

1. Analizati timpul mediu si defavorabil pentru urmatorul subalgoritm. Justificati rezultatul.

```

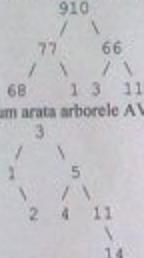
Subalgoritm h(n, A, B, C)
| (pre: n:Intreg; A, B, C:char)
| daca n=1 atunci
| | h(n-1, A, C, B)
| | scrie n, A, B
| | h(n-1, C, B, A)
| altfel
| | scrie 1, A, B
| sfata

```

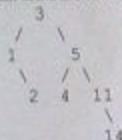
2. Raspundeti la urmatoarele cerinte:

2.1 Fie o colectie de chei naturale si o functie de dispersie definita astfel: $d(c) = \text{numarul zecimal corespunzator bitilor } b_7 b_6 b_5 b_4 \text{ ai cheii } c$. Ilustrati tabela cu adresare deschisa si verificare liniara rezultata in urma inserarii cheilor: 23, 11, 8, 18, 3, 19, 34, 15, 27.

2.2 Aratati ansamblul rezultat prin inserarea valorii 82 in urmatorul ansamblu.



- 2.3 Cum arata arborele AVL de mai jos in urma stergerii cheii 1? Ce operatie se aplica pentru reechilibrare?



3. Raspundeti la urmatoarele intrebari, justificand rezultatul (rezultatele) ales (alese).

3.1 Care este clasa de complexitate pentru functia $f(n) = \sum_{i=1}^n 3i$?

- a) $O(n)$ b) $O(n^2)$ c) $O(n^4)$ d) $\Theta(n^3)$

3.2 Care tip de lista este cea mai potrivita pentru a raspunde la intrebarea "Care este elementul de pe pozitia n ?"

- a) lista implementata pe vector b) lista dublu inlantuita c) lista simplu inlantuita d) liste simplu si dublu inlantuite

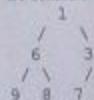
3.3. Intr-o implementare a Cozii folosind o lista inlantuita, unde are loc adaugarea unui element?

- a) la inceput b) la sfarsit c) dupa toate valorile mai mari decat elementul d) dupa toate valorile mai mici decat elementul

3.4 Alegeti afirmatia corecta:

- a) orice arbore binar este fie complet, fie plin b) orice arbore binar complet este plin c) orice arbore binar plin este si complet
d) nici un arbore binar nu este si complet si plin

3.5 Se considera urmatorul arbore binar. Alegeti afirmatiile corecte.



- a) verifica proprietatea de ansamblu, are structura de ansamblu b) nu verifica proprietatea de ansamblu, dar are structura de ansamblu c) nu este un ansamblu d) nu verifica proprietatea de ansamblu, dar are structura de ansamblu e) este ansamblu

3.6 Care este cea mai potrivita definire pentru o colizie? intr-o tabela de dispersie?

- a) doua intrari sunt identice, exceptand cheile b) doua intrari diferite au aceeasi valoare a cheii
c) doua intrari cu chei diferite au aceeasi valoare de dispersie d) doua intrari cu aceeasi cheie nu valoari diferite de dispersie

4. Aratati cum se poate implementa o Coada folosind 2 Stive. Dati reprezentarea Cozii, scriseti in Pseudocod operatiile din interfața Cozii, folosind doar operatiile din interfața Stivei (fara a face apel la reprezentarea acesteia). Analizati timpul de executie pentru operatiile cozii.

5. Sa se descrie subalgoritmul care gaseste parintele unui nod p intr-un arbore binar. Arborele se reprezinta inlantuit cu alocare dinamica a nodurilor. Se va folosi o operatie nerecursiva. Se va preciza complexitatea operatiei.

Punctaj of - 1p; Subiect 1 – 1.5p; Subiect 2 – 1.5p (3*0.5p); Subiect 3 – 2p (6*0.33p); Subiect 4 – 1.5p; Subiect 5 – 2.5p;

Total - 10p

R1

$$1) T(n) = \begin{cases} 2 * T(n-1), & n > 1 \\ 1, & \text{altfel} \end{cases}$$

$$T(n) = 2 * T(n-1)$$

$$T(n-1) = 2 * T(n-2) \mid * 2$$

$$T(n-2) = 2 * T(n-3) \mid * 2^2$$

...

$$T(2) = 2 * T(1) \mid * 2^{n-2}$$

$$T(n) = 2^{n-1} * T(1) = 2^{n-1} \text{ apartine lui } 2^n \Rightarrow \Theta(2^n)$$

In cazul de fata timpul mediu coincide cu cel defavorabil pentru ca acest algoritm executa un anumit numar de pasi exact.

2.1) $23 = 10111 \Rightarrow 0111 = 0111$ (val in baza 2) $\Rightarrow 7 - \text{val fct de dispersie}$

$11 = 1011 \Rightarrow 11 - \text{val fct de dispersie}$

$8 = 1000 \Rightarrow 8 - \text{val fct de dispersie}$

$18 = 2 - \text{val fct de dispersie}$

$3 = 3 - \text{val fct de dispersie}$

$19 = 3 - \text{val fct de dispersie}$

$34 = 2 - \text{val fct de dispersie}$

$15 = 15 - \text{val fct de dispersie}$

$27 = 11 - \text{fct de dispersie}$

Adresarea deschisa si verificarea liniara consta in retinerea elementelor intr-un vector, iar in cazul in care pe pozitia pe care ar trebui sa se puna cheia, este ocupata, se cauta loc liber intr-o permutare a multii pozitii ce incepe cu pozitia pe care ar trebui sa fi fost pusa ea.

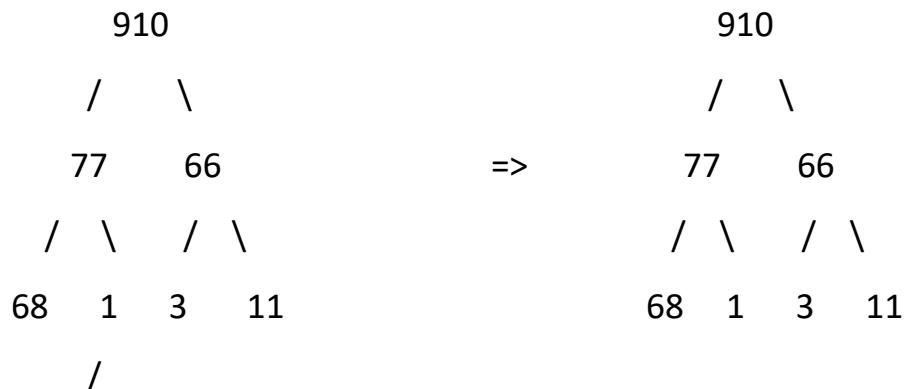
M = 16

I	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C	-1	-1	18	3	19	34	-1	23	8	-1	-1	11	27	-1	-1	15

I = Indice

C = Cheie

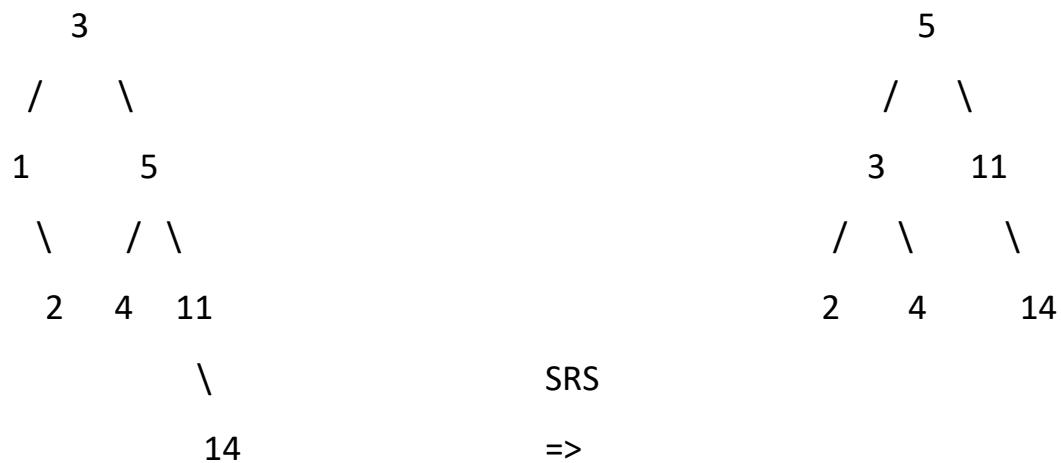
2.2) Cum arata ansamblul rezultat prin inserarea val 82?



82 (dupa care trebuie sa aplicam functia de urcare pentru acest nod)

Rez: 910, 82, 66, 77, 1, 3, 11, 68

2.3) Cum arata arborele AVL de mai jos in urma stergerii cheii 1? Ce operatie se aplica pentru reechilibrare?



3)

3.1) Care este clasa de complexitate pentru fct $f(n) = \sum(1,n) 3i$;

$$f(n) = 3(1 + 2 + \dots + n) = n(n+1)/2 \quad \text{aprox } n^2$$

Rasp: b) , c)

3.2) Care tip de lista este cea mai potrivita pentru a raspunde la intrebarea "Care este elementul de pe pozitia n?"

Rasp: a)

3.3) Intr-o implementare a Cozii folosind o lista inlantuita, unde are loc adaugarea unui element?

Rasp: b)

3.4) Alegeti afirmatia corecta:

Rasp: c)

3.5) Se considera urmatorul arbore binar. Alegeti afir corecte:

Rasp: a), e)

3.6) Care e cea mai potrivita definitie pentru o coliziune intr-o tabela de dispersie?

Rasp: c)

4) Aratati cum se poate implementata o Coada folosind 2 Stive. Dati reprezentarea Cozii, scrieti in Pseudocod op din interfata Cozii fol doar op din interfata Stivei. Analizati timpul de executie pt op.

O coada se poate implementa cu 2 stive in modul urmator: intr-o stiva o sa retinem elementele in ordinea buna, elementul din varful stivei fiind primul element din coada.

TAD Coada:

S1, s2: stiva

Subalgoritm creeaza(c){Complexitatea este Theta(1)}

Creeaza(c.s1)

Creeaza(c.s2)

SfSubalgoritm

Subalgoritm adauga(c, e) {Complexitatea este Theta(n)}

Cat timp vida(c.s1) = fals executa

 sterge(c.s1, elem)

 adauga(c.s2, elem)

SfCatTimp

Adauga(c.s2, e)

Cat timp vida(c.s2) = fals executa

 sterge(c.s2, elem)

 adauga(c.s1, elem)

SfCatTimp

SfSubalgoritm

Subalgoritm sterge(c, e) {Complexitatea este Theta(1)}

 Sterge(c.s1,e)

SfSubalgoritm

Subalgoritm element(c, e) {Complexitatea este Theta(1)}

 Element(c.s1,e)

SfSubalgoritm

Functie vida(c) {Complexitatea este Theta(1)}

 Vida<-vida(c.s1)

SfFunctie

Functie plina(c) {Complexitatea este Theta(1)}

Plina<-plina(c.s1)

SfFunctie

Subalgoritm distruge(c){Complexitatea variaza in functie de reprezentarea stivei}

Distruge(c.s1)

Distruge(c.s2)

SfSubalgoritm

5) Sa se descrie subalgoritmul care gaseste parintele unui nod p intr-un arbore binar. Arboarele se reprez inlantuit cu alocarea dinamica a nodurilor. Se va folosi o operatie nerecursiva. Se va preciza complexitatea operatiei.

Subalgoritm gasesete_parinte(arb, p, parinte)

Creeaza(c){facem o coada}

Daca arb.rad != NIL atunci

 Adauga(c, arb.rad)

 CatTimp vida(c) = fals execută

 Sterge(c, nod)

 Daca [nod].st != NIL atunci

 Daca [[nod].st].e = [p].e atunci

 parinte<-nod

 @break

SfDaca

 Adauga(c, [nod].st)

SfDaca

Daca [nod].dr != NIL atunci

Daca [[nod].dr].e = [p].e atunci

parinte<-nod

@break

SfDaca

Adauga(c, [nod].dr)

SfDaca

SfCatTimp

SfDaca

SfSubalgoritm{Complexitatea algoritmului este $O(h)$, h – inaltimea arborelui}

A. Deducreți timpii mediu și defavorabil pentru următorul subalgoritm. Justificați rezultatul.

Funcția **F**(n, i) este {:Intreg}

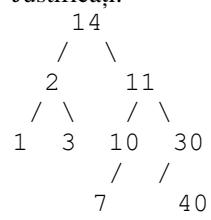
```
| {pre: n, i:Intreg}
| dacă n=1 atunci
| | F←1
| altfel
| | pentru j←1,n executa i←i+1 sfpentru
| | m ← n div 2
| | dacă i mod 2 =0 atunci
| | | F← F(m, i) - i
| | altfel
| | | F← F(m, i) + i
| | Sfdacă
| Sfdacă
```

SfF

B. Fie următorul arbore binar. Indicați ordinea în care se parcurează nodurile în

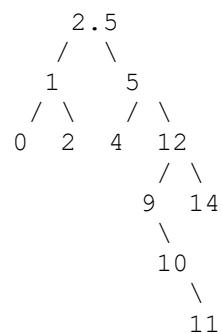
- a) preordine
- b) inordine
- c) postordine
- d) lățime.

Justificați.



