**Cognome**: **Nome**: **# Matricola**: **Riga**: **Col**:

# *Algoritmi e Strutture Dati – 9/1/2024 – Parte A*

**Esercizio -1** Iscriversi allo scritto entro la scadenza. In caso di inadempienza, -1 al voto finale.

**Esercizio 0** Scrivere correttamente nome, cognome, numero di matricola, riga e colonna su tutti i fogli consegnati. Consegnare foglio A4 e foglio protocollo di bella. In caso di inadempienza, -1 al voto finale.

## A1 – Complessità – Punti $\geq 8$

Si consideri questo algoritmo:

```
int fun(int[][] A, int n)
    return frec(A, 1, 1, n, n)
```

```
int frec(int[][] A, int s_i, int s_j, int e_i, int e_j)
    if s_i == e_i or s_j == e_j then
        return 0
    else
        int sumSoFar = 0
        int i = e_i
        int j = e_j
        while i > s_i and j > s_j do
            sumSoFar += A[i][j]
            if A[i][j] < A[i − 1][j] then
                i = i − 1
            else
                j = j − 1
        int m_i = ⌊(s_i + e_i)/2⌋
        int m_j = ⌊(s_j + e_j)/2⌋
        sumSoFar += frec(A, s_i, s_j, m_i, m_j)
        sumSoFar += frec(A, m_i + 1, s_j, e_i, m_j)
        sumSoFar += frec(A, s_i, m_j + 1, m_i, e_j)
        sumSoFar += frec(A, m_i + 1, m_j + 1, e_i, e_j)
        return sumSoFar
```

Scrivere l'equazione di ricorrenza associata a questo algoritmo e calcolare la complessità computazionale che ne deriva, considerando per semplicità una matrice in input di dimensione $n = 2^k$, con $k$ intero non negativo.

## A2 – Tree Repair Shop – Punti $\geq 10$

Avete memorizzato un grafo non orientato $G$ che rappresenta un albero non radicato, pronto per essere utilizzato come testcase per il corso di Algoritmi. Il vostro compagno di appartamento, da burlone qual è, vi ha fatto uno scherzo: ha aggiunto un arco e ora il grafo non è più un albero!
Scrivete un algoritmo

$$\mathsf{repairTree}(\textsc{Graph}\ G)$$

che prenda in input il grafo modificato e lo renda nuovamente un albero di $n$ nodi (non necessariamente quello originale), rimuovendo uno o più archi con l'operazione $G.\mathsf{deleteEdge}(x, y)$ e senza inserire nuovi archi.

Discutere informalmente la correttezza dell'algoritmo e la sua complessità computazionale. In questo caso, si farà attenzione ad algoritmi che hanno complessità definita da $O()$ oppure da $\Theta()$.

## A3 – $k$-Ripetizioni – Punti $\geq 12$

Scrivere un algoritmo

$$\textbf{int}\ \mathsf{krepeating}(\textsc{Item}[]\ S,\ \textbf{int}\ n,\ \textbf{int}\ k)$$

che prenda in input una stringa $S$ contenente $n$ caratteri e un valore intero positivo $k$ e restituisca la lunghezza della più lunga sottostringa contigua in cui tutti i caratteri presenti si ripetono almeno $k$ volte nella sottostringa.

Discutere informalmente la correttezza dell'algoritmo e la sua complessità computazionale. Una soluzione $\Omega(n^4)$ verrà valutata 50%. Esistono soluzioni $O(n^3)$ e $O(n^2)$.

Alcuni esempi:

- $S =$ `"baaab"`, $k = 3$:
  la risposta è 3 (la sottostringa `"aaa"`)
- $S =$ `"babbana"`, $k = 2$:
  la risposta è 5 (la sottostringa `"babba"`)
- $S =$ `"aabb"`, $k = 2$:
  la risposta è 4 (tutta la stringa)
- $S =$ `"banana"`, $k$?1:
  la risposta è 6 (tutta la stringa)
- $S =$ `"banana"`, $k = 3$:
  la risposta è 0 (nessuna sottostringa)

**Cognome**: **Nome**: **# Matricola**: **Riga**: **Col**:

# *Algoritmi e Strutture Dati – 9/1/2024 – Parte B*

**Esercizio -1** Iscriversi allo scritto entro la scadenza. In caso di inadempienza, -1 al voto finale.

**Esercizio 0** Scrivere correttamente nome, cognome, numero di matricola, riga e colonna su tutti i fogli consegnati. Consegnare foglio A4 e foglio protocollo di bella. In caso di inadempienza, -1 al voto finale.

## B1 – Commissioni di laurea – Punti $\geq 8$

L'ufficio Supporto alla Didattica ha chiesto il vostro aiuto. Il DISI ha realizzato un nuovo regolamento di laurea e comporre le commissioni sta diventando sempre più difficile.
Il problema da risolvere è il seguente:

- esistono $k$ aree didattiche;
- dovete creare $k$ commissioni, una per ogni area didattica;
- esistono $n$ professori; ogni professore $i$ è esperto di un'area didattica $p_i$;
- ogni commissione è composta da 5 professori in totale; almeno due di essi devono essere esperti nell'area didattica associata alla commissione; gli altri possono essere esperti oppure no;
- ogni professore può partecipare al massimo a una commissione.

Il vostro compito è trovare un assegnamento fra commissioni e professori che rispetti le regole del DISI. Descrivere un algoritmo che trovi tale assegnamento e discuterne la complessità computazionale.

## B2 – Tutti i cicli! – Punti $\geq 10$

Scrivere un algoritmo

$$\mathsf{printCycles}(\textsc{Graph}\ G,\ \textsc{Node}\ \mathit{start})$$

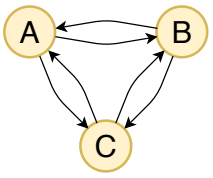
che prenda in input un grafo orientato $G$ e stampi tutti i cicli semplici che contengono *start*.

Discutere informalmente la correttezza dell'algoritmo e la sua complessità computazionale.

In un grafo completo composto da 3 nodi (tutti connessi l'uno con l'altro in entrambe le direzioni), come quello mostrato sotto e dato $\mathit{start} = A$, l'algoritmo deve stampare questi cicli, non necessariamente in quest'ordine:

$$A \to B \to A$$
$$A \to C \to A$$
$$A \to B \to C \to A$$
$$A \to C \to B \to A$$

Notate che distinguiamo i cicli $A \to B \to C \to A$ e $A \to C \to B \to A$ che sono formati dagli stessi nodi, ma percorsi in ordine inverso. In altre parole, stiamo considerando cicli orientati.




## B3 – Quanti quadrati! – Punti $\geq 12$

Scrivere un algoritmo

$$\textbf{int}\ \mathsf{countSquares}(\textbf{int}[][]\ A,\ \textbf{int}\ n)$$

che prenda in input una matrice $n \times n$ contenente valori 0 oppure 1, e restituisca il numero totale di sottomatrici quadrate contenenti solo valori 1.

Discutere informalmente la correttezza dell'algoritmo e la sua complessità computazionale. Esistono soluzioni con complessità $O(n^5)$, $O(n^4)$, $O(n^3)$ e $O(n^2)$.

Per esempio, nella tabella seguente:

| | | | |
|---|---|---|---|
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |

ci sono 10 sottomatrici quadrate di dimensione 1, 4 sottomatrici quadrate di dimensione 2 e 1 sottomatrice quadrata di dimensione 3, quest'ultima evidenziato in rosso; l'algoritmo deve quindi restituire il valore 15 $(10 + 4 + 1)$.