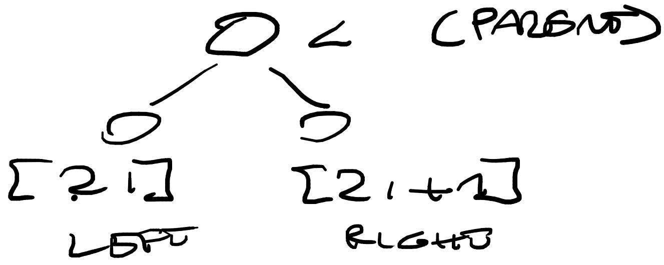
Esercizi

Esercizio 1 *(9 punti)* Realizzare una funzione diff(A1,A2,n) che dati due array di interi A1 e A2, organizzati a min-heap, con capacità n, restituisce un nuovo array A, ancora organizzato a min-heap, che contiene la differenza insiemistica dei valori contenuti in A1 e A2. Nel caso gli array contengano più occorrenze dello stesso valore v, nella differenza si inseriscono il numero di occorrenze di v in A1 meno il numero di occorrenze di v in A2 (ad es. se A1 contiene i valori 1,2,2,2,2,2 e A2 contiene i valori 1,2,1,2 allora la differenza A conterrà [redacted] 2,2). Valutarne la complessità.

MIN-HEAP → ARRAY ... ?



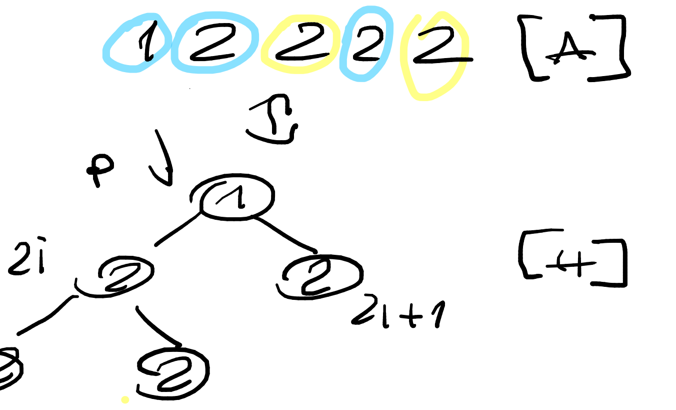
DIFF (A1, A2, N)

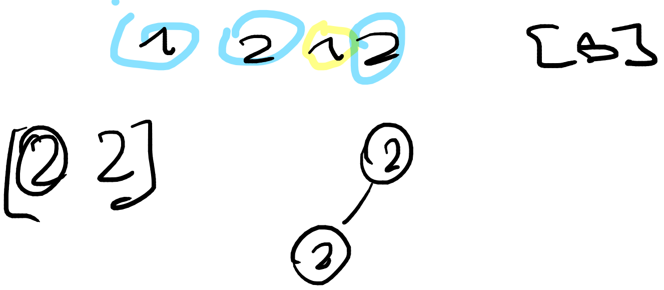
IF (A1.HEAPSIZE < A2.HEAPSIZE)
    RETURN NULL;

WHILE (A1.HEAPSIZE <> NULL)

IF (A1[i] == A2[i])
    i++;

ELSE {
    INT J=0.
    A[J] = A1[i]
    J++;
}





WHILE (i < A.HEAPSIZE/2)

MINHEAPIFY (A, i)

Esercizio 1 (10 punti) Un array di interi `A[1..n]` si dice *3-ordinato* se per ogni coppia di indici $i, j \in [1, n]$ con `i` $\leq$ `j` vale `A[i]%3` $\leq$ `A[j]%3`, dove `k%3` indica il resto della divisione intera di k per 3. Realizzare una funzione `3Order(A)` che dato un array `A[1..n]`, lo rende 3-ordinato. Valutarne la complessità in tempo e in spazio, e indicare se l'algoritmo è stabile. Anche qualora si usasse un algoritmo di ordinamento noto, lo pseudo-codice va comunque scritto esplicitamente.