C. Înălțimea nodului 9 în arborele binar de mai jos este:

- a) 1
- b) 2
- c) 3
- d) 4

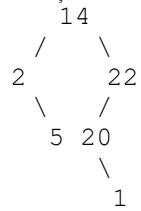


Justificati

C. Într-o implementare a Stivei folosind o listă înlanțuită, unde are loc adăugarea unui element? Justificati

- a) la început
- b) la sfârșit
- c) după toate valorile mai mari decât elementul
- d) după toate valorile mai mici decât elementul

D. Să se scrie în Pseudocod subalgoritmul care găsește numărul asociat unei valori e dintr-un arbore binar, numerotarea nodurilor făcându-se în inordine. Elementele din nodurile arborelui sunt distințe, arborele se reprezintă secvențial, pe vector, folosind ca schemă de memorare ansamblu.. Se va folosi o operație nerecursivă. Se va preciza complexitatea operației. Folosiți comentarii pentru a ușura înțelegerea soluției. Ex : Pentru arborele de mai jos, dacă $e=20$, atunci numarul asociat lui e este **4**.



R2

A. Deduceti timpii mediu si defavorabil pt urmaoturl subalg. Justificati rez.

Functia F (n, i) este {:Intreg}

```
| {pre: n, i:Intreg}
| dacă n=1 atunci
|   | F <- 1
|   | altfel
|   |   pentru j<-1, n executa i <- i+1 sfpentru
|   |   m ← n div 2
|   |   dacă i mod 2 =0 atunci
|   |   | F <- F (m, i) - i
|   |   | altfel
|   |   | F <- F (m, i) + i
|   | Sfdacă
| Sfdacă
```

SfF

$$T(n) = \{T(n/2) + n + 1, n \neq 1$$

$$\{1, n = 1$$

$$N = 2^k$$

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

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

$$T(2^{(k-2)}) = T(2^{(k-3)}) + 2^{(k-2)} + 1$$

...

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

$$T(2^k) = k + (1 + 2 + 2^2 + \dots + 2^k)$$

$$T(2^k) = k + 2^{(k+1)} - 1; \quad n = 2^k$$

$$T(n) = \log_2(n) + 2n - 1$$

$$BC = WC = AC = OC$$

=> Complexitatea alg este $\Theta(n)$, alegem maximul dintre n , $\log_2(n)$ si -1

B. Fie urmatorul arbore binar. Indicati ordinea in care se parcurg nodurile in

14

a) preordine(RSD): 14, 2, 1, 3, 11, 10, 7, 30, 40

/ \

b) inordine(SRD): 1, 2, 3, 14, 7, 10, 11, 40, 30

2 11

c) postordine(SDR): 1, 3, 2, 7, 10, 40, 30, 11, 14

/ \ / \

d) latime(BFS): 14, 2, 11, 1, 3, 10, 30, 7, 40

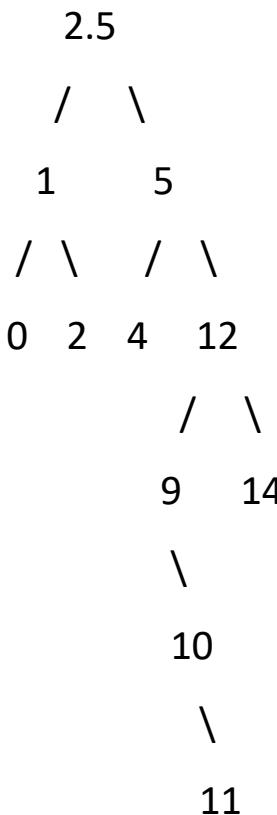
1 3 10 30

/ /

7 40

C. Inaltimea nodului 9 in arborele binar de mai jos este:

Rasp: b) 2, deoarece inaltimea unui nod este inaltimea subarborelui ce il are ca si radacina in cazul de fata pe nodul 9



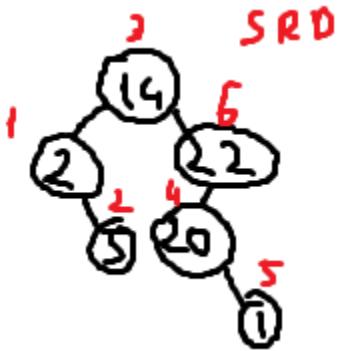
C1. Intr-o implementare a stivei folosind o lista inlantuita, unde are loc adaugarea unui element? Justificati

Rasp: a) pentru ca se pot scoate elemente cu usurinta de pe prima pozitie in lista noastra inlantuita

b) pentru ca se pot scoate la fel de usor elementele de pe ultima pozitie.

Ambele operatii au Theta(1).

D. Să se scrie în Pseudocod subalgoritmul care găsește numărul asociat unei valori e dintr-un arbore binar, numerotarea nodurilor făcându-se în inordine. Elementele din nodurile arborelui sunt distincte, arborele se reprezintă secvențial, pe vector, folosind ca schemă de memorare ansamblul. Se va folosi o operație nerecursivă. Se va preciza complexitatea operației. Folosiți comentarii pentru a ușura înțelegerea soluției. Ex : Pentru arborele de mai jos, dacă $e=20$, atunci numărul asociat lui este 4.



Descrierea alg:

O să parcurgem arborele în inordine, o să numerotăm nodurile acestuia, iar în momentul în care ajungem la elementul căutat, reținem într-o variabilă numărul asociat lui, după care o să continuăm parcurgerea.

TAD Arbore:

V : TElem[]

```

// poz k -=> poz 2k – st, poz 2k+1 – dr
// ar merge mai multe comms

Subalgoritm numar_asociat(arb, e, rez)

{Pre: arb : Arbore, e : TElem}

{Post: rez : Intreg, daca nu exista rez = 0}

Rez<- 0

Daca arb.v[1] != NULL_TELEM atunci

    ct<-1

    I<-1

    Creeaza(s){ inordine} // stiva

    Cat timp vida(s) = fals sau arb.v[i] != NULL_TELEM executa

        Cat timp arb.v[i] != NULL_TELEM executa

            adauga(s, i)

            I<-2*i

            SfCatTimp

            sterge(s, i)

            Daca arb.v[i] = e atunci{daca exista elementul, salvam
                                indiciele}

                rez<-ct

                SfDaca

                ct<-ct+1

                I<-2*I+1

            SfCatTimp

```

SfDaca

SfSubalgoritm

Complexitatea algoritmului este $\Theta(n)$, unde n este numarul de elemente din arbore.



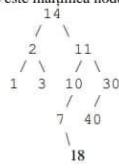
A. Deduceți timpii mediu și defavorabil pentru următorul algoritm. Justificați rezultatul.

Functia f(n) este (.Intreg)
| {pre: n:Intreg}
| S<-0; m <-n
| catimp m=0 execută
| | S<-S+[m/10]; m<-[m/10]
| sfcatimp
| f <-S
SfA
Algoritm A
| S<-0; @citeste n;
| pentru i=1, n execută
| | m<-2*i+1; S<-S+f(m)
| sfpentru
| scrie S
sfA

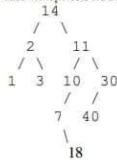
1/5



B. Care este înălțimea nodului 11 în arborele binar de mai jos? Justificați.



B. Care este înălțimea nodului 11 în arborele binar de mai jos? Justificați.



2/5

C. Care este scopul principal al unui Iterator? Justificați

- a) adăugarea unor noi obiecte la un container
- b) să parcurgă elementele unui container căte unul la moment dat
- c) să permită ștergerea unor obiecte dintr-un container

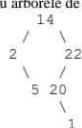
C. Stergerea unui element e dintr-un vector ordonat x_1, \dots, x_n se poate face în:

- a) $O(\log n)$
- b) $O(n)$
- c) $\Theta(n)$
- d) $\Theta(\log \log n)$

Justificati



D. Să se scrie în Pseudocod subalgoritmul care găsește numărul asociat unei valori e dintr-un arbore binar, numerotarea nodurilor făcându-se în inordine. Elementele din nodurile arborelui sunt distincte, arborele se reprezintă înlanțuit, cu înlanțuirile reprezentate pe tablou. Se va folosi o operație nerecursivă. Se va preciza complexitatea operației. Folosiți comentarii pentru a ușura înțelegerea soluției. Ex : Pentru arboarele de mai jos, dacă $e=20$, atunci numarul asociat lui e este 4.



R3

A. Deduceti timpii mediu si defavorabil pentru urmatorul algoritm. Justificati rezultatul.

Functia f are complexitatea $\Theta(\log n)$

Functia A, apeleaza functia f de n ori, dar cu o valoare ce depinde de l

=> Complexitatea algoritmului este $\sum(1,n)(\log(2^l+1))$

$$T(n) = \sum(1,n)(\log(2^l+1))$$

$$= \log(3) + \log(5) + \dots + \log(2n+1)$$

$$= \log(3 * 5 * \dots * (2n + 1))$$

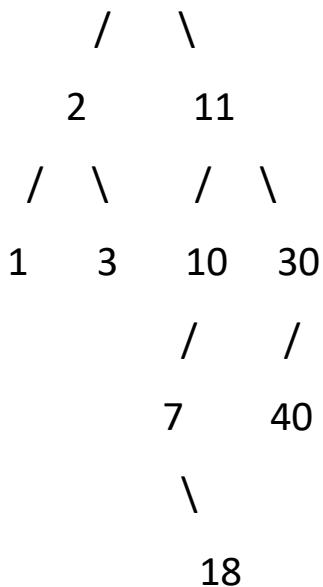
$$= \log((2n+1)! / (2 * 4 * 6 * \dots * 2n))$$

$$= \log((2n+1)! / (2^n * n!))$$

=> Complexitatea este $\Theta(\log((2n+1)! / (2^n * n!))) = n \log n$, dem de mita pe un paint

B. Care este inaltimea nodului 11 in arborele binar de mai jos? Justificati.

14



Inaltimea nodului 11 este 3 pentru ca distanta maxima de la 11 la un descendant de tipul frunza este 3.

C. Care este scopul principal al unui Iterator? Justificati

Rasp: b) rolul acestuia este da a ii returna utilizatorului elementele din containerul iterat pe rand.

C2. Stergerea unui element e dintr-un vector ordonat x_1, \dots, x_n se poate face in:

Rasp: b) $O(n)$, pentru ca stergerea se poate efectua in maximum n pasi aceasta fiind egal cu suma dintre OC la cautarea binara + OC la stergerea elementului

D. Să se scrie în Pseudocod subalgoritmul care găsește numărul asociat unei valori e dintr-un arbore binar, numerotarea nodurilor facându-se în inordine. Elementele din nodurile arborelui sunt distincte, arborele se reprezintă înlanțuit, cu înlanțuirile reprezentate pe tablou. Se va

folosi o operație nerecursivă. Se va preciza complexitatea operației. Folosiți comentarii pentru a ușura

înțelegerea soluției. Ex : Pentru arborele de mai jos, daca e=20, atunci numarul asociat lui e este 4.

Descrierea alg:

O să parcurgem arborele în inordine, să numerotăm nodurile acestuia, iar în momentul în care ajungem la elementul căutat, reținem într-o variabilă numarul asociat lui, după care o să continuăm parcurgerea.

TAD Arbore:

Element : TElem[]

Stanga : TElem[]

Dreapta : TElem[]

Rad: Intreg

// ar merge mai multe comms

Subalgoritm numar_asociat(arb, e, rez)

{Pre: arb : Arbore, e : TElem}

{Post: rez : Intreg, dacă nu există rez = 0}

Rez<- 0

Dacă arb.element[rad] != NULL_TELEM atunci

ct<-1

|<-rad

Creeaza(s){Mai jos se realizeaza parcurgerea inordine}

Cat timp vida(s) = fals sau arb.element[i] != NULL_TELEM executa

Cat timp arb.element[i] != NULL_TELEM executa

adauga(s, i)

|<-arb.stanga[i]

SfCatTimp

sterge(s, i)

Daca arb.element[i] = e atunci{daca exista elementul,

salvam indicele}

rez<-ct

SfDaca

ct<-ct+1

|<-arb.dreapta[i]

SfCatTimp

SfDaca

SfSubalgoritm

Complexitatea algoritmului este Theta(n), unde n este numarul de elemente din arbore.



+40 754 011 508

28.06.2020, 21:46

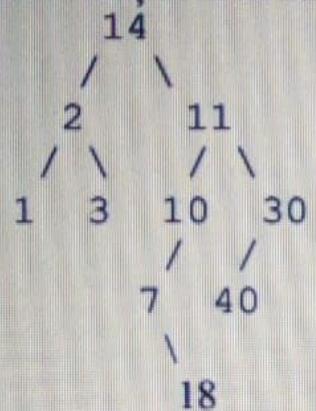


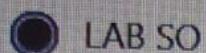
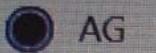
A. Deduceți timpii mediu și defavorabil pentru subalgoritmul **prelucrare**. Justificați rezultatul.

```
Functia f(n) este {:Intreg}
|   {pre: n:Intreg}
|   c←0; i ←1
|   cattimp i*i ≤ n execută c←c+1; i←i+1
|   sfcattimp
|   f ←c
Sf
subalgoritm prelucrare(n) este
|   {pre: n:Intreg}
|   S←0
|   pentru i=1, n execută S←S+f(i)
|   sfpentru
|   serie S
sfPrelucrare
```



B. Care este înălțimea nodului 11 în arborele binar de mai jos? Justificați.





C. Considerăm expresia în forma infixată: $4+3*(6*3-12)$. Presupunem că folosim o Stivă pentru a converti expresia din forma infixată în forma postfixată. Care este numărul maxim de simboluri care vor apărea în stivă la un moment dat de-a lungul conversiei? Justificați

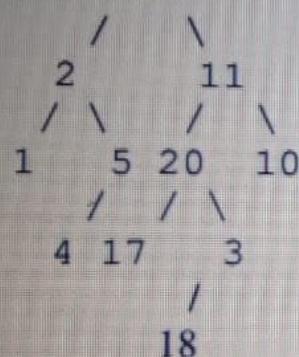
a) 1 b) 2 c) 3 d) 4 e) 5

C. Care formulă este cea mai bună aproximare a înălțimii unui ansamblu cu n noduri? Justificati

- a) \sqrt{n}
- b) n
- c) numărul de cifre ale lui n
- d) n^2
- e) $\log_2 n$

D. Să se determine nivelul pe care apare o valoare e într-un arbore ale cărui elemente sunt distincte. Arboarele se reprezintă înlinățuit, cu alocare dinamică a nodurilor. Se va folosi o procedură nerecursivă. Se va indica reprezentarea și se va preciza complexitatea operației. Folosiți comentarii pentru a ușura înțelegerea soluției.

