# ESAME 13 GENNAIO 2022

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

```
funzione Exam(n):
    tot ← n;  Θ(1)
    if n <= 1:  return tot;  Θ(1) CASO BASE
    j ← 80;  Θ(1)
    while j >= 3 do:  30 VOLTE
        tot ← tot + j;   } Θ(1)
        j ← j − 2;       }
    return tot+Exam(n − j)  T(n-2)
```

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.

$$T(n) = T(n-2) + \Theta(1) \qquad T(1) = \Theta(1)$$

$T(n)$ $\Theta(1)$
|
$T(n-2)$ $\Theta(1)$
|
$T(n-4)$ $\Theta(1)$
|
$T(n-2K)$ FINO A $K = \frac{n}{2}$

$$T(n) = \frac{n}{2} \cdot \Theta(1) = \Theta(n)$$

**Esercizio 2 (10 punti):**
Abbiamo due array **ordinati** $A$ e $B$ di $n$ interi distinti; si vuole sapere se esiste un valore $x$ in $A$ ed un valore $y$ in $B$ che differiscono al più 3 in valore assoluto (vale a dire $|x - y| \leq 3$).

Ad esempio:
per $A = [1, 2, 20, 30]$ e $B = [6, 7, 10]$ la risposta è negativa.
Per $A = [1, 2, 9, 10, 12]$ e $B = [6, 14, 16, 20]$ la risposta è positiva (grazie alla coppia $(9, 6)$ o anche $(12, 14)$).

Progettare un **algoritmo** che risolva il problema restituendo 1 se la risposta è positiva, 0 altrimenti. Il costo computazionale dell'algoritmo deve essere asintoticamente strettamente inferiore a $\Theta(n^2)$.

Dell'algoritmo proposto

a) si dia la descrizione a parole,

b) si scriva lo pseudocodice,

c) si giustifichi il costo computazionale.

USERO' DUE INDICI, UNO SCORRERA' A ED UNO B, SE LA DISTANZA SARA' >3, CAMMINERO' CON L'INDICE SUL VALORE MINORE, UNA VOLTA GIUNTO AL TERMINE, SE NON TROVO VALORI, RITORNERO' 0.

```
DEF ES2(A,B):
    i=j=0;  Θ(1)
    WHILE ( !( i=LEN(A)-1 AND j=LEN(B)-1)):  AL PIU' LEN(A)+LEN(B) VOLTE
        IF (ABS(B[j]-A[i])<=3):
            RETURN 1;
        IF ( i=LEN(A)-1):
            j+=1;
        ELSE IF (j=LEN(A)-1):
            i+=1;
        ELSE:
            IF A[i] < A[j]:
                i+=1;
            ELSE:
                j+=1;
    IF (ABS(B[j]-A[i])<=3):  Θ(1)
        RETURN 1;            Θ(1)
    RETURN 0;  Θ(1)
```

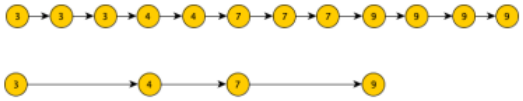
Θ(1)

ESSENDO IL WHILE AL PIU' n VOLTE, ED AVENDO DENTRO SOLO OPERAZIONI Θ(1), IL COSTO SARA' Θ(n).

**Esercizio 3 (10 punti):**

Si consideri una lista non vuota $L$, in cui ogni elemento è un record a due campi, il campo `val` contenente un intero ed il campo `next` con il puntatore al nodo seguente (`next` vale *None* per l'ultimo record della lista).

Gli interi nella lista sono ordinati in modo non decrescente e bisogna eliminare dalla lista i record contenenti duplicati. Si consideri ad esempio la lista $L$ in figura; subito sotto viene riportato il risultato dell'operazione di cancellazione.



Progettare un **algoritmo iterativo** che, dato il puntatore $r$ alla testa della lista effettui l'operazione di modifica in tempo $\Theta(n)$ dove $n$ è il numero di elementi presenti nella lista. Lo spazio di lavoro dell'algoritmo deve essere $O(1)$.

Dell'algoritmo proposto

a) si dia la descrizione a parole,

b) si scriva lo pseudocodice,

c) si giustifichi il costo computazionale.

d) si scriva lo pseudocodice di un algoritmo **ricorsivo** che risolve il problema

SCORRERO' LA LISTA, OGNI QUAL VOLTA TROVO UN NUOVO VALORE, LO SALVO PASSANDOLO ALL'ITERAZIONE SUCCESSIVA, FINCHÉ NON TROVO UN NUOVO VALORE, ELIMINO TUTTI I NODI.

ITERATIVO:

```
DEF ES3(R):
    WHILE(R):                          n VOLTE
        IF(R->NEXT->KEY == R->KEY):
            R->NEXT = R->NEXT->NEXT;
        ELSE:                                   } Θ(1)
            R=R->NEXT;
```

VISITO TUTTI I NODI IN $\Theta(1)$, QUINDI COSTA $\Theta(n)$!

RICORSIVO:

```
DEF ES3(R):
    IF (R->NEXT->KEY==R->KEY):          Θ(1)
        R->NEXT = R->NEXT->NEXT;        b = n-1
        ES3(R);                         T(b)
    ELSE:                               Θ(1)
        ES3(R->NEXT);                   T(n-1)
```

$$T(n) = T(n-1) + \Theta(1)$$