Corso di Laurea in INFORMATICA

Algoritmi e Strutture Dati

MODULO 10

CODE CON PRIORITA'

Il tipo astratto coda con priorità: specifiche sintattiche e semantiche. Realizzazioni.

Questi lucidi sono stati preparati da Oronzo Altamura, Maria Francesca Costabile e Floriana Esposito, Università degli Studi di Bari, per uso didattico. Essi contengono materiale originale di proprietà dell'Università degli Studi di Bari e/o figure di proprietà di altri autori, società e organizzazioni di cui e' riportato il riferimento. Tutto o parte del materiale può essere fotocopiato per uso personale o didattico ma non può essere distribuito per uso commerciale. Qualunque altro uso richiede una specifica autorizzazione da parte dell'Università degli Studi di Bari e degli altri autori coinvolti.

CODE CON PRIORITA'

SONO UN CASO PARTICOLARE DI *INSIEME*, SUGLI ELEMENTI DEL QUALE E' DEFINITA UNA RELAZIONE "£" DI ORDINAMENTO TOTALE. E' POSSIBILE INSERIRE UN NUOVO ELEMENTO O ESTRARRE L'ELEMENTO MINIMO. LA SPECIFICA E' SIMILE A QUELLA DEL TIPO DI DATO INSIEME. LE OPERAZIONI AMMESSE SONO:

CREA, INSERISCI, MIN

E LA NUOVA OPERAZIONE CANCELLAMIN CHE RIMUOVE L'ELEMENTO MINIMO.

IL NOME DERIVA DALL'INTERPRETAZIONE CHE A E' UNA CODA DI ELEMENTI CHE DEVONO ESSERE SERVITI RISPETTANDONE LA PRIORITA'. PER SEMPLICITA' SI FA COINCIDERE LE PRIORITA' CON GLI ELEMENTI STESSI E SI ASSUMONO PRIORITA' DISTINTE IN ACCORDO ALLA SPECIFICA DEL TIPO DI DATO INSIEME.

IN GENERALE, LE PRIORITA' POSSONO INTENDERSI COME PROPRIETA' ASSOCIATE AGLI ELEMENTI CHE, TALVOLTA, POSSONO AVERE LA STESSA PRIORITA'.

ESEMPIO: PRONTO SOCCORSO

SPECIFICA SINTATTICA

TIPI: PRIORICODA, BOOLEANO, TIPOELEM

OPERATORI:

CREAPRIORICODA:	( )	→	PRIORICODA
INSERISCI:	(TIPOELEM, PRIORICODA)	→	PRIORICODA
MIN:	(PRIORICODA)	→	TIPOELEM
CANCELLAMIN:	(PRIORICODA)	→	PRIORICODA

SPECIFICA SEMANTICA

TIPI: PRIORICODA: INSIEME DI CODE CON PRIORITA' CON ELEMENTI DI TIPO TIPOELEM

OPERATORI:

CREAPRIORICODA = A

POST: A = Æ

INSERISCI (x,A) = A'

POST: A'= A È {x} (se x Î A allora A'=A)

MIN(A) = x

PRE: A ¹ Æ

POST: x Î A e x £ y per ogni y Î A

CANCELLAMIN (A) = A'

PRE: A ¹ Æ

POST: A' = A – { x } con x = MIN(A)

RAPPRESENTAZIONE CON STRUTTURE SEQUENZIALI

SI PUO' RAPPRESENTARE UNA CODA CON PRIORITA' DI n ELEMENTI UTILIZZANDO STRUTTURE SEQUENZIALI COME LISTE ORDINATE E LISTE NON ORDINATE.

TUTTAVIA, POICHE' LA CODA CON PRIORITA' E' COSTITUITA DA UN INSIEME DI ATOMI LINEARMENTE ORDINATI MA SENZA ALCUNA RELAZIONE STRUTTURALE SULL'INSIEME DELLE POSIZIONI LA SUA RAPPRESENTAZIONE E' QUASI SEMPRE ASSOCIATA AD UN UNICO MODELLO CONCETTUALE: QUELLO DI ALBERO BINARIO.