14



Ex: Pentru arboarele de mai jos, $e = 20 \Rightarrow$ nivelul 2

R4

A. Deduceti timpii mediu si defavorabil pentru subalgoritmul prelucrare. Justificati rezultatul.

Rez: Functia f are complexitate radical din n, unde n este parametrul functiei

$$T(n) = \text{suma}(1,n)(\text{radical}(l)) = (\text{rad}(1) + \text{rad}(2) + \text{rad}(3) + \dots + \text{rad}(n))$$

Aplicam inegalitatea lui Cauchy-Schwarz:

$$(\text{suma}(1,n)(\sqrt{l} * l))^2 \leq \text{suma}(1,n)(l) * \text{suma}(1,n)(1)$$

$$t(n)^2 \leq n(n+1)/2 * n$$

$$t(n) \leq n^{(3/2)}$$

=> Complexitatea algoritmului este $\Theta(n^{(3/2)})$

SAU

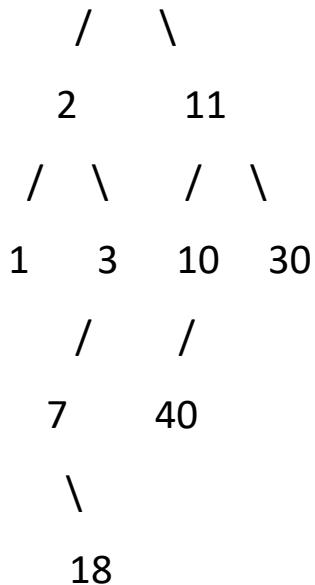
$$n = c^2$$

$$\begin{aligned} \text{Nr_pasi} &= \text{int}(\sqrt{1}) + \text{int}(\sqrt{2}) + \text{int}(\sqrt{3}) + \dots + \text{int}(\sqrt{n}) \\ &= 1 + 1 + 1 + 2 + 2 + 2 + 2 + \dots + c(2c + 1) \\ &= 1 * 3 + 2 * 5 + \dots + c * (2c + 1) \\ &= \text{suma}(1,c)(k*(2k+1)) \\ &= \text{suma}(1,c)(2k^2 + k) \\ &= 2 * c(c+1)(c+2)/6 + c(c+1)/2 \\ &= \text{nsqrt}(n) + n \end{aligned}$$

=> Complexitatea alg este $\Theta(\text{nsqrt}(n))$

B. Care este inaltimea nodului 11 in arborele binar de mai jos? Justificati.

14



Inaltimea nodului 11 este 3 pentru ca distanta maxima de la 11 la un descendant de tipul frunza este 3.

C. Consideram expresia in forma infixata $4 + 3 * (6 * 3 - 12)$. Presupunem ca folosim o Stiva pentru a converti expresia din forma infixata in forma postfixata. Care este numarul maxim de simboluri care vor aparea in stiva la un moment dat de-a lungul conversiei? Justificati

Rasp: d), vezi paint

C1. Care formula este cea mai buna aproximare a inlantuirii unui ansamblu cu n noduri? Justificati.

Rasp: e), deoarece ansamblul este un arbore aproape plin, iar inaltimea unui astfel de arbore se poate aproxima cu $\log_2(n)$

D. Sa se determine nivelul pe care apare o valoare e intr-un arbore ale carui element sunt distincte. Arboarele se reprezinta inlantuit, cu alocare dinamica a nodurilor. Se va folosi o procedura nerecursiva. Se va indica reprezentarea si se va preciza complexitatea operatiei. Folositi comentarii pentru a usura intelegerarea solutiei.

//Trebuie intrebat daca este pentru arbori binari problema sau nu

Nod:

element: TElem

st : \uparrow Nod

dr: \uparrow Nod

TAD Arbore:

rad: \uparrow Nod

Subalgoritm nivel_nod(arb, e, nivel)

{pre: arb : arbore, e : TElem}

{post: nivel : intreg, daca nu este gasit nodul, o sa retinem -1 in nivel}

nivel <- -1

Daca arb.rad != NIL atunci

 creeaza(c1){coada pentru noduri}

 creeaza(c2){coada pentru inalimi}

 nod<-rad

 adauga(c1, nod)

 adauga(c2, 0)

 CatTimp vida(c1) = fals executa

sterge(c1, elem)

sterge(c2, nivel_elem)

Daca [elem].e = e atunci

nivel<-nivel_elem

SfDaca

Daca [elem].st != NIL atunci

adauga(c1, [elem].st)

adauga(c2, nivel_elem + 1)

SfDaca

Daca [elem].dr != NIL atunci

adauga(c1, [elem].dr)

adauga(c2, nivel_elem + 1)

SfDaca

SfCatTimp

SfDaca

SfSubalgoritm

SDA

1.Complexitatea in cazul mediu si in caz defavorabil a algoritmului: (deducerea complexitatii)

Subalgoritm $f(n)$:

i=1

Cat timp $i < n$ executa

$s=s+i$

$i=i*2$

sfCattimp

sfSubalgoritm

Program A

Pentru $i=1, n$ executa

$m=2*i+1$

$s=s+f(m)$

sfPentru

parca o afisare sau ceva de genul era la final

2. Sa dai un exemplu in care sa ilustrezi operatia de dubla rotatie spre stanga + justificare.

3. era grila si cerea sa alegi inaltimea unui ansamblu + justificare

4. iti dadea $f(n)=$ suma de la 1 la n din $(3i)$ si trebuie sa spui carei complexitati apartine +justificare. Variantele erau: $O(n)$, $O(n^2)$, $O(n^4)$ si $\Theta(n^3)$.

5. operatia de adaugare intr-o coada cu prioritati reprezentata sun forma unui ansamblu cuaternar(fiecare nod poate avea 4 fii, nu 2). Reprezentare+implementare operatia+complexitatea algoritmului, fara deductie. Pre si post conditii la algoritm.

R5

1. Complexitatea in cazul mediu si in caz defavorabil a algoritmului:
(deducerea complexitatii)

Subalgoritmului $f(n)$: // Complexitatea este $\Theta(\log_2 n)$

$I = 1$

Cat timp $I < n$ executa

$S = S + I$

$I = I * 2$

SfCatTimp

SfSubalgoritm

Program A

pentru $I = 1, n$ executa

$m = 2*I + 1$

$S = S + f(m)$

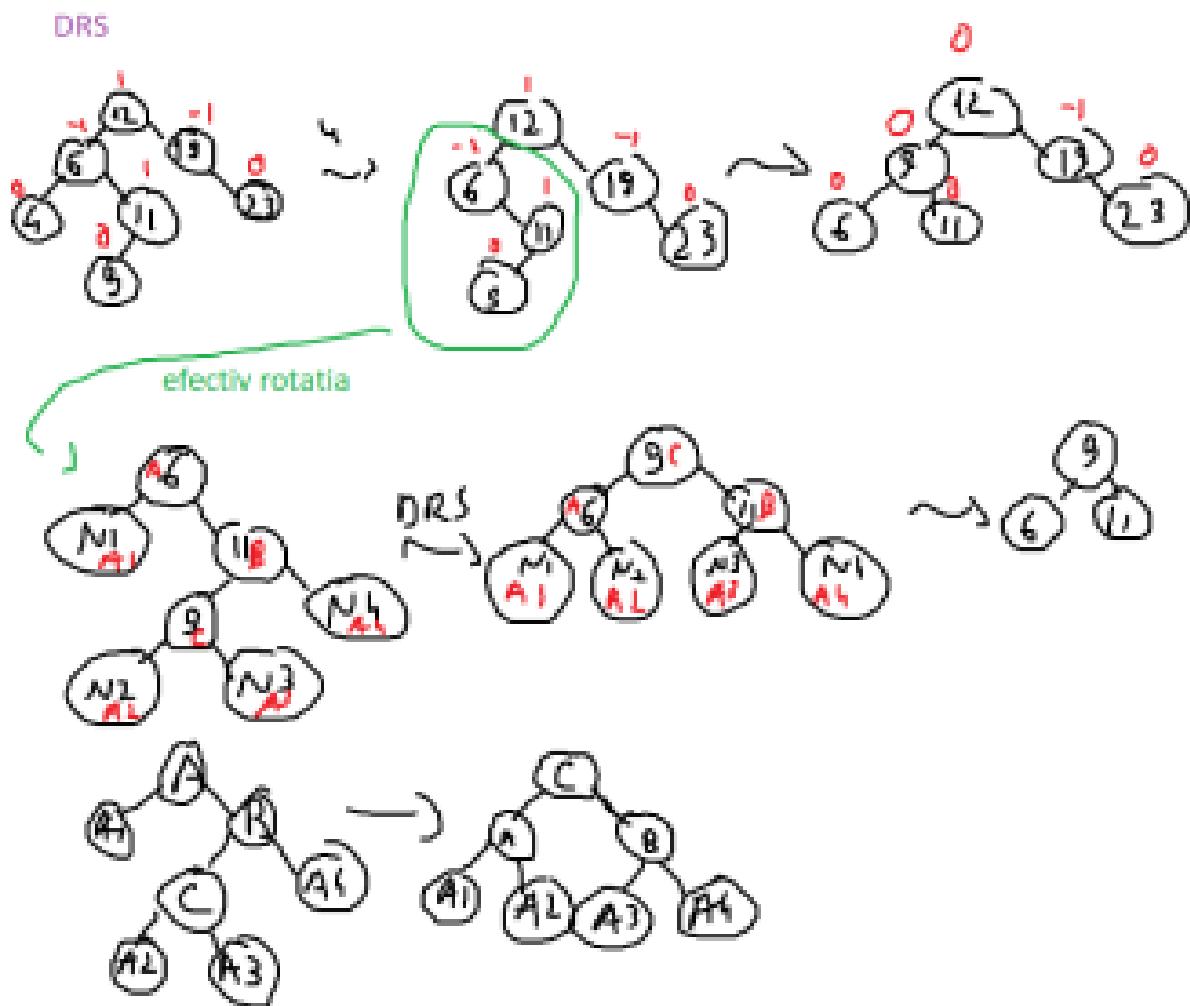
SfPentru

Parca o afisare sau ceva de genul era la final

Rezolvare:

Complexitatea algoritmului este $n \log_2 n$, rezolvarea este pe R3,
doar ca aici avem $\log 2$ nu 10

2. Sa dai un exemplu in care sa ilustrezi operatia de dubla rotatie spre stanga + justificare



3. Inaltimea unui asamblu este in general $\log_2(n)$

4. Care e complexitatea lui Suma(1,n)(3i)?

Rasp: $O(n^2)$, $O(n^4)$

$$\begin{aligned}
 \sum_{i=1}^n 3^i &= 3 \sum_{i=1}^n i = 3 \frac{n(n+1)}{2} = \frac{3}{2} n(n+1) = f(n) \Rightarrow \\
 \Rightarrow f(n) &\simeq n^2 \rightarrow \Theta(n^2); \\
 \Omega(1) &\rightarrow \Omega(n^2); \\
 O(n^2) &\rightarrow O(\infty)
 \end{aligned}$$

5. Operatia de adaugare intr-o coada cu prioritati reprezentata sub forma unui ansamblu cuaternare(fiecare nod poate avea 4 fii, nu 2). Reprezentare + implementare operatia + complexitatea algoritmului, fara educatie. Pre si post conditii la algoritm.

Ansamblu:

v: TElem[]

p : Intreg{ultima pozitie libera}

r : TRelatie

Coada:

h : ansamblu cuaternar

r : TRelatie

Subalgoritm urcare(a)

elem <- a.v[a.p]

poz <- a.p

Cat Timp poz > 1 si a.r(elem, a.v[poz / 4]) = fals executa

a.v[poz] <- a.v[poz / 4]

poz <- poz / 4

SfCatTimp

a.v[poz] <- elem

SfSubalgoritm

Subalgoritm adaugare(a, elem)

a.v[a.p] <- elem

urcare(a)

a.p<-a.p + 1

SfSubalgoritm

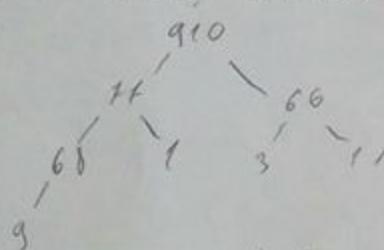
Subalgoritm adaugare(c, elem)

adaugare(c.h, elem)

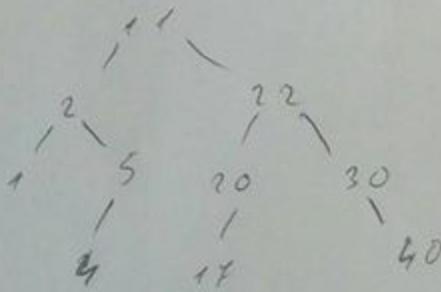
SfSubalgoritm

1. Construiți un algoritm în timpul de execuție $O(n \log n^2)$. Descrieți complexitatea
2. Repundeți la următoarele cerințe:
- 2.1. Fie o tabelă de tiperie initial vidă și locații de disponibilitate deci = $\{ \text{mod } 5 \}$, în care coloanile sunt rezervate prin înaintuire, folosind arborei AVL pentru memorarea coloanei. Arătați ce se întâmplă la inserarea valorii 25, f, 11, 6, 3, 10, 8, 5

2.2. Arătați următorul rezultat după adaugarea elementului 82. Justificați



- 2.3. Fie următorul ABC. Pp. că vom să stergem radacina și să o înlocuim cu cero din mărborele stans, care va fi arborele rezultat în urma stergerii?



