Решения на задачите от второто малко контролно по ДАА на група 5, проведено на 05.06.2024 г.

Задача 1. (60 т.) *Хубаво число ще наричаме всяко естествено число от вида* 2^a3^b *за някои* $a, b \in \mathbb{N}$*. Да се състави* ***итеративен*** *алгоритъм, който при подадено* $n \in \mathbb{N}$ *връща* $(n+1)$*-вото по големина хубаво число. Алгоритъмът* ***трябва*** *да е съставен по схемата* ***динамично програмиране*** *и да има сложност по време* $\Theta(n)$*. Няма нужда да се прави доказателство за коректност, достатъчно е да се напише правилен инвариант.*

Решение. Следният алгоритъм решава задачата:

```
1  Beautiful(Nat n):
2      Array(Nat) B[0..n]
3      B[0] ← 1
4      Nat two_idx ← 0
5      Nat three_idx ← 0
6
7      for i ← 1 to n:
8          two_candidate ← 2 * B[two_idx]
9          three_candidate ← 3 * B[three_idx]
10         B[i] ← min(two_candidate, three_candidate)
11
12         if B[i] = two_candidate:
13             two_idx ← two_idx + 1
14         if B[i] = three_candidate:
15             three_idx ← three_idx + 1
16
17     return B[n]
```

Инвариант. При всяко достигане на условието за край на цикъла на ред 7:

- `B[j]` е `(j+1)`-вото хубаво число за всяко `j` между `0` и `i-1` включително;
- `B[two_idx]` е най-малкото хубаво число `k`, за което `2k` е извън масива `B[0..i-1]`;
- `B[three_idx]` е най-малкото хубаво число `k`, за което `3k` е извън масива `B[0..i-1]`.

Критерии за оценяване:

- за правилен алгоритъм – 30 точки;
- за правилно формулиран инвариант – 30 точки.

Задача 2. **(60 т.)** *Нека разгледаме задачата* TABLE-SUM:

Вход: *Таблица от естествени числа* $T[1 \dots n, 1 \dots m]$ *и* $t \in \mathbb{N}$.

Въпрос: *Има ли* $1 \leq i_1, \dots, i_n \leq m$, *за които* $\sum\limits_{k=1}^{n} T[k, i_k] = t$*?*

Докажете формално, че TABLE-SUM *е* ***NP****-пълна задача.*

Решение. При подадена таблица $T[1 \dots n, 1 \dots m]$ и сертификат $C[1 \dots n]$, представящ индекси $1 \leq i_1, \dots, i_n \leq m$, можем за време $\Theta(n)$ да проверим дали е изпълнено, че:

$$\sum_{k=1}^{n} T[k, i_k] = t.$$

Така задачата TABLE-SUM е в класа **NP**.
Също така много лесно можем за полиномиално време да сведем задачата SUBSET-SUM към задачата TABLE-SUM. При подаден вход масив $A[1 \dots n]$ и число $t \in \mathbb{N}$ за време $\Theta(n)$ можем да построим таблица $T[1 \dots n, 1 \dots 2]$, където $T[i, 1] = A[i]$ и $T[i, 2] = 0$. Тогава са изпълнени следните твърдения:

1. Ако има подредица на $A[1 \dots n]$ със сума на елементите t, то тогава за всяко $1 \leq j \leq n$ ще дефинираме i_j да бъде 1 т.с.т.к. $A[i]$ участва в съответната подредица. Индексите $1 \leq i_1, \dots, i_n \leq 2$ имат желаното свойство.

2. Ако $1 \leq i_1, \dots, i_n \leq 2$ са индекси с желаното свойство, то взимайки тези $1 \leq j \leq n$, за които $i_j = 1$, ще получим подредица на $A[1 \dots n]$, елементите на която се сумират до t.

С това получихме, че задачата TABLE-SUM е **NP**-трудна, и понеже принадлежи на класа **NP**, тя е **NP**-пълна.

Критерии за оценяване:

- за обосновка, че TABLE-SUM е в класа **NP** – 30 точки;

- за обосновка, че TABLE-SUM е **NP**-трудна задача – 30 точки.