RAPPRESENTAZIONE CON STRUTTURE SEQUENZIALI

SI PUO' RAPPRESENTARE UNA CODA CON PRIORITA' DI n ELEMENTI UTILIZZANDO STRUTTURE SEQUENZIALI COME LISTE ORDINATE E LISTE NON ORDINATE.

NEL CASO DI LISTE ORDINATE, MIN E CANCELLAMIN HANNO COMPLESSITA' O(1), INSERISCI E' O(n).

NEL CASO DI LISTE NON ORDINATE, INSERISCI E' O(1), MIN E CANCELLAMIN HANNO COMPLESSITA' O(n).

PER RENDERE UNIFORME LA COMPLESSITA' DEGLI OPERATORI CANCELLAMIN E INSERISCI, REALIZZIAMO LA STRUTTURA PER MEZZO DI ALBERI BINARI

RAPPRESENTAZIONE CON ALBERI BINARI

GLI ELEMENTI DI UNA CODA CON PRIORITA' C POSSONO ESSERE MEMORIZZATI NEI NODI DI UN ALBERO BINARIO B CHE DEVE AVERE LE PROPRIETA':

(1) SE k E' IL LIVELLO MASSIMO DELLE FOGLIE, ALLORA B HA ESATTAMENTE 2^k-1 NODI DI LIVELLO MINORE DI k;

(2) TUTTE LE SUE FOGLIE DI LIVELLO MASSIMO SONO ADDOSSATE A SINISTRA;

(3) OGNI NODO CONTIENE UN ELEMENTO DI C CHE E' MAGGIORE DI QUELLO DEL PADRE (L'ALBERO B E' "*PARZIALMENTE ORDINATO*").

LE PROPRIETA' (1) E (2) DEFINISCONO UN *ALBERO QUASI PERFETTAMENTE BILANCIATO*.

7

ESEMPIO (CODA CON PRIORITA'): LA FIGURA MOSTRA UN ALBERO B CHE VERIFICA LE PROPRIETA' SUDDETTE E CONTIENE GLI ELEMENTI DELLA CODA CON PRIORITA'

C = { 5, 10, 8, 11, 13, 12, 9, 18, 3, 6 }.

IL LIVELLO MASSIMO DELLE FOGLIE DI B E' 3 E CI SONO $2^3-1=7$ NODI DI LIVELLO £ 2. LE TRE FOGLIE DI LIVELLO 3 SONO ADDOSSATE A SINISTRA.

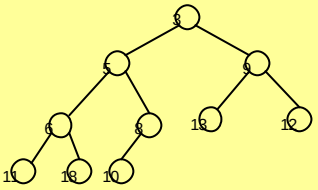


FIG. 1 ALBERO QUASI PERFETTAMENTE BILANCIATO PARZIALMENTE ORDINATO

8

PER REALIZZARE GLI OPERATORI SI OSSERVI CHE:

- **MIN** RESTITUISCE IL CONTENUTO DELLA RADICE DELL'ALBERO B;
- **INSERISCI** DEVE INSERIRE UNA NUOVA FOGLIA IN MODO DA MANTENERE VERIFICATE LE PROPRIETÀ (1) E (2) E QUINDI FAR SALIRE L'ELEMENTO INTRODOTTO FINO A VERIFICARE LA PROPRIETÀ (3).

9

ESEMPIO (INSERISCI): VOLENDO INSERIRE L'ELEMENTO 4 NELL'ALBERO, SI AGGIUNGE UN FIGLIO, CON 4, AL NODO 8, POI SI FA SALIRE PER VERIFICARE LA PROPRIETA' 3, CIOE' SI SCAMBIA 4 CON 8 E SUCCESSIVAMENTE 4 CON 5.

