Prova d'esame di Laboratorio di Programmazione Strutturata
per il corso di laurea in Scienze e Tecnologia dei Media
Prova di programmazione in laboratorio – 15 Settembre 2014

Tema d'esame Conversione di un numero intero positivo nella sua rappresentazione in numeri romani

Metodo di rappresentazione dei numeri romani
Consideriamo un numero intero positivo $n < 4000$. L'algoritmo che ci consente di determinare la rappresentazione in numeri romani di n può essere riassunto come segue.

(1) Si calcolino le unità u, le decine d, le centinaia c e le migliaia m che compongono il numero n, in modo tale che

$$n = m * 1000 + c * 100 + d * 10 + u \ ,$$

dove i numeri interi u, d, c e m sono tutti compresi tra 0 e 9.

(2) La rappresentazione finale di un numero romano si ottiene per "accostamento" (a destra) dei caratteri che la compongono, cominciando prima dai caratteri che corrispondono alle migliaia, poi a quelli delle centinaia, quindi quelli delle decine, infine quelli delle unità.

(3) Come esempio, si ricordi che le regole di rappresentazione delle unità sono le seguenti.
  (3a) *Se* $u = 9$, *allora* aggiungiamo alla rappresentazione del numero romano i caratteri **I** e **X**, *altrimenti* passiamo a considerare i seguenti punti (3b)–(3d).
  (3b) *Se* $u = 4$, *allora* aggiungiamo alla rappresentazione del numero romano il carattere **I** e incrementiamo di uno il valore di u; poi passiamo a considerare i seguenti punti (3c)–(3d).
  (3c) *Se* $u \geq 5$, *allora* aggiungiamo alla rappresentazione del numero romano il carattere **V** e decrementiamo di cinque il valore di u; quindi consideriamo il seguente punto (3d).
  (3d) Si aggiunga u volte il carattere **I** alla rappresentazione del numero romano (dove si intende che il valore di u non è quello definito inizialmente al punto (1), ma quello che può essere stato modificato come ai punti (3b)–(3c)).

(4) Le regole di rappresentazione delle decine sono analoghe a quelle descritte ai punti (3a)–(3d), dove l'unica differenza consiste nel sostituire, rispettivamente, ai simboli **I**, **V** e **X** gli altri caratteri **X**, **L** e **C**.

(5) Le regole di rappresentazione delle centinaia sono analoghe a quelle descritte ai punti (3a)–(3d), dove l'unica differenza consiste nel sostituire, rispettivamente, ai simboli **I**, **V** e **X** gli altri caratteri **C**, **D** e **M**.

(6) Le regole di rappresentazione delle migliaia sono analoghe a quelle descritte ai punti (3a)–(3d), dove l'unica differenza consiste nel sostituire, rispettivamente, il simbolo **I** con il carattere **M**, mentre possiamo rimpiazzare **V** e **X** con degli spazi (questa ultima sostituzione non sarà mai effettuata poiché abbiamo supposto che $n < 4000$).

Obiettivo (intermedio) 1:
si scriva un programma in linguaggio **C**, dove si effettua la stampa su video delle unità u, delle decine d, delle centinaia c e delle migliaia m che compongono il numero intero

positivo n, come descritto al punto (1) dell'algoritmo di cui sopra. Il programma deve contenere:

(a) la gestione dell'input da tastiera del valore di n che deve essere compreso tra 0 e 4000 (esclusi gli estremi);

(b) una funzione che ha come argomento il solo valore di n, ne esegue la scomposizione in migliaia, centinaia, decine e unità e stampa i valori di m, c, d e u.

Un consiglio: la scomposizione in migliaia, centinaia, decine e unità di un numero può essere ottenuta facilmente utilizzando ripetutamente solo gli operatori / (divisione intera) e % (resto della divisione intera).

Obiettivo (intermedio) 2:
si modifichi il programma richiesto dall'obiettivo 1 in modo tale che venga stampata la rappresentazione in numeri romani di n.

Alcuni consigli: il modo piú intuitivo per adempiere alle richieste del presente obiettivo 2 consiste nel modificare solamente la funzione richiesta dall'obiettivo 1, in modo che sia in essa contenuto tutto l'algoritmo di conversione nella rappresentazione in numeri romani, come è stato descritto sopra. Così facendo, si è però costretti a ricopiare molte volte le stesse istruzioni e ad apportare poche modifiche alle istruzioni stesse (con gravi rischi di banali errori di trascrizione).

Un modo più elegante, consiste nello scrivere un'altra funzione che ha quattro argomenti: una cifra compresa tra 0 e 9, e tre stringhe di due caratteri (si faccia attenzione a non dimenticare che una stringa deve contenere l'indicatore **NUL** di fine testo), che conterranno rispettivamente i simboli **I**, **V** e **X** quando vogliamo trattare le unità, oppure **X**, **L** e **C** nel caso delle decine, o **C**, **D** e **M** per le centinaia o **M** e due spazi per le migliaia. Sarà questa nuova funzione ad incaricarsi di stampare su video i caratteri che costituiscono la rappresentazione in numeri romani e dovrà essere chiamata quattro volte (una prima volta per le migliaia, poi per le centinaia, per le decine e le unità).

Obiettivo (intermedio) 3:
si modifichi il programma richiesto dall'obiettivo 2 in modo che la stampa della rappresentazione in numeri romani del numero n introdotto in input sia effettuata alla fine del "main program". A questo scopo, le funzioni descritte negli obiettivi 1 e 2 dovranno trascrivere la rappresentazione in numeri romani all'interno di una stringa, che dovrà essere inclusa nella lista di argomenti delle funzioni stesse.

Nota: tutte le operazioni di "aggiunta" o di "accostamento" di caratteri, descritte ai punti (2)–(6) dell'algoritmo precedentemente descritto, possono essere tradotte in termini di *concatenazione di stringhe* all'interno del programma.

Obiettivo (finale) 4:
si modifichi il programma richiesto dall'obiettivo 3 in modo tale che esso legga sequenzialmente dei numeri interi positivi (< 4000) da un file di input e scriva (nello stesso ordine in cui li legge) la loro corrispondente rappresentazione in numeri romani su un file di output. Ovviamente, prima dell'esecuzione, il file di input deve essere opportunamente preparato in modo che contenga un elenco di numeri. È preferibile che il programma riesca a trattare la fine della lettura del file di input senza conoscere a priori quanti numeri sono contenuti nel file stesso.