Esercizi in vista dell'esonero del 12 aprile 2019 - Introduzione agli algoritmi

1. Dato un Maxheap si scriva una funzione che realizza la cancellazione di un elemento.

2. Dati due Maxheap, si scriva una funzione che realizza un unico Maxheap sugli elementi di entrambi.

3. Si consideri il problema di determinare gli m elementi più grandi in un array di n elementi e si confrontino dal punto di vista del tempo di esecuzione asintotico i due algoritmi qui sotto proposti.

    1. Si ordinano gli elementi
    2. Si costruisce un maxheap dall'array e poi si estrae per m volte il massimo.

4. Disegnare l'albero di decisione del seguente algoritmo di ordinamento, per n=3:

    1. algoritmo unAltroSort(A)
    2. n = A.length
    3. BubbleSort(A,n-1)
    4. Merge(A,1,n-1,n)

    dove la subroutine BubbleSort e' richiamata sui primi n-1 elementi, e la subroutine Merge fonde il sotto-array A[1,n-1] con il sotto-array A[n,n] contenente un solo elemento.
    Qual e' il numero di confronti eseguiti dall'algoritmo nel caso peggiore per n=3?
    Si esprima il tempo di esecuzione asintotico nel caso peggiore, per valori di n arbitrari, usando la notazione asintotica più opportuna.

5. Sia S un insieme di n interi. Vogliamo un algoritmo che determini se ci sono almeno due interi uguali in S. Non è difficile progettare un semplice algoritmo quadratico che risolve il problema. Si modifichi il mergesort per ottenere una soluzione di complessità O(nlgn).

Esercizi analisi di algoritmi

Es. 1

```
Analizza(n)
input: un intero n
count = 0;
for i=1 to n do
        j = 1;
        while ( j+n/2 ≤ n ) do
                k=1;
                while (k ≤ n) do
                        count ++
                        k = 2*k;
                j++;
return count;
```

Sol $\Theta(n^2 \lg n)$

Es.2

```
Analizza(n)
input: un intero n
i = 1; s = 1;
while ( s ≤ n )
        i++
        s=i+s
        print("*")
return;
```

Sol. $\Theta(\sqrt{n})$

Es. 3

```
Analizza(n)
input: un intero n
count = 0;
for i=1 to n do
        j = i;
        while ( j ≤ i^2 ) do
                if j è multiplo di i then
                        for k=1 to j-1 do
                                print("*")
                j++;
return;
```

Sol $O(n^4)$