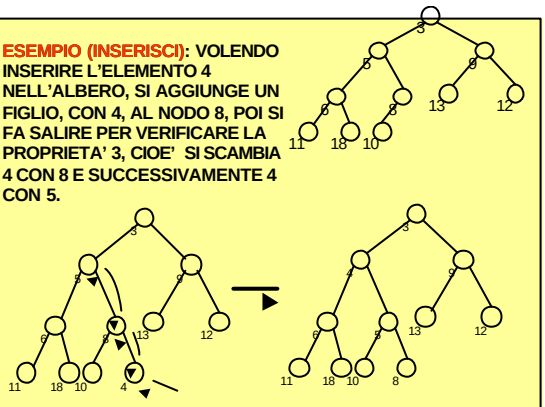


FIG. 2 INSERIMENTO DI 4

10

PER REALIZZARE CANCELLAMIN SI OSSERVI CHE:

- **CANCELLAMIN** PREVEDE LA CANCELLAZIONE DELLA FOGLIA DI LIVELLO MASSIMO PIÙ A DESTRA, IN MODO DA MANTENERE VERIFICATA LE PROPRIETÀ (1) E (2). SI REINSERISCE IL CONTENUTO DELLA FOGLIA CANCELLATA NELL'ALBERO PARTENDO DALLA RADICE E FACENDOLO "SCENDERE" IN MODO CHE L'ALBERO COSÌ MODIFICATO VERIFICHI ANCHE LA PROPRIETÀ (3).

11

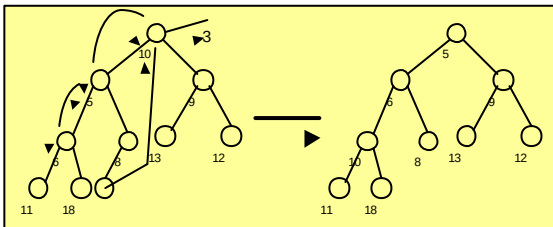


FIG. 3 CANCELLAZIONE DEL MINIMO

VOLENDO CANCELLARE IL MINIMO, CIOE' IL VALORE 3 DALL'ALBERO DI FIG. 1, SI CANCELLA LA FOGLIA PIU' A DESTRA, CONTENENTE 10, SI RICOPIA 10 NELLA RADICE, SI FA SCENDERE PER VERIFICARE LA PROPROETA' 3 CIOE SI SCAMBIA 10 CON 5 E SUCCESSIVAMENTE 10 CON 6.

12

OSSERVAZIONE:

INSERISCI E CANCELLAMIN PREVEDONO DUE FASI, LA PRIMA DI ***MODIFICA*** DELLA STRUTTURA DELL'ALBERO (CONDIZIONATA DALLA PROPRIETÀ (1) E (2)), LA SECONDA DI ***AGGIUSTAMENTO*** DEGLI ELEMENTI IN BASE ALLE PRIORITÀ (PROPRIETÀ (3)).

NELLA PRIMA FASE SI DEVE NECESSARIAMENTE FARE RIFERIMENTO ALLA FOGLIA ALL'ESTREMA DESTRA DELL'ULTIMO LIVELLO: QUESTO EVIDENZIA L'UTILITÀ DI MANTENERE TRACCIA DI TALE FOGLIA.

DI CONSEGUENZA UNA DEFINIZIONE PASCAL DEL TIPO PRIORICODA PUÒ ESSERE LA SEGUENTE:

```
PrioriCoda=RECORD
            albero:BinAlbero;
            ultimo:nodo
          END;
```

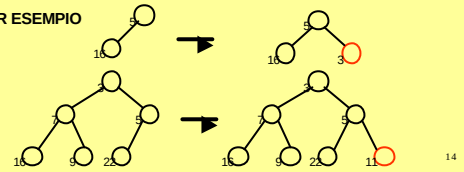
13

ANALIZZIAMO PIÙ IN DETTAGLIO **INSERISCI**.

LA PRIMA FASE, QUELLA DI MODIFICA DELLA STRUTTURA DELL'ALBERO, PREVEDE L'INSERIMENTO DI UNA FOGLIA "DOPO" L'ULTIMA. CASI PARTICOLARMENTE SEMPLICI SONO:

- **L'ALBERO È VUOTO**: BASTA AGGIUNGERE L'ELEMENTO COME RADICE;
- **L'ALBERO È COSTITUITO DALLA SOLA RADICE**: L'ELEMENTO VIENE AGGIUNTO COME FIGLIO SINISTRO DELLA RADICE;
- **L'ULTIMA FOGLIA È UN NODO FIGLIO SINISTRO**: L'ELEMENTO VIENE AGGIUNTO COME FRATELLO DESTRO;

PER ESEMPIO



14

RESTANO DA ESAMINARE DUE CASI CHE PREVEDONO ENTRAMBI UNA PRIMA FASE DI "***SALITA DA DESTRA***" ED UNA FASE SUCCESSIVA DI "***DISCESA VERSO SINISTRA***".