3. Răspundeți ? justificăți

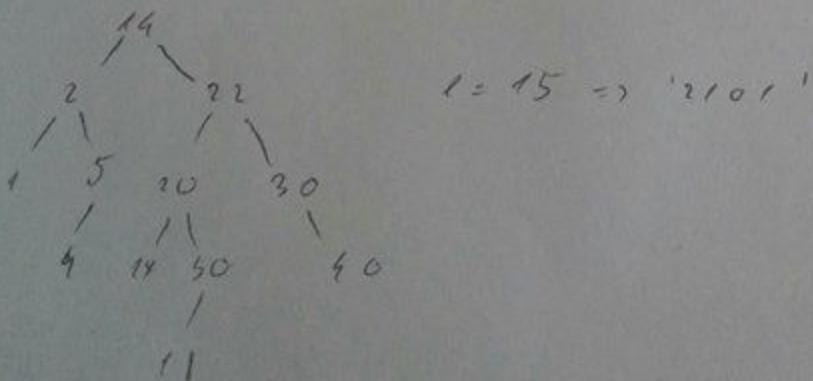
- 3.1. Repozitorul mea roță are proprietatea de urmări datele A/F

3.2. Într-o implementare a stacii folosind o listă simplă în lanțuri, unde are loc adaugarea

- 3.3. Pp. implementație intenționată a cosii, în variabilele primul și ultimul, care dintr-o același se modifică la o adâncime intr-o cale

3.4. Care este numărul minim de noduri între două lăzii care nu pot fi adăugate

- 3.5 Arbore binar. Alegeri apărute
- a) verifică - are drept de aranjare
 b) nu verifică, dar are
 c) nu este vînătoare este aranjare
 d) nu verifică dare este aranjare
- 3.6 TD în colțuri rezolvate prin adorare deschisă și verificare
nătăscă are 2024. Nu maxim de înțeles?
4. Dat odosat sub forma arbore de căutare. Specificați, reprezentând
în pseudocod, stergerea și prelezând complexitatea
5. Arbore binar cu elemente distincte. Se cere operație de
codificare a unui element e astfel nodul să se copieze în 2
descendenti și se concatenează împreună și se adaugă în
Arbore reprezentat printr-un lanțuri care să alocă locuință a nodurilor.
Operația nu să fie recursivă. Prelezând complexitatea



R6

- Construiti un algoritm in timpul de executie $O(n \cdot (\log_2 n)^2)$. Deduceti complexitatea.

Subalgoritm complexitate()

Pentru $i < 1, n$ executa // $n \cdot (\log_2 n)^2$

$j < n$

Cat timp $j \geq 1$ executa // $(\log_2 n)^2$

$J \leftarrow j / 2$

$K \leftarrow n$

Cat timp $k \geq 1$ executa // $\log_2 n$

$k \leftarrow k / 2$

SfCatTimp

SfCatTimp

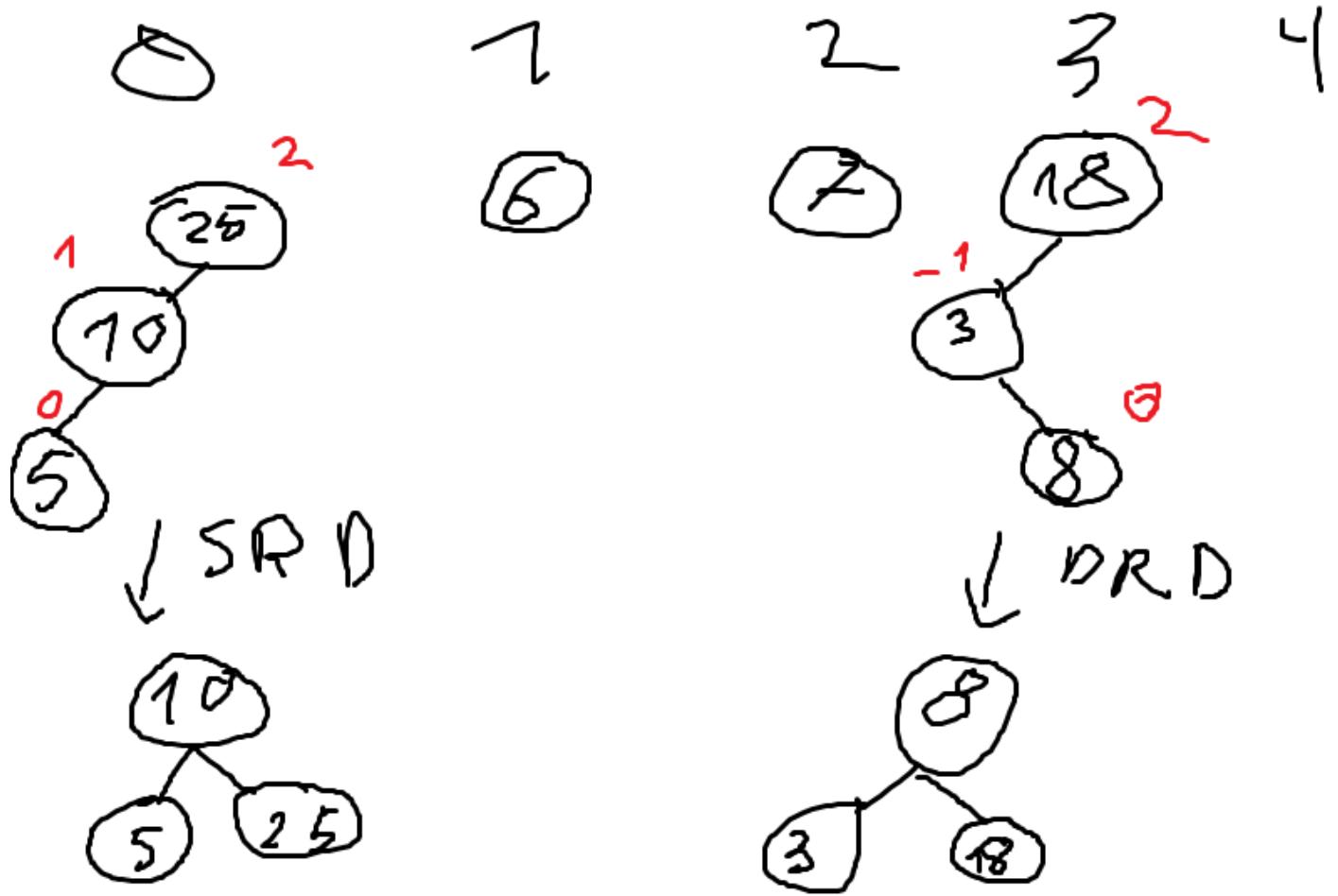
SfPentru

SfSubalgoritm

2.

2.1. Fie o tabela de dispersie initial vida cu 5 locatii de dispersie si $d(c) = c \% 5$, in care coliziunile sunt rezolvate prin inlantuire folosind arbori AVL pentru memorarea coliziunii. Aratati ce se intampla la asezarea cheilor: 25, 7, 18, 6, 3, 10, 8, 5

Cheie	25	7	18	6	3	10	8	5
D(c)	0	2	3	1	3	0	3	0



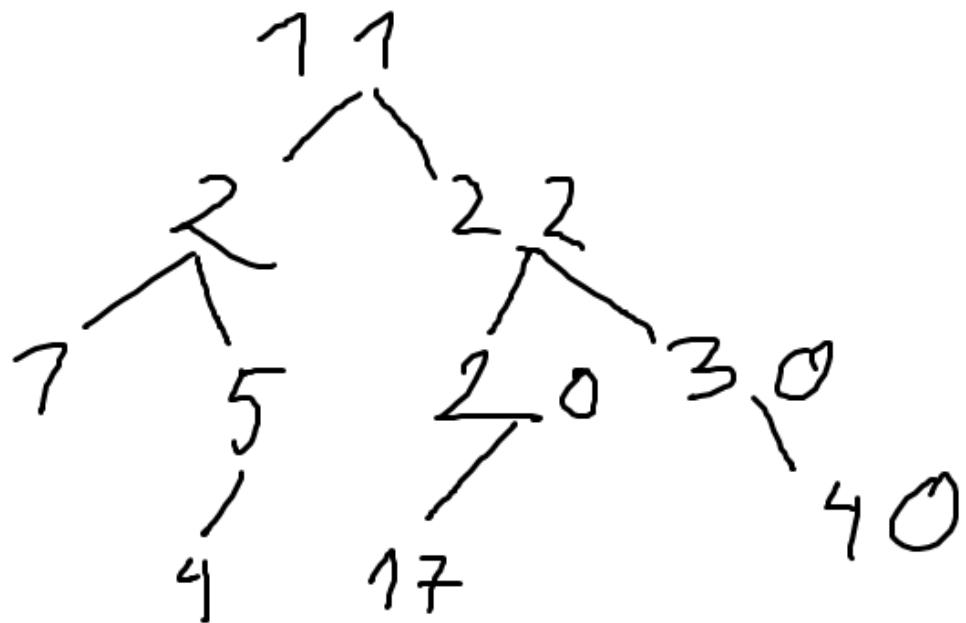
2.2 Aratati asamblul rezultat prin adaugarea elementului 82. Justificati



Ad 82, după cum se apără
urmărește



2.3 Fie urmatorul ABC pp ca vrem sa stergem radacina si sa o inlocuim cu ceva din subarborele stang, care va fi arborele rezultat in urma stergerii?



Stergem nodul 11

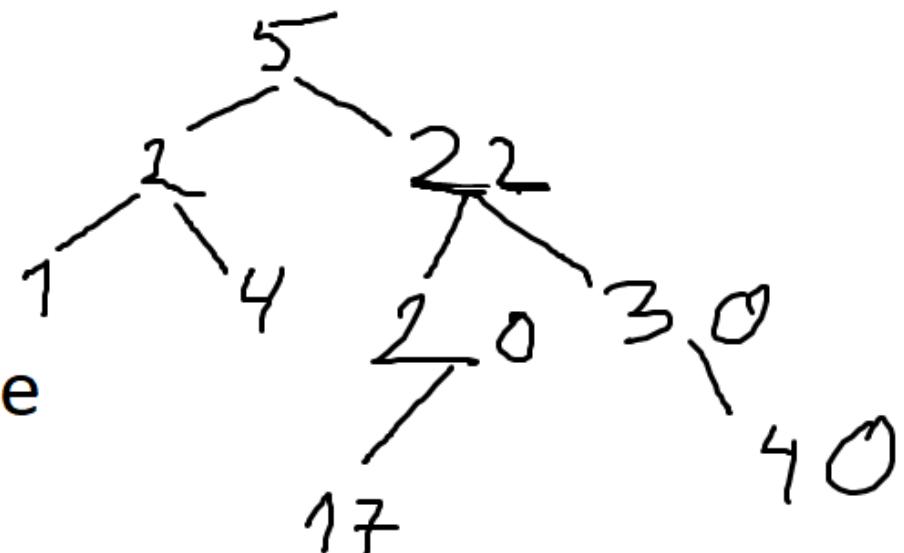
Urcam

elementul

maxim din 1

subarborele

stang



3. Raspundeti si justificati:

- Alg merge sort are proprietate de sortare stabila – A
- Intr-o implementare a stivei folosind o LSI, unde are loc adaugarea?

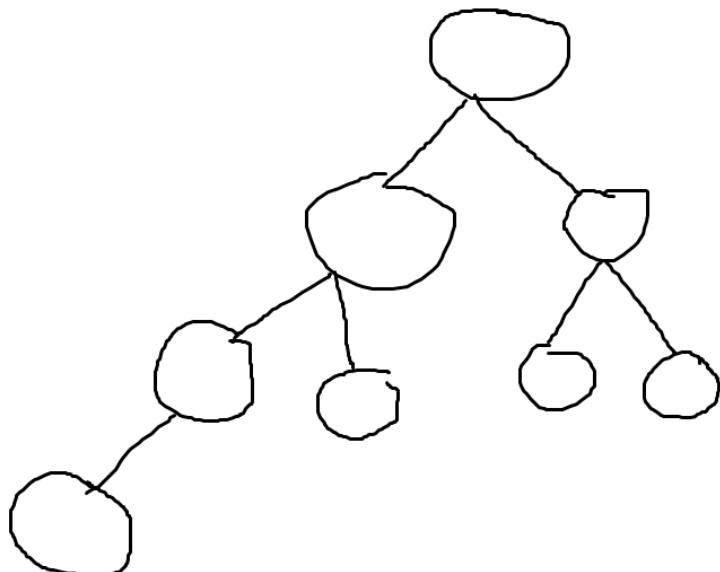
R: Adaugarea se face la inceput, principiul stivei fiind LIFO, astfel cand dorim sa extragem un element din stiva, este de fapt primul element din lista noastra.

- Presupunem implementarea inlantuita a cozii, cu 2 variabile primul si ultimul, care dintre acestea se modifica la o adaugare intr-o coada vida?

