# ALGORITMI E STRUTTURE DATI I
# TEMA D'ESAME DEL 22/02/2022

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si risolva la seguente equazione di ricorrenza, calcolandone l'**andamento asintotico**:

$$\mathsf{T}(n) = \begin{cases} 1, & \text{se } n \leq 5; \\ 2 \cdot \mathsf{T}(\frac{n}{5}) + 3 \cdot \mathsf{T}(\sqrt[5]{n}) + 4, & \text{altrimenti.} \end{cases}$$

2. Si scriva un **algoritmo iterativo** che **simuli precisamente** l'algoritmo ricorsivo di seguito riportato, dove S è una funzione esterna non meglio specificata.

```
function Algoritmo(A, l, r)
    s ← r − l + 1
    if s ≤ 3 then
        return S(A, l, r)
    else
        k ← ⌊s/3⌋
        z ← Algoritmo(A, l, r − k)
        z ← z + Algoritmo(A, l + k, r)
        h ← ⌊s/2⌋
        z ← z − Algoritmo(A, l + h, r − h)
        return z
```

# ALGORITMI E STRUTTURE DATI I
# TEMA D'ESAME DEL 25/03/2022

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si risolva la seguente equazione di ricorrenza, calcolandone l'**andamento asintotico**:
$$\mathsf{T}(n) = \begin{cases} 1, & \text{se } n \leq 2; \\ 3 \cdot \mathsf{T}(\sqrt{n}) + 2 \cdot \mathsf{T}(\sqrt[4]{n}) + \log n, & \text{altrimenti.} \end{cases}$$

2. Si scriva un algoritmo che, dati un grafo $\mathcal{G} = \langle \mathrm{V}, E \rangle$ e due vertici $s, u \in \mathrm{V}$, verifichi in **tempo lineare** sulla dimensione del grafo se **tutti i percorsi infiniti** che **partono** da $s$ **non passano infinite volte** per $u$.

# ALGORITMI E STRUTTURE DATI I
# TEMA D'ESAME DEL 21/06/2022

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si individuino, nel caso esistano, le **costanti moltiplicative** atte a mostrare la seguente relazione asintotica: $\ln\left(\frac{n}{e}\right) = \Theta(\ln(n^e))$ (si ricorda che con 'ln' si indica il **logaritmo naturale** e con '$e$' la **costante di Eulero**, detta anche di **Nepero**). In caso contrario, mostrare la falsità della relazione.

2. Si scriva un **algoritmo ricorsivo** che, dati in ingresso un albero binario di ricerca su interi $\mathcal{T}$ e due valori $k_1, k_2 \in \mathbb{N}$, inserisca in una lista $\mathcal{L}$ le chiavi $k$ contenute in $\mathcal{T}$ comprese tra $k_1$ e $k_2$ ($k_1 \leq k \leq k_2$), in modo che al termine $\mathcal{L}$ contenga valori ordinati in modo decrescente. Tale algoritmo dovrà avere **complessità lineare** nella dimensione dell'albero. Infine, si scriva un **algoritmo iterativo** che **simuli precisamente** l'algoritmo ricorsivo di cui sopra.

**ALGORITMI E STRUTTURE DATI I**
**TEMA D'ESAME DEL 21/07/2022**

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si individuino, nel caso esistano, le **costanti moltiplicative** atte a mostrare la seguente relazione asintotica: $\log_2(n^{2n}) + n - \log_2(n) = \Theta(\log_2(n^n))$. In caso contrario, mostrare la falsità della relazione.

2. Si scriva un **algoritmo ricorsivo** che, dati in ingresso un albero binario di ricerca su interi $\mathcal{T}$ e due valori $k_1, k_2 \in \mathbb{N}$, cancelli da $\mathcal{T}$ le chiavi $k$ comprese tra $k_1$ e $k_2$ ($k_1 \leq k \leq k_2$). Tale algoritmo dovrà essere efficiente e non far uso **nè di variabili globali nè di parametri passati per riferimento**. Infine, si scriva un **algoritmo iterativo** che **simuli precisamente** l'algoritmo ricorsivo di cui sopra.