IL PROCESSO DI ***SALITA DA DESTRA*** (SI PASSA DA FIGLIO DESTRO A PADRE) PROSEGUE FINTANTOCHÉ IL NODO CONSIDERATO È UN FIGLIO DESTRO OPPURE NON È STATA RAGGIUNTA LA RADICE.

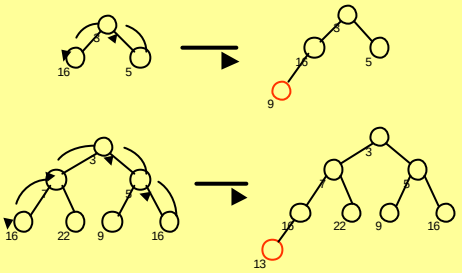
IL PROCESSO DI ***DISCESA VERSO SINISTRA*** (SI PASSA DAL PADRE AL FIGLIO SINISTRO) VIENE ITERATO FINO A QUANDO NON SI TROVA UNA FOGLIA; QUINDI SI PUÒ AGGIUNGERE LA FOGLIA.

QUELLO CHE DIFFERENZIA I DUE CASI È LA NECESSITÀ O MENO DI PASSARE A UN NUOVO LIVELLO.

15

SE SI DEVE PASSARE A UN NUOVO LIVELLO, SI GIUNGE ALLA RADICE DAL FIGLIO DESTRO: SI SCENDE QUINDI PARTENDO DALLA RADICE.

ESEMPI:



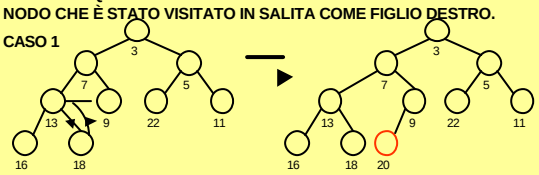
16

SE SI RIMANE ALLO STESSO LIVELLO:

O NON SI GIUNGE ALLA RADICE (CASO 1), OPPURE SI GIUNGE ALLA RADICE DAL FIGLIO SINISTRO (CASO 2);

SI SCENDE QUINDI PARTENDO DAL FRATELLO DESTRO DELL'ULTIMO NODO CHE È STATO VISITATO IN SALITA COME FIGLIO DESTRO.

CASO 1



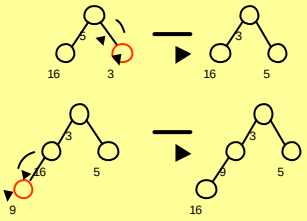
CASO 2



17

NELLA SECONDA FASE DI **INSERISCI**, CIOÈ L'AGGIUSTAMENTO DEGLI ELEMENTI IN BASE ALLE PRIORITÀ, SI PARTE DALLA FOGLIA INSERITA E SI CONFRONTA IL VALORE DI PRIORITÀ CON QUELLO DEL NODO PADRE: I VALORI VENGONO SCAMBIATI SE NON SODDISFANO LA PROPRIETÀ (3) E QUINDI SI RISALE L'ALBERO VERSO LA RADICE.

QUESTOPROCESSO DI CONFRONTO-SCAMBIO SI RIPETE FINO A QUANDO NON SI TROVANO DUE ELEMENTI CHE SODDISFANO GIÀ LA PROPRIETÀ (3) OPPURE SI ARRIVA ALLA RADICE.



18

L'**ALGORITMO** DI **INSERISCI**:

se l'albero è vuoto

allora inserisci l'elemento come radice che diventa anche ultimo
altrimenti

se l'albero ha solo la radice (che coincide con ultimo)
allora inserisci l'elemento come figlio sinistro della radice
altrimenti

se ultimo è un figlio sinistro

allora inserisci l'elemento come suo fratello destro
altrimenti

risali finché il nodo è un figlio destro e non è figlio della radice
se il nodo attuale è ancora un figlio destro

allora (si è raggiunta la radice) passa al fratello sinistro
altrimenti passa al fratello destro
scendi verso sinistra fino ad arrivare ad una foglia
inserisci l'elemento come figlio sinistro
ultimo diventa la foglia inserita

finché la priorità del nodo corrente è maggiore di quella del padre e non si raggiunta la radice
scambia il contenuto di padre e figlio

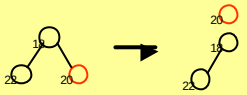
19

ANALIZZIAMO ORA CANCELLAMIN.