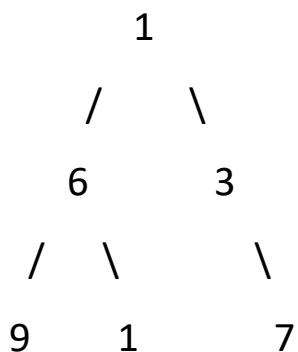
R: In cazul in care dorim sa adaugam un element in coada noastra care este vida, se vor modifica ambele campuri.

- Care este numarul minim de noduri intr-un arbore binar, aproape plin de adancime 3

R: $\log_2(n) = 3 \Rightarrow n = 8$



- Arbore binar. Alegeti afirmatiile



A) verifica si are struc de ansamblu – F

B) nu are structura de ansamblu si nu verifica proprietatea de ansamblu - A

C) nu este ansamblu – A

D) este ansamblu – F

- TD cu coliziuni rezolvate prin adresare deschisa si verificare patratica are 1024. Nr maxim de intrari?
R: 1024 de intrari pentru ca putem adauga elementele de la 0 la 1023
- Dictionar ordonat sub forma ABC. Specificatia, reprezentarea si descriere in pseudocod stergerea si precizati complexitatea.

Dictionar ordonat:

a : abc

Functie sterge_rec(p, e)

Daca p = NIL atunci

sterge_rec<-NIL

altfel

Daca e.c < [p].e.c atunci

[p].st<-sterge_rec([p].st, e)

sterge_rec<-p

altfel

Daca e.c > [p].e.c atunci

[p].dr<-sterge_rec([p].dr, e)

sterge_rec<-p

altfel

Daca [p].st != NIL si [p].dr != NIL atunci

temp<-minim([p].dr)

[p].e<-[temp].e

[p].dr<-sterge_rec([p].dr, [p].e)

sterge_rec<-p

altfel

temp<-p

Daca [p].st = NIL atunci

repl<-[p].dr

altfel

repl<-[p].st

SfDaca

dealloca(temp)

sterge_rec<-repl

SfDaca

SfDaca

SfDaca

SfFunctie

Subalgoritm stergere(d, e)

d.a.rad = sterge_rec(d.a.rad, e)

SfSubalgoritm

5. Arboare binar cu elemente distincte. Se cere operatia de codificare a unui element astfel incat radacina se codifica cu 2, descendantul stang se concateneaza cu 0 iar cel drept cu 1. Arboare reprezentat prin inlantuire cu alocare dinamica a nodurilor. Operatia sa fie nerecursiva. Precizati complexitatea.

R: daca intelege cineva pb sa ne spuna si noua

ex: 1 = 15 => '2101'

Rând 7

1. Analizați timpii mediu și defavorabil pentru următorul algoritm. Justificați.

function F(n)
Require: n: Întreg

s \leftarrow 0

m \leftarrow n

while m \neq 0 **do**

s \leftarrow s+[m/10]

m \leftarrow m/10

end while

f \leftarrow s

end function

procedure A

s \leftarrow 0

for i = 1,n **do**

m \leftarrow 2i + 1

s \leftarrow s+f(m)

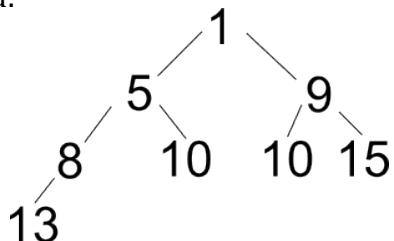
end for

@ scrie s

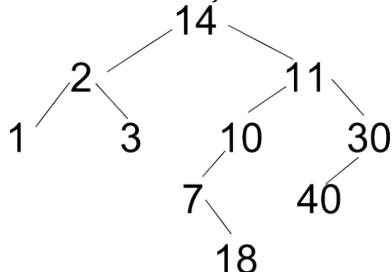
end procedure

2.1. Fie o tabelă de dispersie inițial vidă, cu 5 locații și funcția $d(c) = c \bmod 5$ în care coliziunile sunt rezolvate prin înlănțuire folosind arbore AVL pentru memorarea coliziunilor. Ce se întâmplă la inserarea 35, 2, 18, 6, 3, 10, 8, 5.

2.2. Scrieți ansamblul rezultat la inserarea valorii 3 în următorul ansamblu:



2.3. Care e înălțimea nodului 11 în arborele:



3. Răspundeți, justificând.

3.1. Inserarea unui element într-un vector ordonat x_1, \dots, x_n se poate face în: a) $O(\log_2 n)$ b) $O(n)$ c) $\theta(n)$ d) $\theta(\log_2 n)$

3.2 Considerați $4 + 3 * (6 * 3 - 12)$. Folosiți o stivă pentru a converti expresia în formă postfixată. Care este numărul maxim de simboluri care vor apărea la un moment dat în stivă?

- a) 1 b) 2 c) 3 d) 4 e) 5

3.3. Care metodă de acces definește o Coadă?

- a) FIFO b) LIFO c) FILO d) HPOF

3.4 Fie un arbore binar cu 18 noduri. Care e adâncimea minimă?

- a) 1 b) 2 c) 3 d) 4 e) 5

3.5 Care afirmații sunt adevărate?

a) orice ansamblu este un ABC b) orice arbore binar de căutare are structura de ansamblu c) orice ABC verifică proprietatea de ansamblu d) niciuna

3.6. Ce tip de inițializare trebuie pentru o tabelă de dispersie cu adresare deschisă?

a) nici una b) cheile la fiecare poziție în vector trebuie inițializate c) primul element din fiecare listă înlănțuită trebuie setat pe NIL d) legătura fiecărei locații din tabelă trebuie setată nulă

4. Alegeti o SD pentru implementarea unui Dicționar Ordonat astfel încât operația de căutare să aibă complexitate minimă. Pseudocod. Precizați complexitatea operației.

5. Operația de adăugare într-un ABC. Arborele se reprezintă înlănțuit cu alocare dinamică. Fiecare nod va memora: adâncimea nodului, referința către cei 2 subarbore și către părinte, se va folosi o procedură nerecursivă. Se va preciza complexitatea operației.

- Analizati timpii mediu si defavorabil pentru urmatorul algoritm. Justificati.

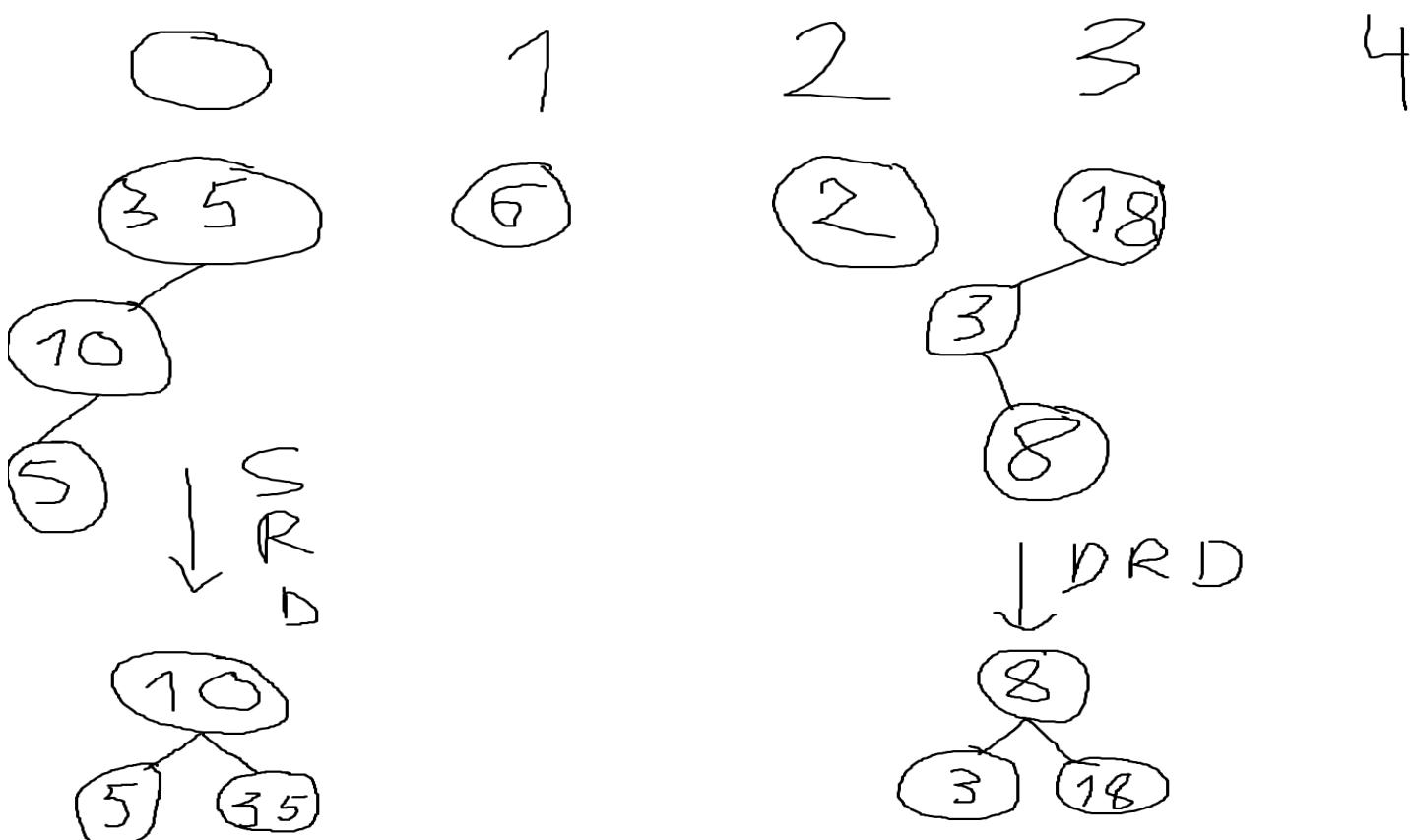
Rez: $\text{Suma}(1,n) \log_{10}(2i+1)$

Complexitatea este $\Theta(\log_{10}((2n+1)! / (2^n * n!))) = n \log n$

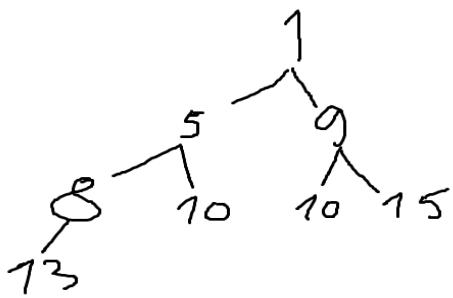
Rez la R3, dem pe paint de Mita

- Fie o tabela de dispersie initial vida, cu 5 locatii si functia $d(c) = c \% 5$ in care coliziunile sunt rezolvate prin inlantuire folosind arbore AVL pentru memorarea coliziunilor. Ce se intampla la inserarea 35,2,18,6,3,10,8,5

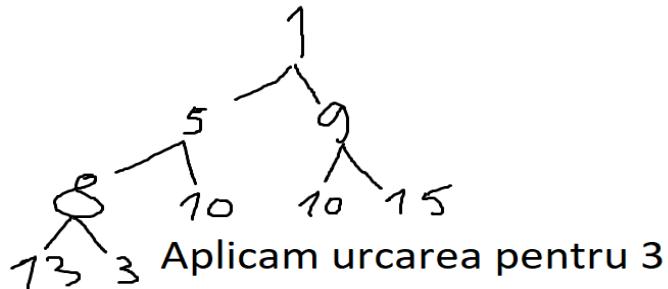
Cheie	35	2	18	6	3	10	8	5
$D(c)$	0	2	3	1	3	0	3	0



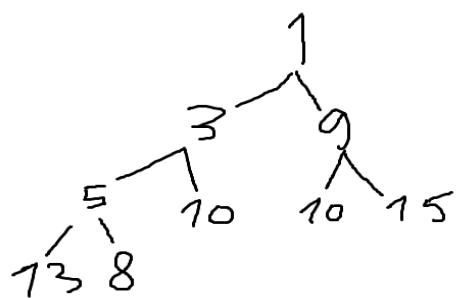
2.2 Scrieti ansamblul rezultat la inserarea valorii 3 in urmatorul ansamblu:



Inseram valoarea 3



Aplicam urcarea pentru 3



2.3 Care este inaltimea nodului 11 in arborele:

14

/ \

Inaltimea nodului 11 este 3.

2 11

/ \ / \

1 3 10 30

/ /

7 40

\

18

3.

- Inserarea unui element intr-un vector ordonat x_1, \dots, x_n se poate face in
b) $O(n)$
- Considerati $4+3*(6*3-12)$. Folositi o stiva pentru a converti expresia in forma
postfixata. Care este numarul maxim de simboluri care vor aparea la un
moment dat in stiva?
D) 4 – vezi paint Mita
- Care metoda de acces defineste o Coada?
A) FIFO
- Fie un arbore binar cu 18 noduri. Care e adancimea minima?
 $\text{int}(\log_2(18)) = 4 \Rightarrow d$
• Care afirmatii sunt adevarate?
D) niciuna
- Ce tip de initializare trebuie pentru o tabela de dispersie cu adresare directa?
C) primul element din fiecare lista inlantuita trebuie setat pe NIL

4. Alegeti o SD pentru implementarea unui DO a.i. operatia de cautare sa aiba complexitatea minima. Pseudocod. Precizati complexitatea operatiei.