**ALGORITMI E STRUTTURE DATI I**
**TEMA D'ESAME DEL 22/09/2022**

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si individuino, nel caso esistano, le **costanti moltiplicative** atte a mostrare la seguente relazione asintotica: $n \cdot \ln\left(e^2 \cdot n\right) = \Theta\left(\ln \sqrt{(n)^n}\right)$ (si ricorda che con 'ln' si indica il logaritmo naturale e con '$e$' la costante di Eulero, detta anche di Nepero). In caso contrario, mostrare la falsità della relazione. Il procedimento seguito va riportato per esteso.

2. Si scriva un algoritmo che, dato in ingresso un grafo orientato $\mathcal{G} = \langle V, E \rangle$, verifichi **in tempo lineare sulla dimensione del grafo** se la seguente condizione è soddisfatta:
 + $\mathcal{G}$ contiene tre vertici distinti, di seguito denominati $a$, $b$ e $c$, tali che:
   - $a$ e $b$ sono entrambi raggiungibili da $c$;
   - $a$ e $c$ sono entrambi raggiungibili da $b$;
   - $b$ e $c$ sono entrambi raggiungibili da $a$.

 Notare che i vertici $a$, $b$ e $c$ **non sono input** del problema.

**ALGORITMI E STRUTTURE DATI I**
**TEMA D'ESAME DEL 19/10/2022**

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si risolva la seguente equazione di ricorrenza, calcolandone l'**andamento asintotico**:

$$\mathsf{T}(n) = \begin{cases} 1, & \text{se } n \leq 2; \\ \sqrt[3]{n^2} \cdot \mathsf{T}(\sqrt[3]{n}) + n, & \text{altrimenti.} \end{cases}$$

2. Si scriva un **algoritmo ricorsivo** che, dati in ingresso un albero binario di ricerca $\mathcal{T}$ e un intero positivo $k$, cancelli da $\mathcal{T}$ tutti i nodi che si trovano in posizioni multiple di $k$ nell'ordinamento totale delle chiavi dell'albero. Tale algoritmo dovrà essere efficiente e non far uso **né di variabili globali né di parametri passati per riferimento**. Infine, si scriva un **algoritmo iterativo** che **simuli precisamente** l'algoritmo ricorsivo di cui sopra.

# ALGORITMI E STRUTTURE DATI I
# TEMA D'ESAME DEL 25/01/2023

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si risolva la seguente equazione di ricorrenza, calcolandone l'**andamento asintotico**:

$$\mathsf{T}(n) = \begin{cases} 1, & \text{se } n \leq 2; \\ 2 \cdot \mathsf{T}(\sqrt[4]{n}) + \log(2n), & \text{altrimenti.} \end{cases}$$

2. Si scriva un **algoritmo iterativo** che **simuli precisamente** l'algoritmo ricorsivo di seguito riportato, dove Z è una funzione esterna non meglio specificata.

```
function Algoritmo(T, h)
    if T = Nil then
        return Z(0, h)
    else
        a ← 0
        if T→key ≡ 0 mod 2 then
            a ← a + Algoritmo(T→dx, 2 · h)
        if T→key ≡ 1 mod 3 then
            a ← a − Algoritmo(T→sx, 3 · h)
        return Z(T→key, a)
```

**ALGORITMI E STRUTTURE DATI I**
**TEMA D'ESAME DEL 22/02/2023**

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si risolva la seguente equazione di ricorrenza, calcolandone l'**andamento asintotico**:

$$\mathsf{T}(n) = \begin{cases} 1, & \text{se } n \leq 27; \\ 3n^2 \cdot \mathsf{T}(\sqrt[3]{n}) + 2n^3, & \text{altrimenti.} \end{cases}$$

2. Si scriva un **algoritmo iterativo** che **simuli precisamente** l'algoritmo ricorsivo di seguito riportato, dove $\mathsf{Z}_l$ e $\mathsf{Z}_r$ sono due funzioni esterne non meglio specificate che soddisfano la seguente proprietà: $p < \mathsf{Z}_l(A, p, s) < \mathsf{Z}_r(A, p, s) \leq s$, quando $p + 1 < s$.

```
function Algoritmo(A, p, s)
  if s ≤ p + 1 then
    return 0
  else
    q ← Z_l(A, p, s)
    r ← Z_r(A, p, s)
    a ← Algoritmo(A, p, q)
    a ← a − Algoritmo(A, q, r)
    a ← a + Algoritmo(A, r, s)
    return a + (r − q)
```