LA PRIMA FASE, QUELLA DI MODIFICA DELLA STRUTTURA DELL'ALBERO, PREVEDE L'ELIMINAZIONE DELLA FOGLIA DI LIVELLO MASSIMO PIÙ A DESTRA (IL CUI CONTENUTO DEVE PERÒ ESSERE RISISTEMATO NELLA SECONDA FASE). CASI PARTICOLARMENTE SEMPLICI SONO:

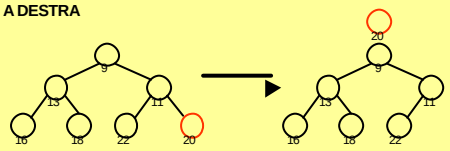
- **L'ALBERO È COSTITUITO DALLA SOLA RADICE**: SI CANCELLA L'INTERO ALBERO E NON È NECESSARIA LA SECONDA FASE;
- **L'ULTIMA FOGLIA E' UN NODO FIGLIO DESTRO**: DOPO LA CANCELLAZIONE L'ULTIMA FOGLIA DIVENTA IL NODO FRATELLO SINISTRO.

ESEMPIO



20

ALTRO ESEMPIO DI MODIFICA DELLA STRUTTURA DELL'ALBERO ELIMINANDO LA FOGLIA DI LIVELLO MASSIMO PIÙ A DESTRA



21

RESTANO DA ESAMINARE DUE CASI CHE PREVEDONO ENTRAMBI UNA PRIMA FASE DI "SALITA DA SINISTRA" ED UNA FASE SUCCESSIVA DI "DISCESA VERSO DESTRA" ALLA RICERCA DELLA NUOVA ULTIMA FOGLIA.

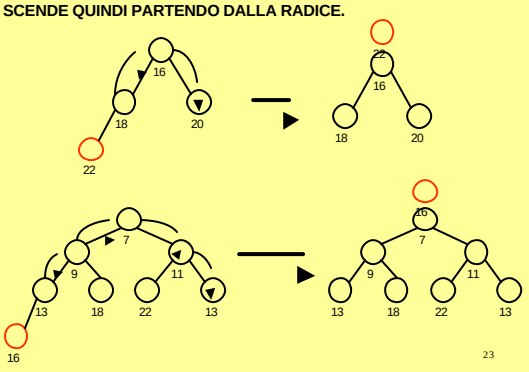
IL PROCESSO DI SALITA (SI PASSA DA FIGLIO A PADRE) PROSEGUE FINTANTOCHÉ IL NODO CONSIDERATO È UN FIGLIO SINISTRO OPPURE NON È STATA RAGGIUNTA LA RADICE

IL PROCESSO DI DISCESA (SI PASSA DAL PADRE AL FIGLIO DESTRO) VIENE ITERATO FINO A QUANDO NON SI TROVA UNA FOGLIA.

QUELLO CHE DIFFERENZIA I DUE CASI È SE IL LIVELLO MASSIMO VIENE O MENO SVUOTATO.

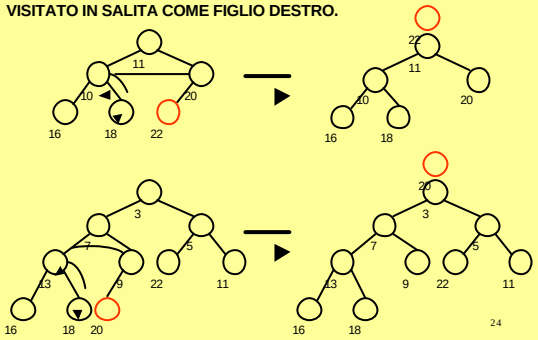
22

NEL PRIMO CASO SI GIUNGE ALLA RADICE DAL FIGLIO SINISTRO: SI SCENDE QUINDI PARTENDO DALLA RADICE.



