементация на познатата ви функция `Heapify` е коректна:

```
Heapify(A, i)
 l ← Left(i)
 r ← Right(i)
 if l ≤ size(A) and A[l] > A[i]
     largest ← l
 if r ≤ size(A) and A[r] > A[i]
     largest ← r
 if largest ≠ i
     swap(A[i], A[largest])
 Heapify(A, largest)
```

б) Съществува имплементация на функцията Heapify, работеща във време $O\left(\frac{\lg n}{\lg \lg n}\right)$.

в) Алгоритъмът Mergesort е стабилен сортиращ алгоритъм.

г) Нека $G(V, E)$ е ориентиран граф. Нека (u, v) е произволно ребро от E, такова че алгоритъмът за обхождане в дълбочина DFS класифицира (u, v) като ребро напред. Тогава задължително u и v са върхове от една и съща силно свързана компонента на G.

д) Следният алгоритъм `ShortPath(G, s, t)` връща дължината на най-лек път между дадени върхове `s` и `t` в произволен свързан тегловен неориентиран граф $G(V, E, w)$:

```
ShortPath(G, s, t)
count ← 0
l е променлива-връх
A и B са множества
l ← s
A ← върховете, съседни на s
B ← {s}
Докато l ≠ t
    Намери най-леко ребро (u, v),
        такова че u ∈ B и v ∈ A
    count ← count + w((u,v))
    B ← B ∪ {v}
    Добави към A всички съседи на v,
        които не са в B
    A ← A \ {v}
    l ← v
return count
```

Всяка от следващите пет задачи иска да се измисли и опише алгоритъм. Описанието на алгоритмите трябва да бъде съвършено ясно, без никакво двусмислие или неясноти. Пишете детайлен

псевдокод или словесно описание, от което да е абсолютно ясно как да се напише програма, имплементираща предложения алгоритъм. Не дискутирайте примери и не разказвайте как бихте решили задачата за конретен пример. Описания на алгоритми, които съдържат неясноти или са прекалено недетайлни или използват конкретни примери няма да бъдат оценявани.

Добре е да се даде аргументация за коректността на всеки описан алгоритъм – колкото по-формална и прецизна, толкова по-добре. Задължително е да се даде оценка на сложността по време на всеки алгоритъм. Колкото по-прецизно бъде обоснована дадена оценка за сложност, толкова по-добре.

Зад. 2 Даден е неориентиран тегловен свързан граф $G(V, E, w)$. Предложете колкото е възможно по-бърз (в асимптотичния смисъл) алгоритъм, който изчислява покриващо дърво T на G, такова че теглото на най-тежкото ребро в T е минимално.
Забележка: Изучаваните алгоритми за минимални покриващи дървета пресмятат дърво, на което общото тегло е минимално. Тук се иска да се намери дърво, в което се минимизира теглото на най-тежкото ребро.

Зад. 3 Предложете алгоритъм, който по дадени n естествени числа, всяко от които е от интервала $[1, \ldots, k]$, прави първоначална обработка (preprocessing) на числата, след което отговаря на запитвания от типа *"Колко числа от дадените са в интервала* $[a, \ldots, b]$*?"*. На всяко запитване трябва да се отговори във време $\Theta(1)$. Първоначалната обработка трябва да бъде извършена във време $\Theta(n + k)$.

Зад. 4 Предложете алгоритъм с колкото е възможно по-малка сложност по време, който при вход неориентиран граф $G(V, E)$ връща Да, ако графът има цикъл с дължина 3, и Не, в противен случай.

Зад. 5 Илюстрирайте работата на алгоритъма QUICKSORT върху следния входен масив: $[4, 1, 3, 8, 2, 5, 7, 9]$. За целта напишете ясно като псевдокод алгоритъма и после покажете колкото можете по-подробно как точно се променя този масив, докато стане сортиран.

Бонус задача - 10 т. Решете задача 4 при допълнителното условие, че G е такъв, че всяко ребро участва в най-много един цикъл.
Упътване: Използвайте модифициран DFS, който класифицира върховете в нива. Намерете необходимо и достатъчно условие да има цикъл с дължина три, което условие е изразено чрез класификацията на ребрата, която правим в контекста на DFS.

Бонус задача - 10 т. Дадено е множество интервали $J = \{j_1, j_2, \ldots, j_n\}$, където всеки интервал j_i е наредена двойка (s_i, f_i) от положителни числа, такива че $s_i < f_i$. Казваме, че два интервала j_i и j_k се пресичат, ако $s_i \leq s_k < f_i$ или $s_k \leq s_i < f_k$. Всеки интервал j_i има цена c_i, която е някакво положително число. Предложете колкото е възможно по-бърз (в асимптотичния смисъл) алгоритъм, който намира подмножество $S \subseteq J$, такова че никои два интервала $x, y \in S$ не се пресичат, и освен това сумата от цените на елементите на S е максимална. Не е необходимо алгоритимът ви да връща S като множество интервали, достатъчно е да пресметне $\sum_{x \in S} c(x)$.
Упътване: Използвайте динамично програмиране. Лакомата статегия не дава оптимален резултат.