Писмен изпит (поправителна сесия) по ДАА, 06.09.2014г.

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

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

*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.

$$\sum_{i=1}^{n} \frac{n^2}{i}, \qquad \lg((n!)^n), \qquad \left(\frac{3}{2}\right)^{2n}, \qquad n^2 + n \lg n, \qquad \sum_{i=0}^{n} \frac{1}{i!}, \qquad \frac{3^n}{2^n}, \qquad \binom{n}{2}\frac{1}{\lg n}, \qquad \sum_{i=1}^{n} \left(\frac{3}{2}\right)^i$$

**Задача 2** Решете следните рекурентни отношения:

а) $T(n) = 3T(\frac{n}{2}) + n^2$ б) $T(n) = 5T(\frac{n}{2}) + n^2 \lg n$ в) $T(n) = 9T(n-2) + 3^n$ г) $T(n) = \sum_{i=0}^{n-1} T(i) + 3^n$

**Задача 3** Докажете, че InversionSort е сортиращ (**20 т.**) и че сложността му по време е $\Theta(n^2)$ (**20 т.**).

```
InversionSort(A[1...n]: array of integers)
1  i ← 1
2  while i < n do
3      if A[i] > A[i+1]
4          swap(A[i], A[i+1])
5          if i = 1
6              i ← i+1
7          else i ← i−1
8      else i ← i+1
```

**Задача 4** *Подпоследователност на масив* $A[1, \ldots, n]$ е всяка редица $A_{i_1}, A_{i_2}, \ldots, A_{i_m}$, където $1 \leq i_1 < i_2 < \ldots i_m \leq n$ и $1 \leq m \leq n$. Неформално, подпоследователност се получава след изтриване на някакви (може и нула) елементи от оригиналния масив, като редът на останалите е същият като в оригиналния масив. Редица от числа $a_1, a_2, \ldots, a_t$ за произволно $t \geq 2$ се нарича *алтернираща*, ако в сила е точно едно от следните:

- $a_i < a_{i+1}$ за всички четни $i$ и $a_i > a_{i+1}$ за всички нечетни $i$, където $1 \leq i \leq t-1$;
- $a_i > a_{i+1}$ за всички четни $i$ и $a_i < a_{i+1}$ за всички нечетни $i$, където $1 \leq i \leq t-1$.

Предложете колкото е възможно по-бърз алгоритъм, който намира алтернираща подпоследователност с максимален брой елементи в масив от цели числа $A[1, \ldots, n]$.

**Задача 5** Предложете колкото е възможно по-бърз алгоритъм, който намира най-дълъг път в дърво (**10 т.**). Предложете колкото е възможно по-бърз алгоритъм за същата задача, но сега в тегловен вариант, тоест ребрата имат положителни тегла, а дължината на път е сумата от теглата на ребрата му (**10 т.**).

**Задача 6** Задачата МаксКлика се дефинира така: при даден граф $G(V, E)$ и число $k$, дали най-голямата клика в $G$ има поне $k$ върха. Задачата МинВърховоПокриване се дефинира така: при даден граф $G(V, E)$ и число $k$, дали най-малкото върхово покриване на $G$ има не повече от $k$ върха. Да си припомним две дефиниции: *клика* в граф е всеки подграф, който е пълен граф; *върхово покриване* на граф $G(V, E)$ е всяко $U \subseteq V$, такова че за всяко ребро $(u, v) \in E$ е изпълнено, че $u \in U$ или $v \in U$.

Предложете полиномиална сводимост МаксКлика $\propto$ МинВърховоПокриване. Обосновете коректността ѝ.

**Решение на зад. 3:** Ще ползваме очевидната инварианта: Всеки път когато алгоритъмът е на ред 2, подмасивът $A[1 \dots i]$ е подреден (не съдържа инверсии).

Интересен е въпросът дали $i$ достига стойност $n$, ако това стане, масивът ще бъде сортиран.

Ще докажем по индукция лема, която дава позитивен отговор на горния въпрос и влече коректността на алгоритъма:

Лема: За всяко $k,\ 1 \leq k \leq n$ променливата $i$ достига стойност $k$ при работата на алгоритъма $InversionSort$.

Доказателство: $k = 1$ се достига още при първото изпълнение на ред 2.

Нека $i$ достига стойност $k$ на ред 2. Има два случая – проверката на ред 3 сработва. В този случай елементът $A[k+1]$ след най-много $k$ на брой размени ще бъде поставен на точното си място в подмасива $A[1 \dots k+1]$ и след още най-много $k$ преминавания през цикъла, $i$ ще достигне стойност $k+1$.

В другият случай, когато проверката на ред 3 не сработва, $i$ веднага достига стойност $k+1$.

Прилагаме принципа на математическата индукция, и завършваме доказателството на лемата.

Сега да означим с $T_k$ броят изпълнения на цикъла $while$, когато за пръв път $i$ достига стойност $k$. От разсъжденията, проведени в доказателството на лемата се вижда, че след момента $T_k$ са достатъчни не повече от $2k$ изпълнения на цикъла за достигане на стойност $k+1$, т.е. функцията, изразяваща сложността на алгоритъма ще расте по-бавно от решението на рекурентната зависимост $T(k+1) = T(k) + ck$, където $c$ е подходяща константа. Решението на рекурентната зависимост е $\Theta(k^2)$.

Алгоритъмът завършва, когато $i$ достигне $n$, следователно сложността му е ограничена от $\Theta(n^2)$.

Остава да забележим, че при всяко преминаване през цикъла алгоритъмът унищожава най-много една инверсия (той унищожава инверсия от вида $< i, i+1 >$, която не влияе на останалите инверсии в масива). Ако подадем на входа обратно нареден масив, броят на инверсиите в него е $\Theta(n^2)$, следователно сложността му е точно $\Theta(n^2)$.