23

NEL SECONDO CASO O NON SI GIUNGE ALLA RADICE OPPURE SI GIUNGE ALLA RADICE DAL FIGLIO DESTRO: SI SCENDE PARTENDO DAL FRATELLO SINISTRO DELL'ULTIMO NODO CHE È STATO VISITATO IN SALITA COME FIGLIO DESTRO.



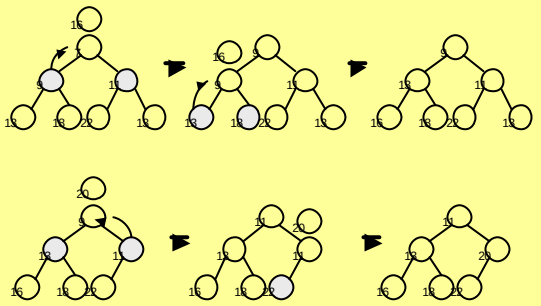
24

NELLA SECONDA FASE DI CANCELLAMIN, CIOÈ L'*AGGIUSTAMENTO* DEGLI ELEMENTI IN BASE ALLE PRIORITÀ, SI PARTE DALLA RADICE (CHE NON CONTIENE PIU' UN VALORE SIGNIFICATIVO) E SI RICERCA LA POSIZIONE DOVE INSERIRE IL CONTENUTO DELLA FOGLIA CANCELLATA IN MODO CHE SIA SODDISFATTA LA PROPRIETÀ (3): PER OGNI NODO ESAMINATO SI CONSIDERANO I FIGLI E, SE IL CONTENUTO DEL NODO CANCELLATO HA PRIORITÀ MINORE, SI SCRIVE NEL NODO ATTUALE IL CONTENUTO DEL FIGLIO CON PRIORITÀ MAGGIORE E SI CONSIDERA SE LA POSIZIONE DI TALE FIGLIO PUÒ ACCOGLIERE IL CONTENUTO DEL NODO CANCELLATO.

QUESTO PROCESSO SI RIPETE FINO A QUANDO SI TROVA UNA CONFIGURAZIONE CHE SODDISFA LA PROPRIETÀ (3), E CIOE' IL CONTENUTO DEL NODO CANCELLATO HA PRIORITÀ MAGGIORE DI ENTRAMBI I FIGLI DEL NODO ATTUALE, OPPURE SI ARRIVA AL LIVELLO FOGLIARE.

SI OSSERVI CHE, COME CONSEGUENZA DELLE PROPRIETÀ (1) E (2), IL CONFRONTO COINVOLGE ENTRAMBI I FIGLI, TRANNE NEL CASO IN CUI SI È RAGGIUNTO IL PENULTIMO LIVELLO, DOVE PUÒ ESSERCI UN NODO CON IL SOLO FIGLIO SINISTRO.

25



26

L'ALGORITMO DI CANCELLAMIN:

```
se l'albero non è vuoto
   allora se l'albero ha solo la radice
      allora cancella l'intero albero
      altrimenti
         memorizza il contenuto dell'ultima foglia    {per inserirlo
                                                       successivamente}
         inizio della fase di modifica della struttura
          se ultimo è un figlio destro
             allora in nuovo ultimo sarà il fratello sinistro
             altrimenti
               risali finché il nodo è figlio sinistro e non è figlio della radice
               se il nodo attuale è ancora un figlio sinistro
                  allora (si è raggiunta la radice)
                     se la radice non ha figlio destro
                        allora in nuovo ultimo è la radice stessa
                     altrimenti si deve scendere partendo dal figlio destro
                  altrimenti passa al fratello sinistro
               scendi verso destra fino ad arrivare ad una foglia
             cancella l'ultimo attuale e imposta il nuovo ultimo
          fine della fase di modifica della struttura
```

