Семестриално контролно за курса ДАА-избираем, 16.12.2014г.

Име: _______________________________, ФН:_____, Спец./курс:_______

Задача	1	2	3	4	5	6	Общо
получени точки							
максимум точки	20	20	20	20	20	20	120

Забележка: За отлична оценка са достатъчни 100 точки!

Задача 1 Подредете по асимптотично нарастване функциите по-долу. Обосновете отговора си и напишете в явен вид подредбата.

$$n!, \qquad 7n^2 + 150n + 4, \qquad \sqrt[3]{n^7 + 8\ln n}, \qquad 9^n, \qquad 4^{2n-1}, \qquad n^{10}7^n, \qquad (3n)^{2n}$$

Задача 2 Решете следните рекурентни уравнения:

а) $T(n) = 12T(n-1) - 35T(n-2) + 8n^2 5^n$ б) $T(n) = T(n-1) + \sqrt[5]{n^7}$

в) $T(n) = 1000T(\frac{n}{10}) + 2n^4$ г) $T(n) = 128T(\frac{n}{2}) + 9n^3 \lg n$

Задача 3 Алгоритъмът по-долу получава масив A, който съдържа $n > 1$ числа:

```
GetFreq(A[1...n])
w ← A[1]
cnt ← 1
for i = 2 to n
    if cnt = 0
        cnt ← 1
        w ← A[i]
    else if A[i] = w
            cnt ← cnt + 1
         else cnt ← cnt − 1
return w
```

(а - 5 точки) Оценете времевата сложност на алгоритъма.

(б - 15 точки) Докажете, че ако масивът съдържа елемент, който се повтаря повече от $\frac{n}{2}$ пъти, то алгоритъмът връща стойността на този често срещан елемент.

Задача 4 Всеки от общо $2n$ играчи има рейтинг - число, което показва колко добър състезател е съответният играч. Треньорът им иска да ги раздели на два отбора от по n играчи по такъв начин, че разликата на рейтингите на двата отбора да е максимална, т.е. двата отбора да са възможно най-неравни противници. (Рейтинг на отбор е сборът от рейтингите на състезателите от отбора.)

а) Съставете възможно най-бърз алгоритъм за разделяне на играчите в два отбора според желанието на треньора.

б) Оценете времевата сложност на предложения от Вас алгоритъм.

Задача 5 Докажете, че не съществува реализация на процедурата $Heapify(A, i)$ за оправяне на дефект надолу в пирамида, чиято времева сложност е в асимптотичен клас $o(\lg n)$.

Задача 6 Разглеждаме следния рекурсивен алгоритъм:

```
HasDuplicates(A[1...n])
if n = 1
    return false
k ← 1
last ← A[n]
while k < n do
    if A[k] = last
        return true
    k ← k + 1
return HasDuplicates(A[1...n − 1])
```

а) Докажете, че алгоритъмът връща *true* точно когато масивът A съдържа двойка еднакви елементи.

б) Дайте оценка за сложността на алгоритъма по време.

Решения:

Задача 3

а) Тъй като в единствения цикъл се извършват констатен брой сметки, сложността е $\Theta(n)$.

б) Представете си атобус с двойни седалки. Пътниците пристигат един по един, а стюардесата ги пуска по двойки, така че на всяка двойна седалка да седнат пътници с различни имена.

Пред автобуса ще се образува опашка от адаши (едноименници) с име w и с дължина cnt. След пристигането на n пътници единствено тия с име w могат да са повече от $n/2$, тъй като пътниците с други имена са вътре в автобуса и са на различни седалки.

Тази интерпретация може да се формулира със следната инварианта:

Всеки път когато сме на ред 3, елементите на подмасива $A[1, 2, \ldots, i-1]$ могат да се представят разделени в две групи: едната съдържа cnt елемента със стойност w, а другата се състои от останалите $i - 1 - cnt$ елемента, групирани в двойки различни елементи.

След доказване на верността на инвариантата получаваме исканото свойство на алгоритъма.

Задача 4

а) Сортираме състезателите по рейтинг и слагаме най-слабите (първата половина от сортирания масив) в първия отбор, а останалите – във втория.

б) Очевидно скоростта на горната алгоритмична схема е $\Theta(n \lg n)$, ако използваме оптимален сортиращ алгоритъм ($HeapSort$, $MergeSort$).

Задачата може да се реши и за линейно време. Първо намираме медианата на масива (елемент, който се намира в средата на сортирания масив), после разделяме елементите на по-малки и по-големи от медианата.

Малка модификация на $QuickSort$ намира медианата за линейно време в средния случай. Модификацията вика рекурсивно алгоритъма само за интервала, съдържащ индекса $n/2$.

Съществува линеен алгориъм за намиране на медианата: Time Bounds for Selection.

Той не е задължителна част от курса ДАА, затова и решението с оценка $\Theta(n \lg n)$ носи пълен брой точки.

Задача 5

В лекциите на курса ДАА за алгоритъма $HeapSort$ даваме реализация, която използва $2n$ пъти процедурата $Heapify(A, i)$ в два отделни цикъла за изграждане и разграждане на пирамида.

Тази реализация има скорост $\Theta(nf(n))$, където с $f(n)$ означаваме скоростта на $Heapify$.

Ако $f(n) = o(\lg n)$, ще получим алгоритъм за сортиране със сложност $o(n \lg n)$, което противоречи на теоремата за долна граница на скоростта на сортировките.

Задача 6

а) Тъй като алгоритъмът $HasDuplicates$ е рекурсивен, удобно е да извършим доказателството на коретността му с индукция. В конкретния случай индукцията провеждаме по дължината на входа n.

Очевидно алгоритъмът е коректен за вход с дължина 1, тогава се изпълняват само редове 1 и 2.

Нека алгоритъмът е коректен за вход с дължина $n - 1$ и $n > 1$.

При вход с дължина n $HasDuplicates$ първо проверява дали $A[n] = A[k]$ за $k = 1, 2, \ldots, n - 1$. С проста инварианта на цикъл се установява, че ако намери такова съвпадение, той ще завърши коректно, връщайки $true$.

В противен случай в масива има съвпадащи елементи само ако има съвпадащи в по-малкия масив $A[1, 2, \ldots, n - 1]$. Точно това се установява с рекурсивното извикване на ред 9.

б) Скоростта е ограничена от решението на рекурентното уравнение $T(n) = T(n - 1) + cn$. Решението му е $\Theta(n^2)$ и тази граница се достига когато масивът съдържа различни елементи.