3ORDER(A)

INT I=0; J=0. N = LENGTH (A)

WHILE (I < N || J < N)

IF (A[i] % 3 ≤ A[J] % 3)

I++;

J++;

```
3Order (A, p, r)
    if p < r
       q = (p+r)/2
       3Order (A, p, q)
       3Order (A, q+1, r)
       Merge (A, p, q, r)
```

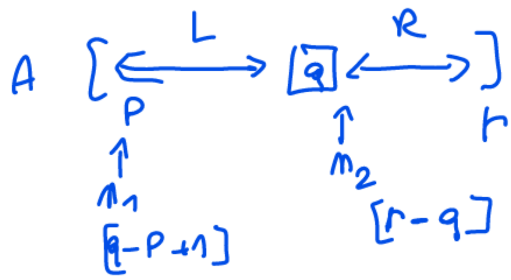
← UGUALE A MERGE

```
Merge (A, p, q, r)
    n1 = q-p+1
    n2 = r-q
    for i = 1 to n1
        L[i] = A[i]
    for j = 1 to n2
        R[j] = A[p+j]

    L[n1+1]=R[n2+1] = infinity

    i=p
    j=q+1
    for k = p to r
        if ((A[i]%3 <= B[j]%3)k)   // si assume infinity%3 = infinity
            A[k] = L[i]
            i++
        else
            A[k]=B[j]
            j++
}
```

I ≤ J (I++)



Esercizio 2 (9 punti) Sia $n > 0$ un intero. Si consideri la seguente ricorrenza $M(i, j)$ definita su tutte le coppie (i, j) con $1 \leq i \leq j \leq n$:

$$M(i,j) = \begin{cases} 1 & \text{se } i = j, \\ 2 & \text{se } j = i+1, \\ M(i+1, j-1) \cdot M(i+1, j) \cdot M(i, j-1) & \text{se } j > i+1. \end{cases}$$

RIEMPIMENTO

RIC.

VALORIZZAZIONE →

1. Scrivere una coppia di algoritmi INIT_M(n) e REC_M(i, j) per il calcolo memoizzato di $M(1, n)$.
2. Calcolare il numero esatto $T(n)$ di moltiplicazioni tra interi eseguite per il calcolo di $M(1, n)$.

Esercizio 2 (9 punti) Sia $n > 0$ un intero. Si consideri la seguente ricorrenza $M(i,j)$ definita su tutte le coppie (i,j) con $1 \le i \le j \le n$:

$$M(i,j) = \begin{cases} 1 & \text{se } i = j, \\ 2 & \text{se } j = i+1 \\ \underbrace{M(i+1,j-1)}_{\text{REC\_M}} \cdot \underbrace{M(i+1,j)}_{\text{REC\_M}} \cdot \underbrace{M(i,j-1)}_{\text{REC\_M}} & \text{se } j > i+1. \end{cases}$$

$\rightarrow$ INIT

M[i,j] =>

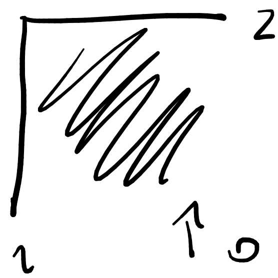
1. Scrivere una coppia di algoritmi INIT_M(n) e REC_M(i,j) per il calcolo memoizzato di $M(1,n)$.
2. Calcolare il numero esatto $T(n)$ di moltiplicazioni tra interi eseguite per il calcolo di $M(1,n)$.

```
INIT_M (M)
N = LENGTH(M)
 FOR I = 1 TO N
    M[I, I] = 1
    M[I, I+1] = 2

 FOR I = 1 TO N - 1
   FOR J = 2 TO N - 1
    M[I, J] = 0
```

$1 \le i \le j \le n$

$j > i + 1$




```
REC_M (I, J)
  IF (M[I, J] = 0)
M[I, J] = REC_M(I+1, J-1) · REC_M ......
```

$1 \le i \le j \le N$

$M(i+1,j-1) \cdot M(i+1,j) + M(i,j-1)$ se $j > i+1$.