27

```
inizio della fase di aggiornamento delle priorità partendo dalla radice
finché non si è trovato dove sistemare il contenuto del nodo cancellato
ripeti
        se sono presenti entrambi i figli
          allora
            scegli il figlio con priorità maggiore
            se la priorità del figlio è maggiore di quella del nodo cancellato
              allora
                 scrivi il contenuto del figlio scelto nel nodo attuale
                 passa al figlio selezionato
              altrimenti si è trovato dove sistemare il contenuto del nodo
                         cancellato
          altrimenti {ha solo il figlio sinistro}
            si è trovato dove sistemare il contenuto del nodo cancellato
            se la priorità del figlio sinistro è maggiore di quella del nodo
                         cancellato
              allora
                 scrivi il contenuto del figlio sinistro nel nodo attuale
                 passa al figlio sinistro
        scrivi il contenuto del nodo cancellato nella posizione determinata
fine della fase di aggiornamento delle priorità
```

28

REALIZZAZIONE CON HEAP

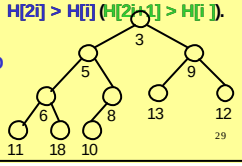
GLI ELEMENTI DI B POSSONO ESSERE DISPOSTI IN UN VETTORE H (LO *HEAP*) NELL'ORDINE IN CUI SI INCONTRANO VISITANDO L'ALBERO PER LIVELLI CRESCENTI ED ESAMINANDO DA SINISTRA VERSO DESTRA I NODI ALLO STESSO LIVELLO.

IN TAL CASO, SI HA CHE:

- H[1] E' L'ELEMENTO CONTENUTO NELLA RADICE DI B;
- H[2i] E H[2i+1] SONO GLI ELEMENTI CORRISPONDENTI AL FIGLIO SINISTRO E AL FIGLIO DESTRO DI H[i]

SE B CONTIENE n ELEMENTI, IL FIGLIO SINISTRO (DESTRO) DI H[i] NON ESISTE NELL'ALBERO SE E SOLO SE 2i > n (2i > n+1). INOLTRE, PER LA PROPRIETA' (3), SE IL FIGLIO SINISTRO (DESTRO) DI H[i] ESISTE, ALLORA H[2i] > H[i] (H[2i+1] > H[i]).

ESEMPIO: GLI ELEMENTI DELL'ALBERO SI MEMORIZZANO NELLO HEAP H COME SEGUE: H[1] = 3, H[2] = 5, H[3] = 9, H[4] = 6, H[5] = 8, H[6] = 13, H[7] = 12, H[8] = 11, H[9] = 18, H[10] = 10.



29

UN ESEMPIO DI REALIZZAZIONE IN PASCAL

```
Type
        prioricoda = record
                elementi: array[1..maxlung] of tipoelem;
                ultimo: 0..maxlung;
        end;
Procedure CREAPRIORICODA(var C:prioricoda);
Begin
        C.ultimo := 0;
End;


Function MIN(var A: insieme): tipoelem;
Begin
        if C.ultimo = 0
                then { errore: prioricoda vuota }
                else MIN := C.elementi[1]
end;
```

30

```
Procedure INSERISCI(x: tipoelem; var C: prioricoda);
var i,k: integer; temp: tipoelem;
Begin
        if C.ultimo = maxlung then
                { errore: prioricoda vuota }
        else begin
                C.ultimo := C.ultimo +1;
                C.elementi[C.ultimo] := x;   i := C.ultimo;
                if i>1 then k := i div 2;
                while ( i>1 ) and (C.elementi[i] < C.elementi[k] ) do begin
                        temp := C.elementi[i];
                        C.elementi[i] := C.elementi[k]; C.elementi[k] := temp;
                        i := k; if i>1 then k := i div 2;
                end;
        end;
end;
```

31

```
Procedure CANCELLAMIN(var C: prioricoda);
Var i,k: integer; temp: tipoelem; scambio: boolean;
Begin
        if C.ultimo = 0 then
                { errore: prioricoda vuota }
        else begin
                C.elementi[1] := C.elementi[C.ultimo];
                C.ultimo := C.ultimo –1; i := 1; scambio := true;
                while ( i<=C.ultimo div 2) and (scambio) do begin
                        k := 2*i;
                        if k < C.ultimo then
                          if C.elementi[k] > C.elementi[k+1] then k:=k+1;
                        if C.elementi[k] < C.elementi[i] then begin
                          temp := C.elementi[i];  C.elementi[i] := C.elementi[k];
                          C.elementi[k] := temp;   i := k
                        end else
                          scambio := false;
                end
        end
end;
```

32