Kraków 22.10.2018

Zestaw zadań nr. 2

- Zadanie 1
  Zakładając że $f_1(n)$ jest $O(g_1(n))$ i $f_2(n)$ jest $O(g_2(n))$ udowodnij (wprost z definicji) następujące własnoci:
  - a) $f_1(n) + f_2(n) = O(max(g_1(n)), g_2(n)))$
  - b) Jeśli istnieje liczba k taka, że dla każdego $n > k$, $g_1(n) < g_2(n)$
    to $f_1(n) + f_2(n) = O(g_2(n))$
  - c) $f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n))$
  - d) $O(c \cdot g(n)) = O(g(n))$
  - e) c jest O(1)
- Zadanie 2
  Udowodnij że:
  - a) $\sum_{i=1}^{n} i^2$ jest $O(n^3)$ i ogólniej $\sum_{i=1}^{n} i^k = O(n^{k+1})$
  - b) $an^k/ln(n) = O(n^k)$ ale nie $\Theta(n^k)$
  - c)$n^{1,1} + nln(n) = \Theta(n^{1,1})$
  - d)2^n jest $O(n!)$, a $(n!)$!= $O(2^n)$
- Zadanie 4
  Przeprowadź analizę czasu działania bloków programu:
  - Pętla *while*, *do while*, *for* (nie zawierających wywołań funkcji)
  - Instrukcja *for* sekwencyjnego bloku instrukcji
  - Czas działania programu zawierającego wywołanie funkcji
  - Czas działania bloku zawierającego funkcje rekurencyjne
- Zadanie 5
  Rozważmy problem wykrywania powtarzających się elementów w ciągu n liczb $< x_1, x_2, ..., x_n >$. Pokaż, że można rozwiązać ten problem w czasie $\Theta(n\ lg_2 n)$.
  Wskazówka: Przyjmij że sortowanie możesz wykonać w czasie $\Theta(n\ lg_2 n)$.