ESAME 31 MARZO 2022

Esercizio 1 (10 punti):
Si consideri la seguente funzione:

```
funzione Exam(n):
    tot ← 1;                                   Θ(1)
    if n <= 1: return tot;                     Θ(1)  CASO BASE
    j ← 63;                                    Θ(1)
    while j > 0 do:                            9 VOLTE
        k ← 0;
        while 3 * k <= n do: k ← k + 1;        n/3 VOLTE
        tot ← tot + 2 * Exam(k);               T(n/3)
        j ← j − 7;
    while k > 0 do:                            n/3 VOLTE
        for i = 1 to n do: tot ← tot − 1       n VOLTE
        k ← k − 1;
    return tot
```

1	3
2	6
3	9
4	12
5	15

$9\left[\frac{n}{3}\right]\Theta(1)$

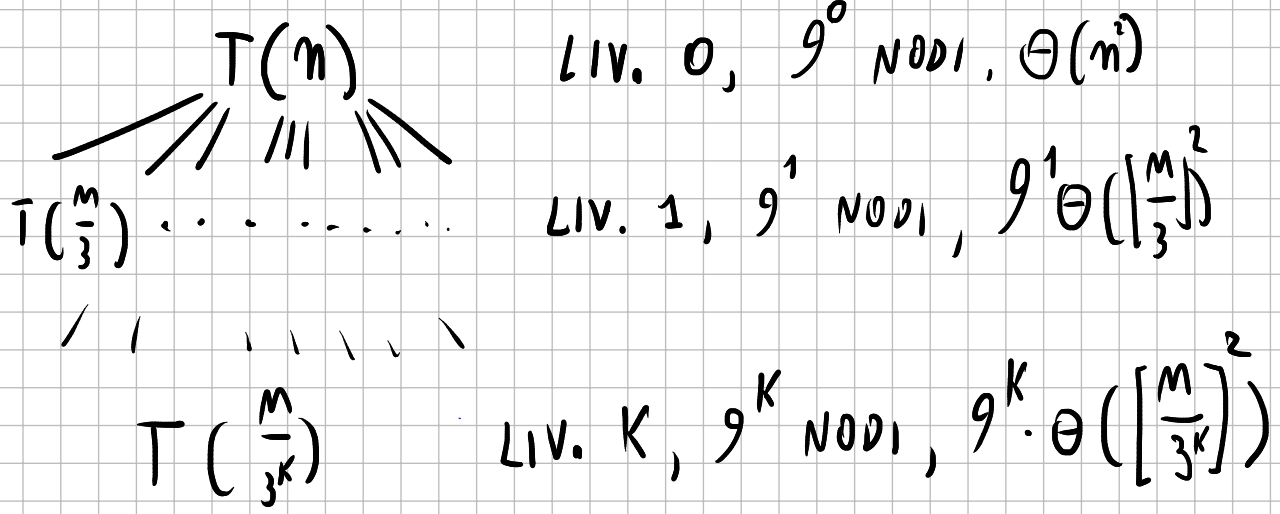
$$T(n) = 9T\left(\frac{n}{3}\right) + \Theta(n^2)$$

$$T(1) = \Theta(1)$$

a) Si imposti la relazione di ricorrenza che ne definisce il tempo di esecuzione giustificando dettagliatamente l'equazione ottenuta.

b) Si risolva la ricorrenza usando il **metodo dell'albero** dettagliando i passaggi del calcolo e giustificando ogni affermazione.

OGNI LIVELLO DI ALTEZZA h HA 9^h NODI, OGNI LIVELLO h HA COSTO $9^h\Theta\left(\left[\frac{n}{3^h}\right]^2\right)$.



Esercizio 2 (10 punti): Dato un array **ordinato** A di n interi ed un intero k vogliamo sapere quante coppie in A hanno somma k. Si progetti un algoritmo **iterativo** che risolva il problema in tempo $\Theta(n)$.

