Mihăilă Andrea

24.06.2021

gr. 214

Examen scris

- SDA -

Subiect 112

A. $T(n) = \begin{cases} 1, & n = 1 \\ T(\frac{n}{2}) + n + 2, & n > 1 \end{cases}$

$T(n) = T(\frac{n}{2}) + n + 2$

Notez $n = 2^k$

$T(2^k) = T(\frac{2^k}{2}) + 2^k + 2 \Rightarrow T(2^k) = T(2^{k-1}) + 2^k + 2$

$T(2^{k-1}) = T(2^{k-2}) + 2^{k-1} + 2$

$T(2^{k-2}) = T(2^{k-3}) + 2^{k-2} + 2$

—— —— —— ——

$T(2^2) = T(2^1) + 2^2 + 2$

$T(2^1) = T(2^0) + 2^1 + 2$

$T(2^0) = 1 \quad (+)$

$T(2^k) = 2^k + 2^{k-1} + \dots + 2^2 + 2^1 + 2^0 + \underbrace{2 + 2 + 2 + \dots + 2}_{\text{de } k \text{ ori}}$

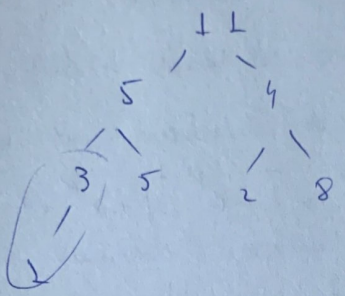
$T(2^k) = 2^{k+1} - 1 + 2 \cdot k$

$T(2^k) = 2^k \cdot 2 - 1 \overset{2^k = n}{\Rightarrow} T(n) = 2 \cdot n - 1 + 2 \cdot k \in \Theta(n)$

Mihăilă Andrea
gr. 214
S. 112

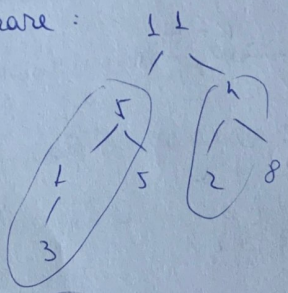
avem complexitate medie liniară, iar cazul defavorabil este când $n = 2^k$.

B. Ansamblul inițial:

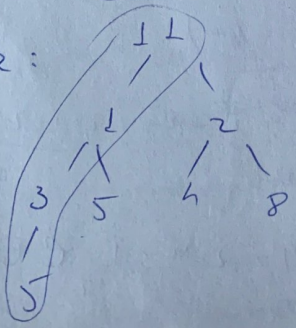


Restructurare:

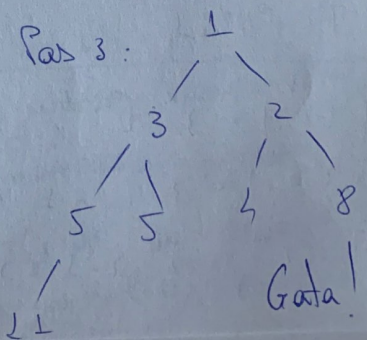
Pas 1:



Pas 2:



Pas 3:



Mihăilă Andrea
gr. 214
S. 112

Ansamblul 1, 3, 2, 5, 5, 4, 8, 11 final este un min-heap. Folosind heapsort in place, restructurăm vectorul incât să devină un ansamblu, astfel:
- se incearcă refacerea proprietății de ansamblu de jos in sus (de la frunze spre rădăcină), pornind de la nodurile de inaltime h-1, h-2 ... până la 0.

c. Varianta corectă este: c) data[11]. După ce se face redimensionarea tabelei pentru a putea fi adăugat un element nou, noul element se va adauga la finalul cozii, deoarece coada funcționează pe principiul FIFO.

c. Varianta corectă este: d) 14 2 11 1 3 10 30 7 40. Parcurgerea pe niveluri, denumită și în latime, presupune ~~să noduriile~~ trecerea prin noduri pe niveluri, in ordine de la stanga la dreapta, adică începem cu nivelul 0, coborâm la 1

Mihăilă Andra
gr. 214
S. 112

și parcurgem de la stânga la dreapta, coborâm la 2
și tot așa.

D. Rezolvare recursivă:

TAD Arbore:

v: TElem [ ] dreapta: TElem[ ]
rad: intreg K - intreg
stânga: TElem[ ]

Subalgoritm descendenti_pe_nivel (arb, K, vector)
  dacă ¬ vid(arb) atunci // dacă (sub) arborele
  ~~Cât timp K<>0 execută~~ curent este nevid
  ~~dacă K=0 atunci~~ // se execută până când ajungem la înălțimea maximă
  dacă ¬ vid(stâng(arb)) atunci
// dacă avem subarbore stâng, reapelăm
//recursiv subalgoritmul incrementând nivelul
    nivel(stâng(arb), K+1, V)
  sf dacă
  dacă ¬ vid(drept(arb)) atunci
// analog pt. subarborele drept
    nivel(drept(arb), K+1, V)
sf cât timp sf dacă

Mihăilă Andra
gr. 214
S. 112

~~#~~

sf_daca

sf - subalgoritm

Pentru varianta nerecursivă trebuia implementată o coadă (sau o stivă.) cu perechi.

{ l : îNod
{ nivel : intreg

sau

{ arb: AB
{ nivel: intreg

Complexitatea este Theta (n^2), unde n este nr de elemente.

pre: arb; arbore, v: vector, k: intreg
post: v cu elementele cerute in el