$$\begin{matrix} 2 & 3 & 0 & 0 \\ 0 & 2 & 3 & 0 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 2 \end{matrix}$$

FILLING THESE ...

```
(a) INIT_M(n)
    if n=1 then return 2
    if n=2 then return 3
    for i=1 to n-1 do
        M[i,i] = 2
        M[i,i+1] = 3
    M[n,n] = 2
    for i=1 to n-2 do
        for j=i+2 to n do
            M[i,j] = 0
    return REC_M(1,n)
```

Se gli indici sono uno solo o due soli, allora, seguendo i casi base si ritorna 2 e 3.

Ricordandosi che:
- "i" va da i ad (n-1), quindi diventa perché abbiamo già inizializzato, (n-2) la scansione
- "j" va da (n-1) a 0 compresi, quindi dato che abbiamo inizializzato, parte da (n-2)

Una volta inizializzato secondo i casi base, tutto il resto della matrice viene riempita di zeri. Poi, si considera, essendo una scansione per diagonale, noi partiamo dal basso e riempiamo solo la parte sopra delle diagonali principali, ovviamente riempita anche questa di zeri.

Ultimo caso dell'indice [i, i]

```
REC_M(i,j)
if M[i,j] = 0 then
    M[i,j] = REC_M(i+1,j-1) * REC_M(i+1,j) + REC_M(i,j-1)
return M[i,j]
```

1

$n - 1 \; (*) \; n - 1$

N. ESATTO DI MOLTIPLIC.

$$\sum_{i=1}^{n-2} \sum_{j=i+2}^{n} 1 = \overset{\text{"}}{n} - \overset{\text{"}}{j} \quad (\text{LOGICA} \ldots)$$

GAUSS

$$\left[\sum_{k=1}^{n} k = \frac{n(n-1)}{2} \right]$$

$$\sum_{i=1}^{n-2} n - i - 1 =$$

$$\sum_{k=1}^{(n-2)} k = \frac{(n-2)(n-1)}{2}$$

Soluzione:


```
(a) INIT_M(n)
    if n=1 then return 2
    if n=2 then return 3
    for i=1 to n-1 do
        M[i,i] = 2
        M[i,i+1] = 3
    M[n,n] = 2
    for i=1 to n-2 do
        for j=i+2 to n do
            M[i,j] = 0
    return REC_M(1,n)
```

Se gli indici sono uno solo o due soli, allora, seguendo i casi base si ritorna 2 e 3.

Ricordandosi che:
- "i" va da i ad (n-1), quindi diventa perché abbiamo già inizializzato, (n-2) la scansione
- "j" va da (n-1) a 0 compresi, quindi dato che abbiamo inizializzato, parte da (n-2)

Una volta inizializzato secondo i casi base, tutto il resto della matrice viene riempita di zeri. Poi, si considera, essendo una scansione per diagonale, noi partiamo dal basso e riempiamo solo la parte sopra delle diagonali principali, ovviamente riempita anche questa di zeri.

Ultimo caso dell'indice [i, i]

```
REC_M(i,j)
if M[i,j] = 0 then
    M[i,j] = REC_M(i+1,j-1) * REC_M(i+1,j) + REC_M(i,j-1)
return M[i,j]
```


1

$$\sum_{i=1}^{n-2} \sum_{j=i+2}^{n} 1$$

$$\sum_{i=1}^{n-2} \underbrace{n-i-1}_{k}$$

$$\sum_{k=1}^{(n-2)} k \quad \frac{(n-2)(n-1)}{2}$$

(b)

$$T(n) = \sum_{i=1}^{n-2} \sum_{j=i+2}^{n} 1 = \sum_{i=1}^{n-2} n - i - 1 = \sum_{k=1}^{n-2} k = (n-2)(n-1)/2$$

La sommatoria interna è costituita da soli 1, in quanto richiede una moltiplicazione tra tre numeri interi, nello specifico (n-2), (n-1), n.