Ad esempio:

- se $A = [1, 2, 2, 3, 4, 5, 5, 5, 8, 9, 9]$ e $k = 7$ l'algoritmo deve restituire 7 (le coppie a somma 7 sono infatti $(1, 5)$, $(1, 6)$, $(1, 7)$, $(2, 5)$, $(2, 6)$, $(2, 7)$ e $(3, 4)$).
- se $A = [1, 5, 5, 5, 9]$ e $k = 10$ l'algoritmo deve restituire 4 (le coppie a somma 10 sono infatti $(0, 4), (1, 2), (1, 3), (2, 3)$).

Dell'algoritmo proposto:

a) si dia la descrizione a parole,

b) si scriva lo pseudocodice,

c) si giustifichi il costo computazionale.

applicherò un algoritmo simile al partiziona del quick sort, avendo un indice i che va da sinistra a destra ed un indice j che va da destra a sinistra, se la somma sarà maggiore di k decremento j, se maggiore incremento i, quando è uguale, aumento il numero delle coppie, e sposto il puntatore del numero che ha, come suo prossimo indice, valore uguale a se stesso.

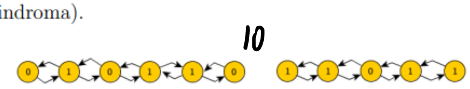
```
DEF ES2(A,K):
    i=0;                          Θ(1)
    J=LEN(A)-1;                   Θ(1)
    COPPIE=0;                     Θ(1)
    WHILE(J-i>1):                 AL PIU' N VOLTE
        SOMMA=A[i]+A[J];          Θ(1)
        IF(SOMMA==K):             Θ(1)
            COPPIE+=1;,           Θ(1)
            IF(A[J-1]==A[J]):     Θ(1)
                J-=1;             Θ(1)
            ELSE: i+=1            Θ(1)
        ELSE IF(SOMMA>K):         Θ(1)
            J-=1;                 Θ(1)
        ELSE:                     Θ(1)
            i+=1;                 Θ(1)
    RETURN COPPIE;                Θ(1)
```

$$COSTO = 2\Theta(1) + n[10\Theta(1)] = \Theta(n)$$

Esercizio 3 (10 punti):

Si consideri una lista a puntatori L, in cui ogni elemento è un record a tre campi: il campo `val` contenente un bit (cioé un valore 0 o 1), il campo `next` con il puntatore al nodo seguente (`next` vale *None* per l'ultimo record della lista) ed il campo `prec` con il puntatore al nodo precedente (`prec` vale *None* per il primo record della lista).

Bisogna verificare se la stringa che si ottiene considerando i bit dei vari nodi della lista è palindroma. Ad esempio, se la lista L in input è quella di sinistra nella figura che segue, la risposta è NO (la stringa binaria 010110 non è palindroma, mentre se L è la lista di destra la risposta è SI (la stringa binaria 11011 è palindroma).



Progettare un algoritmo (iterativo o ricorsivo) che, dato il puntatore s alla testa della lista, risolve il problema in tempo $\Theta(n)$, dove n è il numero di nodi della lista a puntatori. Lo spazio di lavoro dell'algoritmo proposto deve essere $O(1)$ (in altri termini NON è possibile definire e utilizzare altre liste). Dell'algoritmo proposto:

a) si dia la descrizione a parole,

b) si scriva lo pseudocodice,

c) si giustifichi il costo computazionale.

```
DEF ES3(S):
    Q=NONE; Q=S;          Θ(1)
    CNT=0;                Θ(1)
    WHILE(S):             M VOLTE
        Q=Q->NEXT;        Θ(1)
        CNT+=1;           Θ(1)
    MID=CNT//2;           Θ(1)
    CNT=0;                Θ(1)
    WHILE(CNT<MID):       M/2 VOLTE
        IF(Q->VAL!=P->VAL): Θ(1)
            RETURN FALSE; Θ(1)
        CNT+=1;           Θ(1)
        P=P->NEXT;        Θ(1)
        Q=Q->PREC;        Θ(1)
    RETURN TRUE;          Θ(1)
```

COSTO = $\Theta(M)$

OTTENGO UN PUNTATORE ALLA CODA, CONTROLLO OGNI VOLTA I DUE PUNTATORI OPPOSTI, SE DIVERSI RITORNA FALSE, SE UGUALI PRENDO IL SUCCESSORE DI UNO ED IL PREC DELL'ALTRO, SE NON TROVERO' DIVERGENZE, RITORNERO' TRUE.