Algoritmi e Strutture Dati **Anno Accademico 2022–2023**

Esercizi: Ricorsione e non solo

Maneula Montangero

ATTENZIONE: nell'ultima pagina delle tracce ci sono dei suggerimenti per risolvere gli esercizi. Prima di guardarli cercate di pensare un pò alle soluzioni da soli, ma se non vi vengono idee andate a vedere i suggerimenti senza problemi.

1 Esercizi

Esercizio 1 (Calcolo n-esimo numero di Fibonacci senza ricorsione) *I numeri di Fibonacci sono definiti ricorsivamente nel modo sequente:*

$$Fib(n) = \begin{cases} 1 & \textit{se } n = 0 \\ 1 & \textit{se } n = 1 \\ Fib(n-1) + Fib(n-2) & \textit{altrimenti.} \end{cases}$$

A lezione abbiamo visto un algoritmo ricorsivo di costo computazionale $\Theta(2^n)$ per calcolare $Fib(n)$. Se ne dia una versione **iterativa** *(cioè che non usa la ricorsione) di costo computazionale $\Theta(n)$.*

Esercizio 2 (SelectionSort iterativo - fare dopo la lezione sul SelectionSort) *A lezione abbiamo visto un'implementazione ricorsiva dell'algoritmo di ordinamento* `SelectionSort`*. Scriverne una implementazione iterativa (che non usi, quindi, la ricorsione) e studiarne il costo computazionale.*

Esercizio 3 (Calcolo di potenza) *Dati due numeri interi positivi a e n, definiamo $a^n = \underbrace{a \cdot a \cdot \ldots \cdot a}_{n \textit{ volte}}$. Scrivere:*

1. *Un algoritmo iterativo che calcoli a^n e abbia costo computazionale $\Theta(n)$.*
2. *Un algoritmo ricorsivo per il calcolo di a^n che abbia costo computazionale $O(\log n)$.*

Esercizio 4 *Sia $A = A[0] \ldots A[n-1]$ una sequenza di n interi, che possono essere sia positivi che negativi. Si supponga che $A[0] \leq \ldots \leq A[n-1]$. Progettare un algoritmo ricorsivo che conti il numero di elementi > 0 in A. Si analizzi il costo computazionale di tempo dell'algoritmo proposto, giustificando le affermazioni fatte.*

Esercizio 5 *Si scriva una procedura ricorsiva che, dato in input un intero $n \geq 1$, stampi la lista di tutti i numeri interi compresi tra 1 e n, cioè $1, 2, 3, \ldots, n$. Se ne dia anche una versione iterativa.*

Esercizio 6 *Si scriva una procedura che dato in input un intero $n \geq 1$, stampi PRIMA la lista di tutti i numeri interi PARI compresi tra 1 e n, e DOPO la lista di tutti i numeri interi DISPARI compresi tra 1 e n. Esempio: se $n = 6$, la procedura prima stampa la lista $2, 4, 6$ e dopo $1, 3, 5$. Se ne dia una versione iterativa e una versione in cui si utilizza una procedura ricorsiva.*

Esercizio 7 *Si consideri il seguente problema: data una sequenza L di n interi, stampare tutti gli elementi della sequenza L che sono uguali alla differenza tra il valore dell'elemento immediatamente precedente e quello dell'elemento immediatamente successivo (esempio: se $L = 5, 2, 3, 1, 2$ allora si stampa $2, 1$). Si scriva una procedura ricorsiva e una iterativa per il problema.*

Esercizio 8 (aspettare la lezione sulla tecnica divide-et-impera) *Data una matrice M $n \times n$ di interi, per n potenza di 2, scrivere una funzione che determini l'elemento minimo in M utilizzando la tecnica* divide-et-impera.