DO:

A : abc{folosim un avl pentru a eficientiza operatia de cautare}

R: TRelatie : TCheie x TCheie $\rightarrow \{A / F\}$

Nod:

E : TElem

Dr : \uparrow Nod

St : \uparrow Nod

TElem:

C : TCheie

Val : TValoare

Subalgoritm cauta(d, c, v)

nod <- d.a.rad

V<-NULL_TVALOARE

Daca nod != NIL atunci

Cat timp nod != NIL executa

Daca [nod].e.c = c atunci

V <- [nod].e.val

SfDaca

Daca d.r(c, [[nod].st].c) = adevarat atunci

Nod<-[nod].st

Altfel

Nod<-[nod].dr

SfDaca

SfCatTimp

SfDaca

SfSubalgoritm

5. Operatia de adaugare intr-un ABC. Arboarele se reprezinta inlantuit cu alocare dinamica. Fiecare nod va memora: adancime nodului, referinta catre cei 2 subarbori si catre parinte, se va folosi o procedura nerecursiva

Nod:

E : TElemenT

St : \uparrow Nod

Dr: \uparrow Nod

Adancime : intreg

Parinte: \uparrow Nod

ABC:

Rad : Nod

Subalgoritm adauga(a, elem)

Nod_curent<-a.rad

Aloca(nod)

[nod].e <- elem

[nod].st <- NIL

[nod].dr<-NIL

[nod].adancime<-0

[nod].parinte<-NIL

Daca a.rad = NIL atunci

a.rad <- nod

Altfel

Cat Timp nod_curent != NIL executa

Daca [nod].e < [nod_curent].e atunci
 [nod].parinte <- nod_curent
 [nod].adancime <- [nod_curent].adancime + 1
 nod_curent <- [nod_curent].st

Altfel

 [nod].parinte <- nod_curent
 [nod].adancime <- [nod_curent].adancime + 1
 nod_curent <- [nod_curent].dr

SfDaca

SfCatTimp

 nod_curent <- nod

SfDaca

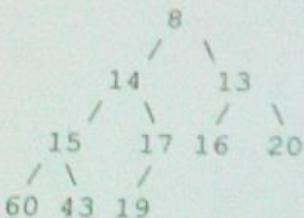
SfSubalgoritm

1. Scrieti un subalgoritm care sa aiba timpul de executie dat de urmatoarea recurenta. Dedugeti complexitatea.

$$T(n) = \begin{cases} 0 & n = 1 \\ 4T\left(\frac{n}{2}\right) + O(n^2) & altfel \end{cases}$$

2. Raspundeti la urmatoarele cerinte:

- 2.1 Fie TD cu coliziuni rezolvate prin liste intrepatrunse, cu 13 locatii, functia de dispersie prin divizune, rezultata in urma cheilor 18, 31, 29, 28, 26, 44, 39. Aratati tabela rezultata in urma stergerii cheii 18. Gestioneaza spatiul liber se face de la dreapta (de la 0 spre 12).
- 2.2 In ansamblul de mai jos construit cu relatia \leq , se aplica de doua ori operatia de stergere. Indicati ansamblul rezultat (se si pasul intermediar dupa prima stergere).



- 2.3 Ilustrati pe un exemplu concret operatia de dubla rotatie spre dreapta intr-un arbore AVL.

3. Raspundeti la urmatoarele intrebari, justificand rezultatul (rezultatele) ales (alese).

- 3.1 Un vector x_1, \dots, x_n de numere intregi cuprinse in intervalul [10, 2000] pot fi sortate crescator folosind BucketSort in

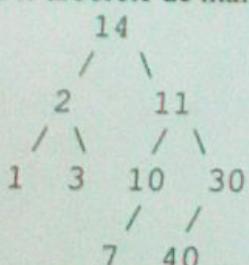
- a) $O(n)$ b) $\theta(n)$ c) $\theta(n^2)$

- 3.2 In implementarea Stivei folosind o lista inlantuita, ce operatii necesita timp liniar in cazul defavorabil?
- a) vida b) accesare (operatia element) c) stergere d) adaugare e) nici una din operatiile mentionate

- 3.3 Care este valoarea urmatoarei expresii in forma postfixata 6 3 2 4 + - *:

- a) o valoare intre -15 si -100 b) o valoare intre -5 si -15 c) o valoare intre 5 si -5 d) o valoare intre 5 si 15
e) o valoare intre 15 si 100

- 3.4 Fie arborele de mai jos.



Care este preordinea arborelui?

- a) 1 2 3 7 10 11 14 30 40 b) 1 2 3 14 7 10 11 40 30 c) 1 3 2 7 10 40 30 11 14 d) 14 2 1 3 11 10 7 30 40

- 5 Avand un arbore binar de cautare cu n elemente, elementele acestuia se pot afisa in ordine crescatoare in $\theta(n)$

- a) adevarat b)fals

- 6 O TD cu coliziuni rezolvate prin liste intrepatrunse are 512 locatii. Care este numarul maxim de intrari care pot fi efectuate in tabela?

- a) 256 b) 511 c) 512 d) 1024 e) oricat

se translateze o expresie aritmetica cu paranteze corecta din forma infixata in forma postfixata. Operatorii sunt binari : +, -, *, /. Ex : $(1+2)*3 \Rightarrow 1\ 2\ +\ 3\ *$

7 Descrieti operatia de dubla rotatie spre stanga pentru reechilibrire intr-un Arbore Binar de Cautare. Arborele se va rebalanceaza dinamica a nodurilor. Descrieti in Pseudocod subalgoritmul. Precizati complexitatea operatiei.

1) Scrieti un subalgoritm care sa aiba timpul de executie dat de urmatoarea recurenta. Deduceti complexitatea

$$T(n) = \begin{cases} 0, & n = 1 \\ 4T(n/2) + O(n^2) & \text{altfel} \end{cases}$$

Subalgoritm complexitate(n)

Daca $n > 1$ atunci

$s \leftarrow 0$

Pentru $i \leftarrow 1, n$ executa

Pentru $j \leftarrow 1, i$ executa

$s \leftarrow s + j$

SfPentru

SfPentru

complexitate($n/2$)

complexitate($n/2$)

complexitate($n/2$)

complexitate($n/2$)

SfDaca

SfSubalgoritm

$$T(n) = 4T(n/2) + n^2$$

$$T(n/2) = 4T(n/4) + (n/2)^2$$

....

$$T(2) = 4T(1) + 2^2$$

$$N = 2^k$$

$$T(2^k) = 4T(2^{k-1}) + 2^{2k} = 4T(2^{k-1}) + 4^k$$

$$T(2^{k-1}) = 4T(2^{k-2}) + 2^{2(k-1)} = 4T(2^{k-2}) + 4^{k-1} \quad | * 4$$

$$T(2^{k-2}) = 4T(2^{k-3}) + 2^{2(k-2)} = 4T(2^{k-3}) + 4^{k-2} \quad | * 4^2$$

....

$$T(2) = 4T(1) + 2^2 = 4T(1) + 4 \quad | * 4^{k-1}$$

$$T(2^k) = 4^k * k$$

$$T(n) = n^2 * \log_2(n)$$

Complexitatea algoritmului este $O(n^2 * \log_2(n))$

2. Raspundeti la urmatoarele cerinte:

- Fie TD cu coliziuni rezolvate prin liste intrepatrunse, cu 13 locatii, functia de dispersie prin diviziune, rezultata in urma cheilor 18, 31, 29, 28, 26, 44, 39. Aratati tabela rezultata in urma stergerii cheii 18. Gestiunea spatiului liber(se face de la stanga la dreapta)

Cheie	18	31	29	28	26	44	39
D(c)	5	5	3	2	0	5	0

$m = 13$

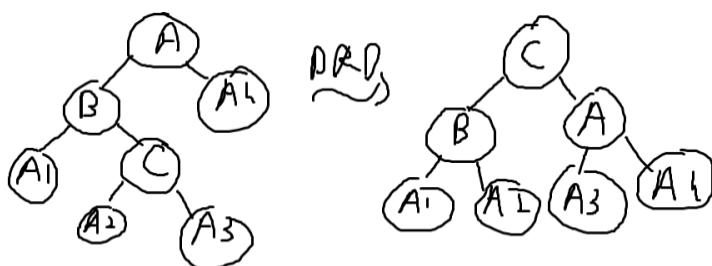
Ind	0	1	2	3	4	5	6	7	8	9	10	11	12
Che	31	26	28	29	44	18	39	-1	-1	-1	-1	-1	-1
Urm	1	4	-1	-1	6	0	-1	-1	-1	-1	-1	-1	-1

PrimLiber = 7

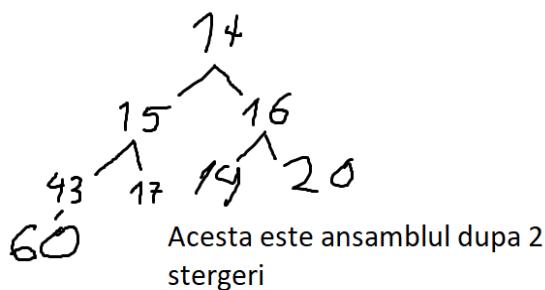
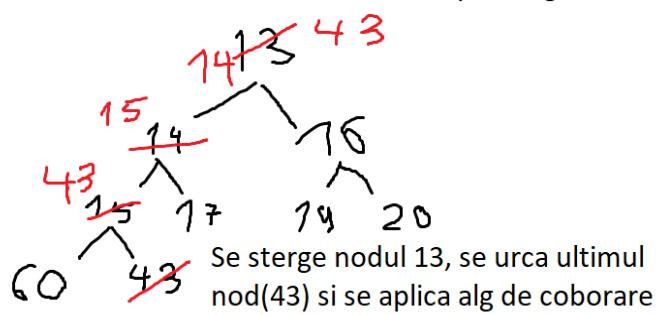
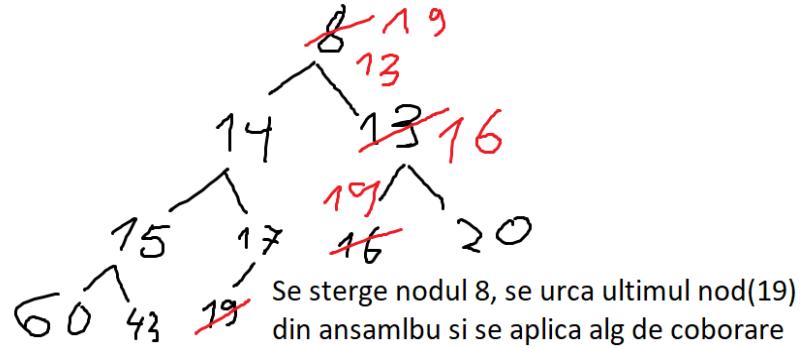
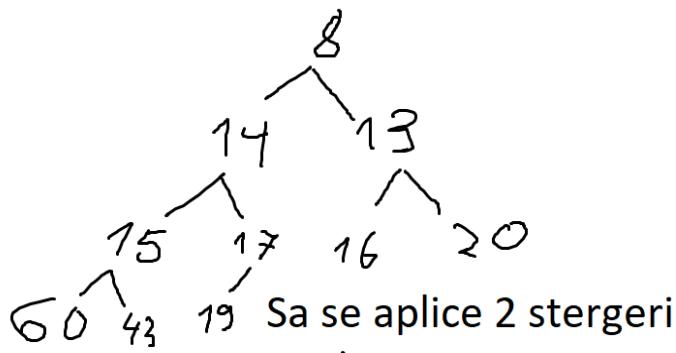
Asa arata tabela noastra dupa ce adaugam toate elementele
Urmeaza sa stergem elementul 18.

Ind	0	1	2	3	4	5	6	7	8	9	10	11	12
Che	26	-1	28	29	44	31	39	-1	-1	-1	-1	-1	-1
Urm	4	-1	-1	-1	6	0	-1	-1	-1	-1	-1	-1	-1

- Ilustrati pe un exemplu concret operatia de dubla rotatie spre dreapta intr-un arbore avl



- In ansamblul de mai jos construit cu realtia \leq , se aplica de 2 ori operatia de stergere. Indicati ansamblul rezultat(se reprezinta toate etapele stergerii)



3. Raspundeti la urmatoarele intrebari justificand rezultatul(rezultatele alese):

- Un vector x_1, \dots, x_n de numere intregi cuprinse in intervalul $[10, 2000]$ pot fi sortate crescator folosind BucketSort in

R: $O(n)$ si $\Theta(n)$

- În implementarea stivei folosind o LI, ce operații necesită timp liniar în cazul defavorabil?

R: e) nici una din operațiile menționate pentru ca toate operațiile menționate anterior au complexitate Theta(1)

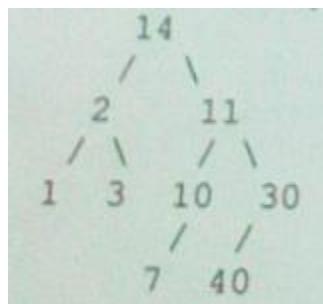
- Care este valoarea următoarei expresii în forma postfixată $6324+-*?$

R:

$$(2 + 4 - 3) * 6 = (6 - 3) * 6 = 18$$

=> e)

- Fie arboarele de mai jos. Care este preordinea arborelui(RSD)?



=> 14, 2, 1, 3, 11, 10, 7, 30, 40 d)

- Având un arbore binar de căutare cu n elemente, elementele acestuia se pot afisa în ordine crescătoare în $\Theta(n)$

R: Parcurgerea inordine se realizeaza în $\Theta(n)$ => Adevarat

- O TD cu coliziuni rez prin liste întrepatrunse are 512 locuri. Care este numărul maxim de intrări pe care le poate contine tabela?