**ALGORITMI E STRUTTURE DATI I**
**TEMA D'ESAME DEL 17/03/2023**

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si risolva la seguente equazione di ricorrenza, calcolandone l'**andamento asintotico**:

$$\mathsf{T}(n) = \begin{cases} 1, & \text{se } n \leq 4; \\ 8 \cdot \mathsf{T}(\frac{n}{4}) + \sqrt{n}, & \text{altrimenti.} \end{cases}$$

2. Si scriva un **algoritmo ricorsivo** che, presi in ingresso un albero binario $\mathcal{T}$ contenente dati interi e un intero positivo $k$, restituisca il valore della massima profondità dei nodi con valore di chiave multiplo di $k$. Nel caso l'insieme di tali nodi fosse vuoto, è richiesta la restituzione del valore di default $-1$. L'algoritmo dovrà essere efficiente e non far uso **né di variabili globali né di parametri passati per riferimento**. Infine, si scriva un **algoritmo iterativo** che **simuli precisamente** l'algoritmo ricorsivo di cui sopra.

**ALGORITMI E STRUTTURE DATI I**
**TEMA D'ESAME DEL 20/06/2023**

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 1h 30m**

1. Si dimostri, **esplicitando il procedimento seguito nella sua interezza**, la verità o la falsità della seguente affermazione:
$$\text{se } \mathsf{f}(n) = \Theta\left(\sqrt{\mathsf{g}(n)}\right) \text{e } \mathsf{g}(n) = \Theta\big((\mathsf{k}(n))^4\big), \text{ allora } \mathsf{f}(n) = \Theta\big((\mathsf{k}(n))^2\big).$$

2. Si scriva un algoritmo che, dati in ingresso un grafo $\mathcal{G} = \langle \mathrm{V}, E \rangle$, rappresentato per mezzo di una **lista di adiacenza**, un array $\boldsymbol{A}$ di dimensione $k \leq |\mathrm{V}|$, tale che $\boldsymbol{A}[i] \in \mathrm{V}$ per ogni $i \in \{0, \ldots, k\}$, e un vertice $v \in \mathrm{V}$, calcoli in **tempo lineare** l'insieme dei vertici $\mathrm{Z} \subseteq \mathrm{V}$ contenente tutti e soli i vertici di $\mathcal{G}$ che soddisfano la seguente condizione:

$z \in \mathrm{Z}$
se e solo se
esiste in $\mathcal{G}$ un percorso da $z$ a $v$ che, **prima di raggiungere** $v$,
passa per almeno un vertice contenuto in $\boldsymbol{A}$.

Successivamente, si analizzi il **tempo di esecuzione** dell'algoritmo proposto.

**ALGORITMI E STRUTTURE DATI I**
**TEMA D'ESAME DEL 20/07/2023**

M. BENERECETTI & F. MOGAVERO

**Tempo a disposizione: 2h**

1. Si risolva la seguente equazione di ricorrenza, calcolandone l'**andamento asintotico**:

$$\mathsf{T}(n) = \begin{cases} 1, & \text{se } n \leq 1; \\ 2 \cdot \mathsf{T}(\frac{n}{5}) + 3 \cdot \mathsf{T}(\frac{n}{25}) + 4, & \text{altrimenti.} \end{cases}$$

2. Si scriva un algoritmo che, dati in ingresso un grafo $\mathcal{G} = \langle \mathrm{V}, E \rangle$, rappresentato per mezzo di una **lista di adiacenza**, un array $\boldsymbol{A}$ di dimensione $k \leq |\mathrm{V}|$, tale che $\boldsymbol{A}[i] \in \mathrm{V}$ per ogni $i \in \{0, \ldots, k\}$, e due vertici $v, w \in \mathrm{V}$, calcoli in **tempo lineare** l'insieme dei vertici $\mathrm{Z} \subseteq \mathrm{V}$ contenente tutti e soli quei vertici di $\boldsymbol{A}$ che soddisfano la seguente condizione:

$$z \in \mathrm{Z}$$
se e solo se
tutti i percorsi semplici in $\mathcal{G}$ che partono da $v$ e terminano in $w$ non passano per $z$.

Successivamente, si analizzi il **tempo di esecuzione** dell'algoritmo proposto.