Esercizio 9 *Una sequenza $A[0]A[1]\ldots A[n-1]$ di n elementi DISTINTI si dice* unimodale *se esiste un indice i, con $0 \leq i \leq n-1$, per cui $A[0] < A[1] < A[i]$ e $A[i] > A[i+1] > A[i+2] > \ldots > A[n-1]$ (se $i = 0$ o $i = n-1$ la sequenza è ordinata). Per esempio: la sequenza $2, 4, 5, 6, 8, 7, 3$ è unimodale e $i = 4$, mentre la sequenza $2, 4, 8, 6, 7, 3$ non lo è. Si scriva una funzione ricorsiva che, in tempo $O(\log n)$, trovi il massimo di una sequenza unimodale data in input.*

2 Suggerimenti

Suggerimento esercizio 1: calcolare l'n-esimo numero di Fibonacci partendo "dal basso", ovvero calcolare $Fib(2)$ usando $Fib(0)$ e $Fib(1)$, poi $Fib(3)$ usando $Fib(1)$ e $Fib(2)$, poi $Fib(4)$ usando $Fib(2)$ e $Fib(3)$, e così via fino ad arrivare a $Fib(n)$ calcolato usando $Fib(n-2)$ e $Fib(n-1)$.

Suggerimento esercizio 3.2: Calcolo di potenza. Per applicare Divide-et-Impera osserviamo che

$$a^n = a^{\lceil n/2 \rceil} \cdot a^{\lfloor n/2 \rfloor}.$$

Inoltre, $a^{\lceil n/2 \rceil} = a^{\lfloor n/2 \rfloor}$ se n è pari, e $a^{\lceil n/2 \rceil} = a \cdot a^{\lfloor n/2 \rfloor}$ se n è dispari.

Suggerimento esercizio 4: per l'algoritmo, preso l'indice centrale $k = \lfloor n/2 \rfloor$, il numero di valori > 0 è uguale alla somma dei valori > 0 che si trovano nella sottosequenza $A[0] \dots A[k]$ e dei valori > 0 che si trovano nella sottosequenza $A[k+1] \dots A[n-1]$.

Per il costo computazionale, osserviamo che, poiché gli elementi sono ordinati, al più una metà (tra quella sinistra e quella destra) può contentere sia numeri negativi che positivi.

Suggerimento esercizio 6: per la procedura con la ricorsione: definire una procedura ricorsiva che, preso un numero naturale n lo stampa se è > 0 e richiama se stessa con input $n-2$.

Suggerimento esercizio 7: Se L ha meno di tre elementi, non viene stampato nessun valore. In generale, invece, il primo valore che potrebbe essere stampato è il secondo della sequenza (ovvero $L[1]$), l'ultimo è il penultimo della sequenza (ovvero $L[n-2]$). Il problema può essere risolto (a) iterativamente, scorrendo gli elementi della sequenza e controllando, per ogni elemento, se questo debba essere stampato o no; (b) ricorsivamente, osservando che, controllato un elemento, quello che rimane da fare è risolvere esattamente lo stesso problema su una sequenza di numeri più corta (che perde il primo elemento).

Suggerimento esercizio 8: Dividere la matrice in quattro sottomatrici di dimensione $n/2 \times n/2$. Fare attenzione agli indici! In generale avrete a che fare con una matrice quadrata $m \times m$ con $m < n$, che sarà una sottomatrice qualunque di quella di partenza. È quindi necessario definire un modo in cui indicare la matrice che si sta considerando. Esempi: coppia di indici (i, j) che determinano l'angolo in alto a sinistra della matrice, più un valore m che indica la dimensione della matrice (quindi si potranno calcolare gli indici di riga e colonna degli altri angoli della matrice, per esempio quello in alto a destra sarà in $(i, i+m-1)$); oppure, una terna di indici (i, j, k) in cui (i, j) che determinano l'angolo in alto a sinistra della matrice e k che è l'indice di colonna dell'angolo in alto a destra (che quindi avrà indici (i, k) e avremo che la dimensione della matrice può essere calcolata come $m = k - i + 1$).

Suggerimento esercizio 9: Considerare l'elemento nel centro della sequenza e determinare se il massimo si trova a sinistra o destra di quello centrale (attenzione: quello centrale potrebbe essere il massimo).