R: c)

4. Sa se translateze o expresie aritmetică cu paranteze corectă din forma infixată în forma postfixată. Operatorii binari sunt: +, -, *, /.

Ex: $(1+2)*3 \Rightarrow 12+3*$

$$(3+5)*(7-2*3)/87$$

$$\Rightarrow 35+723*-*87/$$

5. Scrieti operatia de dubla rotatie spre stanga pentru reechilibrare intr-un ABC. Arborele se reprezinta cu alocare dinamica a nodurilor. Descrieti in Pseudocod subalgoritmul. Precizati complexitatea.

Nod:

e: TElem

st: \uparrow Nod

dr: \uparrow Nod

ABC:

rad: \uparrow Nod

Functie DRS(a, p) // in cazul de fata nu e nevoie sa trimitem si arborele
{pre: sa existe p, sa aiba descendant drept, iar descendantul drept al nodului primit sa aiba un descendant stang}
{post: rad: sageata sus Nod}

A <- p

B <- [p].dr

C <- [B].st

[A].dr <- [C].st

[B].st <- [C].dr

[C].st <- A

[C].dr <- B

DRS <- C

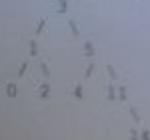
SfSubalgoritm

Complexitatea este Theta(1)

1. Scrieti un subalgoritm necursiv pentru a inversa o lista simplu inlantuita cu n elemente (se cunoaste referinta spre primul element al listei) in $\theta(n)$. Spatiul suplimentar de memorare va fi $\theta(1)$. Justificati complexitatele algoritmului.

2. Raspundeti la urmatoarele cerinte:

- * 2.1 Fie TD cu coliziuni rezolvate prin liste intrepatrunse, cu 9 locatii, functia de dispersie prin divizuire, rezultata in urma inserarii cheilor 5, 28, 19, 15, 20, 33, 18, 17, 9. Aratati ce rezulta in urma stergerii cheii 28? Gestuirea spatiului liber se face de la stanga la dreapta (de la 0 spre 8).
- 2.2 Vrem sa creem un ansamblu binar continand cheile DATASTRUCTURE (comparatiile folosesc ordinea alfabetica). Aratati care este ansamblul binar rezultat in urma operatiilor succesive de adaugare (incepand cu cheia D).
- * 2.3 Cum arata arborele AVL de mai jos in urma operatiei inserare a cheii 12? Ce operatie se aplica pentru reechilibrare?



3. Raspundeti la urmatoarele intrebari, justificand rezultatul (rezultatele) alese (alese).

- * 3.1 Care este cea mai mica valoare a lui n astfel incat un algoritm cu timpul de executie $10 \cdot n^3$ este mai rapid decat un algoritm cu timpul de executie $5 \cdot 2^{n+7}$? Justificati alegerea.

a) 2 b) 4 c) 9 d) 8

- * 3.2 Presupunem o colectie implementata folosind o lista inlantuita. Care din aceste operatii au complexitatea defavorabila $\theta(1)$?

a) adaugare b) stergere c) numarul operatiilor

- * 3.3 Presupunem ca avem o reprezentare incrementală circulară a unei Cozi, cu 10 elemente memorate de la data[2] pana la data[11]. Capacitatea maxima a cozii este 42. Vrem sa adaugam in coada un nou element. Unde se va adauga acest element in vector?

a) data[11] b) data[2] c) data[11] d) data[12]

- * 3.4 Care este numarul minim de noduri intr-un arbore binar pînă de adâncime 3?

a) 3 b) 4 c) 8 d) 11 e) 15

- * 3.5 Se considera urmatorul arbore binar. Alegeti afirmatiile corecte. Justificati.



a) verifica proprietatea de ansamblu, dar nu are structura de ansamblu b) nu verifica proprietatea de ansamblu, dar are structura de ansamblu c) nu este un ansamblu

- 3.6 Care dintre afirmatiile de mai jos NU este sugerata din studii teoretice sau experimentale pentru tabele de dispersie:

a) dimensiunea tabelei trebuie sa fie produsul a doua numere prime b) dimensiunea tabelei trebuie sa fie mai mare decat produsul a doua numere prime c) dimensiunea tabeliei trebuie sa fie de forma $4K+3$ d) dimensiunea tabelei trebuie sa nu fie aproape de o putere a lui 2

- * 4. Scrieti un algoritm pentru a verifica daca parantezele dintr-o secventa de paranteze se inchid corect - ex: (()) (() ()). Veti folosi o stiva auxiliara, fara a face apel la reprezentarea stivei, folosind doar operatiile din interfata stivei.

5. Descrieti operatia de singura rotatie spre dreapta pentru reechilibrare intr-un Arbore Binar de Cautare. Arborele se reprezinta in lantul cu alocare dinamica a nodurilor. Decrieti in Pseudocod subalgoritmul. Precizati complexitatea operatiei.

1. Scrieti un subalgoritm nerecursiv pentru a inversa o lista simplu inlantuita cu n elemente(se cunoaste referinta spre primul element al listei) in Theta(n). Spatiul suplimentar de memorare va fi Theta(1). Justificati complexitatea algoritmului.

Nod:

$e : TElement$

$urm : \uparrow \text{Nod}$

LSI:

$\text{prim} : \uparrow \text{Nod}$

Subalgoritm inversare(l)

{pre: $l : \text{LSI}$ }

{post: inverseaza o lista simplu inlantuita cu n elemente }

precedent $\leftarrow \text{NIL}$

Daca $l.\text{prim} \neq \text{NIL}$ atunci

 current $\leftarrow l.\text{prim}$

Cat Timp current $\neq \text{NIL}$ executa

 urmator $\leftarrow [\text{current}].\text{urm}$

$[\text{current}].\text{urm} \leftarrow \text{prec}$

$\text{prec} \leftarrow \text{current}$

 current $\leftarrow \text{urmator}$

SfCatTimp

SfDaca

SfSubalgoritm {Complexitate Theta(n)}

2. Raspundeti la urmatoarele cerinte:

2.1 Fie TD cu coliziuni rezolvate prin liste intrepatrunse cu 9 locatii, functia de dispersie prin diviziune, rezultatul in urma inserarii cheilor 5,28,19,15,29,33,18,17,9. Aratati ce rezulta in urma stergerii cheii 28? Gestiunea spatiului liber se face de la stanga la dreapta.

Cheie	5	28	19	15	29	33	18	17	9
D'(c)	5	1	1	6	2	6	0	8	0

M = 9

Ind	0	1	2	3	4	5	6	7	8
Cheie	19	28	29	33	18	5	15	9	17
Urm	4	0	-1	-1	7	-1	3	-1	-1

PrimLiber = -1

Trebuie sa stergem elementul cu, cheia 28

Ind	0	1	2	3	4	5	6	7	8
Cheie	18	19	29	33	-1	5	15	9	17
Urm	7	0	-1	-1	-1	-1	3	-1	-1

PrimLiber = 4

2.2 Vrem sa creem un ansamblu binar continand cheile D A T A S T R U C T U R E(comparatiile folosesc ordinea alfabetica). Aratati care este ansamblul binar rezultat in urma operatiilor successive de adaugare(incepand cu cheie D).

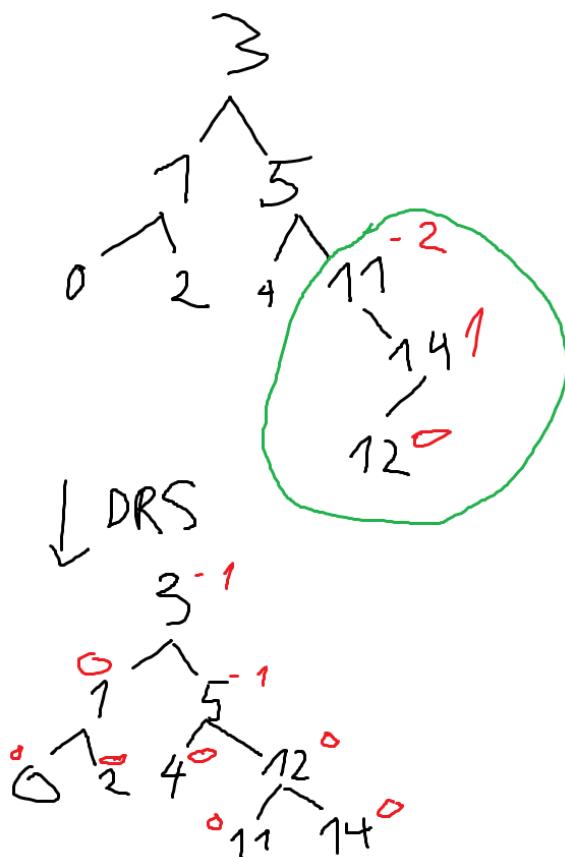
Elem: D A T A S T R U C T U R E

~~D A T A S T R U C T U R E~~



A: AAECSRTUDTUTR

2.3 Cum arata arborele AVL de mai jos in urma operatiei de inserare a cheii 12? Ce operatie se aplica pentru reechilibrare?



3. Raspundeti la urmatoarele intrebari, justificand rezultatul(resultatele) ales(alese)

3.1 Care este cea mai mica valoare a lui n astfel incat un algoritm cu timpul de executie $10 * (n^2)$ este mai rapid decat un algoritm cu timpul de executie $5 * 2^{(n-1)}$? Justificati alegerea

R: c)

3.2 Pp o Colectie implementarea folosind o LI. Care din aceste operatii au complexitatea defavorabila Theta(1)?

R: a)

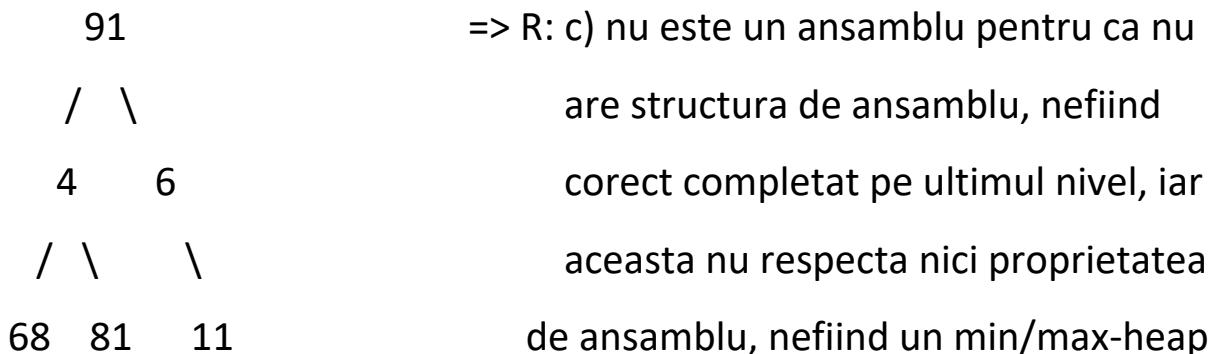
3.3 Pp ca avem o reprezentare sequentiala circulara a unei Cozi, cu 10 elemente memorate de la data[2], pana la data[11]. Capacitatea maxima a cozii este 42. Vrem sa adaugam in coada un nou element. Unde se va adauga acest element in vector?

R: d)

3.4 Care e numarul minim de noduri intr-un arbore binare plin de adancime 3?

R: $2^4 - 1 = 15 \Rightarrow$ e)

3.5 Se considera urmatorul arbore binar. Alegeti afirmatiile corecte. Justificati.



3.6 Care dintre afirmatiile de mai jos NU este sugerata din studii teoretice sau experimentale pentru tabelele de dispersie:

R: b) si c)

4. Scrieti un algoritm pentru a verifica daca parantezele dintr-o secventa de paranteze se inchid corect – ex : [()]. Veti folosi o stiva auxiliara, fara a face apel la reprezentarea stivei, folosind doar operatiile din interfata acesteia.

Subalgoritm verificare_paranteze(secv, corect)

{pre: secv : coada}

{post: corect : boolean}

creeaza(s)

sterge(secv, e)

adauga(s, e)

Cat Timp vida(s) = fals si vida(secv) = fals executa

Sterge(secv, e2)

Element(s, e1)

Daca e1 = '(' si e2 = ')' atunci

Sterge(s, e1)

Altfel

Daca e1 = '[' si e2 = ']' atunci

Sterge(s, e1)

Altfel

Daca e1 = '{{' si e2 = '}' atunci

Sterge(s, e1)

Altfel

Adauga(s, e2)

SfDaca

SfDaca

SfDaca

SfCatTimp

Daca vida(s) = adevarat atunci

corect <- adevarat

Altfel

corect <- fals

SfDaca

SfSubalgoritm

5. Descrieti operatia de singura rotatie spre dreapta pentru reechilibrare intr-un ABC. Arboarele se reprezinta inlantuit cu alocare dinamica a nodurilor. Descrieti in Pseudocod subalgoritmul. Precizati complexitatea operatiei.

Nod:

E : TValoare

St: \uparrow Nod

Dr: \uparrow Nod

Functie SRD(p){Complexitate Theta(1)}

{pre: p : \uparrow Nod, p sa aiba descendant stang}

{post: rad : \uparrow Nod}

A<-p

B<-[p].st

[A].st<-[B].dr

[B].dr<-A

SRD<-B

SfFunctie

R10

Randul 10 suna in felul urmator:

1. se da o secventa random de nr (x_1, x_2, \dots, x_n) . se cere sa se scrie un subalgoritm de complexitate minima pt a determina daca exista cel putin 2 valori consecutive in secventa . De ex : 2,7,10,3,23,5 apar 2 si 3 ca valori consecutive.

