Изпит по ДАА (поправителна сесия), 26.08.2015 г.

Име: _____, ФН:_____, Спец.:____, Курс:___

Задача	1	2	3	4	5	6	Общо
получени точки							
максимум точки	20	20	20	40	20	20	140

Забележка: За отлична оценка са достатъчни 100 точки!

Задача 1. Решете следните рекурентни уравнения:

а) $T(n) = 2\sqrt{2}\,T\left(\frac{n}{\sqrt{2}}\right) + n^3$; б) $T(n) = T(n-1) + \frac{1+n}{n^2}$;

в) $T(n) = \sum_{i=0}^{n-1} T(i) + 2^{\frac{n}{2}}$; г) $T(n) = 5\,T\left(\frac{n}{2}\right) + n^2 \lg n$.

Задача 2. Съставете алгоритъм с времева сложност $O(n \,.\, \log k)$, който намира най-малките k елемента в числовия масив $A[1 \ldots n]$, $k < n$ (12 т.). Докажете коректността му (8 т.).

Задача 3. Разполагаме с данни за взаимните задължения между n банки. Ако банката X има задължение към банката Y и Y има задължение към Z, и Z има задължение към X, то те могат да погасят взаимно задълженията си до размера на най-малкото от тях. Такова погасяване се нарича прихващане и може да включва произволен брой банки (поне две). Предложете алгоритъм за разпознаване дали има възможност за прихващане, който работи в линейно време $\Theta(m+n)$, където m е броят на задълженията между всичките n банки.

Задача 4. Даден е речник (масив), съставен от n думи, всичките k-буквени и на кирилица. От една дума можем да получим друга чрез последователни еднобуквени замени, при условие че междинните думи се съдържат в речника. Например при $k = 4$ от "трън" се получава "глог" чрез следната редица от замени: трън → трен → тлен → клен → клон → слон → слог → глог. Съставете алгоритъм с времева сложност $O(kn \,.\, \log n)$, който по дадени две думи и речник намира най-късата редица от замени, водеща от едната дума до другата.

Задача 5. Студент се намира пред стълба с n стъпала. Той се изкачва по стълбата, като на всяка крачка взима или по едно, или по две стъпала. Съставете бърз алгоритъм, който пресмята по колко начина студентът може да се изкачи по стълбата.

Задача 6. Разглеждаме задачите за разпознаване *SubsetSum* и *Knapsack*:

SubsetSum: Дадени са масив $A[1 \ldots n]$ от цели положителни числа и цяло число $S > 0$.
Има ли множество $M \subseteq \{1, 2, 3 \ldots n\}$, за което $\sum_{k \in M} A[k] = S$?

Knapsack: Дадени са два масива $W[1 \ldots n]$ и $V[1 \ldots n]$ от цели положителни числа (съответно тегла и цени на някакви предмети); дадени са и две цели положителни числа B (максимално тегло — капацитет на раницата) и Z (минимална цена на взети предмети).
Има ли множество $M \subseteq \{1, 2, 3 \ldots n\}$, за което $\sum_{k \in M} W[k] \leq B$ и $\sum_{k \in M} V[k] \geq Z$?

Намерете полиномиална сводимост на задачата *SubsetSum* към *Knapsack*.

РЕШЕНИЯ

Задача 1.
а) С помощта на мастър-теоремата намираме $T(n) = \Theta\left(n^3 \,.\, \log n\right)$.
б) Развиваме уравнението и получаваме $T(n) = \Theta(\log n)$.
в) Това е линейно рекурентно уравнение с променлива дължина на историята. $T(n) = \Theta(2^n)$.
г) Отново чрез мастър-теоремата намираме $T(n) = \Theta\left(n^{\log_2 5}\right)$.

Задача 3. Информацията от условието на задачата може да се представи чрез насочен граф, чиито върхове съответстват на банките, а ребрата съответстват на задълженията между тях. Възможност за прихващане съществува тогава и само тогава, когато в графа има цикъл. Така дадената задача се свежда до разпознаване дали насочен граф съдържа цикъл, а това става за време $\Theta(m+n)$ чрез обхождане в дълбочина.

Задача 4. Алгоритъмът се състои от следните стъпки:

1) Сортираме речника за време $\Theta(n \,.\, \log n)$ с някой бърз алгоритъм, например HeapSort.

2) Построяваме ненасочен граф, чиито върхове са думите от речника, а ребра са възможните замени. За всяка дума възможните замени са не повече от $29k$, защото в думата има k букви, всяка от които може да бъде заменена с 29 други букви (в кирилицата има общо 30 букви). Всяка от думите кандидати се проверява за наличие в речника чрез двоично търсене за време $\Theta(\log n)$. Затова добавянето на ребрата за всяка отделна дума изисква време $\Theta(29k \,.\, \log n) = \Theta(k \,.\, \log n)$. А за всичките n думи нужното време е n пъти по-голямо, т.е. $\Theta(kn \,.\, \log n)$. Това е времето за построяване на графа.

3) Пускаме търсене в ширина, като за начален връх служи едната от двете дадени думи. Така намираме най-късия път до другата дума, т.е. най-кратката редица от замени. Тази стъпка има времева сложност $\Theta(n+m)$, където m е броят на ребрата на графа. По-горе видяхме, че за всяка дума има не повече от $29k$ на брой възможни замени; тогава за всичките n думи възможните замени са не повече от $29kn$. Тъй като графът е ненасочен, то броят на ребрата не надхвърля половината от броя на възможните замени, т.е. $m \leq 14,5kn$. Следователно търсенето в ширина изразходва време $\Theta(n+m) = \Theta(n+14,5kn) = \Theta(kn)$.

Времевата сложност на целия алгоритъм е сборът от времената на отделните стъпки, т.е. $\Theta(n \,.\, \log n) + \Theta(kn \,.\, \log n) + \Theta(kn) = \Theta(kn \,.\, \log n)$.

Задача 5. Нека X_n е броят на начините за изкачване на стълба с n стъпала. Лесно се вижда, че $X_1 = 1,\ X_2 = 2$. С последната крачка студентът изкачва или едно, или две стъпала. Ето защо $X_k = X_{k-1} + X_{k-2}\,,\ k \geq 3$. Чрез това уравнение пресмятаме X_k за k от 3 до n и връщаме X_n (получават се числа на Фибоначи). Този алгоритъм има времева сложност $\Theta(n)$.

Задача 6. Входните данни на задачата *SubsetSum* са масивът A и числото S. Конструираме задача *Knapsack*, като правим масивите W и V равни на A, а числата B и Z — равни на S.

Описаната сводимост води до еднакви отговори на двете задачи, защото са в сила твърденията:

$$\sum_{k\in M} W[k] \leq B \iff \sum_{k\in M} A[k] \leq S;$$

$$\sum_{k\in M} V[k] \geq Z \iff \sum_{k\in M} A[k] \geq S;$$

$$\left(\sum_{k\in M} A[k] \leq S\right) \wedge \left(\sum_{k\in M} A[k] \geq S\right) \iff \sum_{k\in M} A[k] = S.$$