2.1 se dadea o secventa si trebuia sa o adaug intr'o tabela de dispersie cu adresare deschisa folosind dispersia dubla . stiind ca $d_1(c) = c \bmod m$, $d_2(c) = 1 + (c \bmod (m-1))$

2.2 aratati ca inaltimea unui heap e $\log_2(n)$

2.3 se dadea un arbore binar la care trebuia sa scrisa postordinea , inordinea , preordinea , si latime pe ex dat +justifycare

La 3 au fost 6 grile legate de arbori , stiva si coada . Tin minte doar ca la una se cerea sa se calculeze factorul de incarcare al unei TD daca se stiu capacitatea maxima 811 si nr de elemente: 81.

4. Dictionar ordonat. Se cerea sa se gaseasca o reprezentare a acestuia pt care operatia de stergere se face cu complexitate minima . De specificat si proiectat in pseudocod operatia + complexitate.

5. Stiind preordinea si inordinea unui arbore se cere sa se descrie un subalgoritm de constructie al arborelui. + complexitate.

- Se da o secventa random de nr (x_1, x_2, \dots, x_n) . se cere sa se scrie un subalgoritm de complexitate minima pt a determina daca exista cel putin 2 valori consecutive in secventa . De ex: 2,7,10,3,23,5 apar 2 si 3 ca valori consecutive.

Subalgoritm valori_consecutive(l, rezultat)

{pre: l: Intreg[]}

{post: rezultat:boolean}

Rezultat<-fals

Creeaza(a, " \leq "){facem un min-heap}

Pentru $l < 0, n-1$ executa

adauga(a,l[l])

SfPentru

Sterge(a, el1)

Cat Timp vida(a) = fals si rezultat = fals executa

sterge(a, el2)

Daca $el2 - el1 = 1$ atunci

rezultat<-adevarat

SfDaca

$el1 <- el2$

SfCatTimp

SfSubalgoritm

3. Care este factorul de incarcare al unei tabele de dispersie?

$$\text{Alfa} = n / m$$

N – numarul de chei ce urmeaza sa fie introduse in tabela noastra

M – valoarea cu ajutorul careia se realizeaza dispersia

Ex: o tabela cu capacitatea maxima = 811 si numarul de elemente = 81

$$\Rightarrow \text{alfa} = 81 / 811 = 0.10 \text{ aproximativ}$$

4. DO, o reprezentare ideală ce duce la o complexitate minima pentru stergerea unui element. De specificat si proiectat in pseudocod operatia + complexitatea

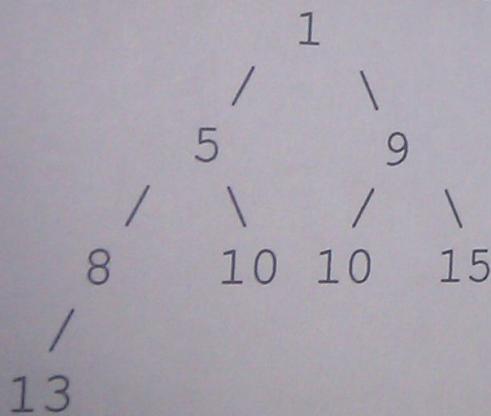
R: se foloseste un abc si se copiaza alg din curs – are complex O(h) - h e inaltimea

5. Stiind preordinea si inordinea unui arbore se cere sa se descrie un subalgoritm de constructie al arborelui. + complexitate.

R: este o solutie prea exotica pentru a putea fi descrisa

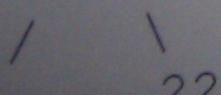
R5

1. Se considera problema de a determina daca un vector arbitrar x_1, \dots, x_n de numere intregi contine cel putin doi termeni egali $5, 1, 4, \underline{2}, 9, 1, 8, \underline{2}, 7$. Aratati ca acest fapt poate fi realizat in $\theta(n \log_2 n)$. Deduceti complexitatea algoritmului.
2. Raspundeti la urmatoarele cerinte:
 - 2.1 Fie o colectie de chei naturale si o functie de dispersie definita astfel: $d(c) = \text{numarul zecimal corespunzator bitilor } b_3, b_2, b_1$ cheii c. Ilustrati tabela in care coliziunile sunt rezolvate prin liste intrepatrunse rezultata in urma inserarii cheilor: 23, 13, 19, 34, 20, 18. Gestiunea spatiului liber se face de la stanga la dreapta.
 - 2.2 Aratati ansamblul rezultat prin stergerea unei valori din urmatorul ansamblu.



- 2.3 Fie urmatorul ABC. Presupunand ca vom sa stergem radacina si sa o inlocuim cu ceva din subarborele stang, care va fi rezultat in urma stergerii?

14



1. Se considera problema de a determina daca un vector arbitrar $x_1,..,x_n$ de numere intregi contine cel putin doi termeni egali ($5,1,4,2,9,1,8,2,7$). Aratati ca acest fapt poate fi realizat in $\Theta(n \log n)$. Deduceti complexitatea algoritmului.

R: HeapSort este aplicat si se optine complexitate ceruta in enunt

2. Raspundeti la urmatoarele cerinte:

2.1 Fie o colectie de chei naturale si o functie de dispersie definita astfel: $d(c) =$ numarul zecimal corespunzator bitilor $b_3b_2b_1b_0$ coresp cheii c . Ilustrati tabela in care coliziunile sunt rezolvate prin liste intrepatrunse rezultata in urma inserarii cheilor: $23, 3, 19, 34, 20, 18$. Gestiunea spatiului liber se face de la stanga la dreapta.

Cheie	23	3	19	34	20	18
$D'(c)$	7	3	3	2	4	2

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
c	19	18	34	3	20	-1	-1	23	-1	-1	-1	-1	-1	-1	-1	-1
u	-1	-1	1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

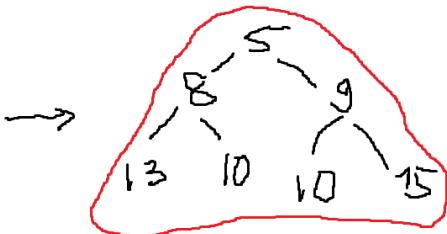
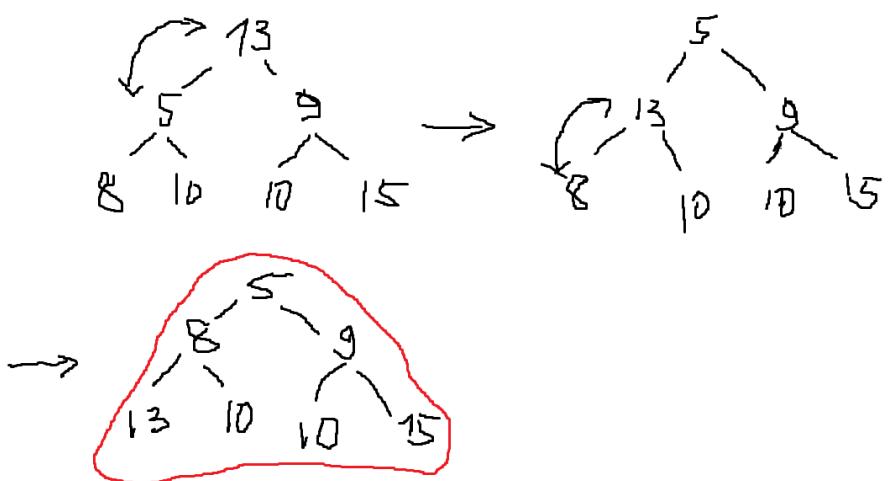
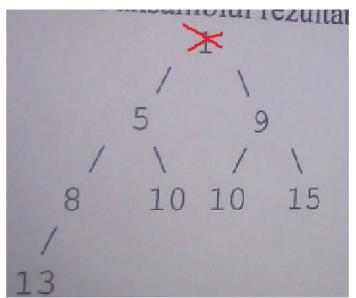
PrimLiber = 5

I = indice

C = cheie

U = urmator

2.2 Aratati ansamblul rezultat prin stergerea unei valori din urmatorul ansamblu



Structuri de date și algoritmi

- examen scris -

Notă

1. Subiectele se notează astfel: of - 1p; A – 2p; B - 1.5p; C1 - 1p; C2 – 1p; D - 3.5p.
2. Pentru cerința A, justificarea unei complexități presupune deducția acesteia.
3. Pentru cerințele B și C (C1, C2) se cer justificări, care vor fi punctate.
4. Problema de la D se va rezolva în Pseudocod. Se cer și se vor puncta: (1) descrierea ideii de rezolvare și comentarii despre soluția propusă; (2) scrierea reprezentării indicate în enunț; (3) (specificare și) implementare subalgoritm(i); (4) complexitate.

Nu se acceptă cod C++. Nu se acceptă pseudocod fără comentarii despre soluția propusă.

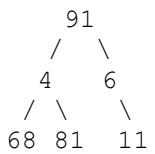
A. Deduceți timpii mediu și defavorabil pentru următorul subalgoritm. Justificați rezultatul.

Subalgoritm **S(n, i)** este

```
{pre: n:Intreg; i:Intreg}
daca n>1 atunci
|   |   i←2*i
|   |   pentru j←1,n executa i←i+1 sfpentru
|   |   m←[n/2]
|   |   daca i mod 2=0 atunci S(m, i-2)
|   |   |   altfel S(m, i-1)
|   |   sfdaca
|   |   altfel
|   |   |   scrie i
|   |   sfdaca
sfS
```

B. Care este cel mai mic, respectiv cel mai mare număr de elemente dintr-un ansmblu având înălțimea **h**? Justificați.

C. Se consideră urmatorul arbore binar. Alegeți afirmațiile corecte. Justificați.



- a) verifică proprietatea de ansamblu, dar nu are structură de ansamblu
b) nu verifică proprietatea de ansamblu, dar are structură de ansamblu
c) nu este un ansamblu

C. Care este cazul defavorabil pentru căutare secvențială într-un vector? Justificati

- a) timp constant b) timp logaritmic c) timp liniar d) timp pătratic

D. Descrieți operația de dublă rotație spre dreapta pentru reechilibrare într-un Arbore Binar de Căutare. Arboarele se reprezintă înlănit cu înlăturările reprezentate pe tablou. Indicați grafic situația de rotație, reprezentarea arborelui și descrieți în Pseudocod subalgoritmul. Precizați complexitatea operației. Folosiți comentarii pentru a ușura înțelegerea soluției.

A. Deducreți timpii mediu și defavorabil pentru următorul subalgoritm. Justificați rezultatul.

$$T(n) = \begin{cases} T(n/2) + n, & n > 1 \\ 1, & \text{altfel} \end{cases}$$

$$T(n) = T(n/2) + n$$

$$T(n/2) = t(n/4) + n / 2$$

$$T(n/4) = t(n/8) + n / 4$$

....

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

$$N = 2^k$$

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

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

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

....

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

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

$$= 2^{k+1} - 1$$

$$T(n) = 2n - 1$$

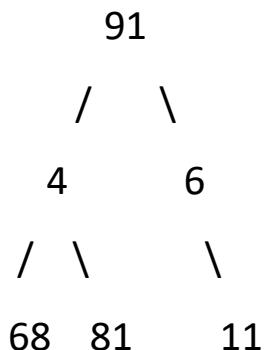
$T(n) = n \Rightarrow \Theta(n)$ este complexitatea algoritmului nostru

B. Care este cel mai mic, respectiv cel mai mare numar de elemente dintr-un ansamblu avand inaltimea h?

R: Numarul minim de elemente este 2^h deoarece avem $h-1$ nivele complete ceea ce ne duce la $2^h - 1$ elemente, dar pentru a avea un ansamblu mai trebuie sa adaugam un nod ceea ce ne conduce la 2^h noduri.

Numarul maxim de elemente este $2^{h+1} - 1$, pentru ca numarul de elemente pe un arbore binar complet de inaltime h, este $2^{h+1} - 1$.

C. Se considera urmatorul arbore binar. Alegeti afirmatiile corecte. Justificati.



Rasp: c) nu este un ansamblu deoarece nu are structura de ansamblu pentru ca pe ultimul nivel nu sunt complete pozitiile de la stanga la dreapta, nodul 6 avand descendantul stang liber, iar descendantul drept ocupat, iar acesta nu respecta proprietatea de ansamblu pentru ca nu este un min-heap sau un max-heap, elementele fiind puse intr-o ordine oarecare.

C1. Care este cazul defavorabil pentru cautarea sequentiala intr-un vector? Justificati.

Rasp: c) timp liniar deoarece cautarea sesequentiala consta in parcurgerea vectorului de la inceput pana la sfarsit.

D. Descrieti operatia de dubla rotatie spre dreapta pentru reechilibrare intr-un Arbore Binar de Cautare. Arboarele se reprezinta inlantuit cu inlantuirile reprezentate pe tablou. Indicati grafic situatia de rotatie, reprezentarea arborelui si descrieti in Pseudocod subalgoritmul. Precizati complexitatea operatiei. Folositi comentarii pentru a usura intelegherea solutiei.

Rasp:

ABC:

Elem: TElem[]

St: Intreg[]

Dr:Intreg[]

Rad:intreg

R:Relatie

Functie DRD(a, p){Complexitate Theta(1)}

{pre: a : arbore, exista radacina, exista descendantul stang al radacinii si descendantul drept al descendantului stang al radacinii}

{post: arborele a suferit o dubla rotatie spre dreapta}

A<-p

B<-a.st[A]

C<-a.dr[B]

a.dr[B] <- a.st[C]

a.st[A] <- a.dr[C]

a.st[C] <- B

a.dr[C] <- A

DRD<-C

SfFunctie

Descrierea algoritmului:

Folosind desenul de mai jos incercam sa explicam.

