Informatica Generale
Homework 1: Algoritmi sui Numeri Naturali

docente: Ivano Salvo
Sapienza Università di Roma

pubblicazione: 4.IV.2014 - consegna 28.IV.2014

Esercizio 1 Scrivere un programma C che presi in input due interi positivi n ed m ($n, m > 0$) restituisca in output il *minimo comune multiplo* tra n ed m. Un modo facile è adattare opportunamente una delle funzioni viste per il massimo comun divisore (vedi file `naturali.c` nella pagina delle dispense).

Esempi: Presi in input 12 e 8, il programma deve essere restituito come risultato 24. Presi in input 7 e 13, deve essere restituito come risultato 91.

Esercizio 2 Scrivere un programma C che presi in input due interi positivi a ed b ($a, b > 0$) calcola il logaritmo intero di a in base b.

Più formalmente, il programma deve restituire in output un numero intero k, per cui sia verificata la disuguaglianza $b^k \leq a < b^{k+1}$.

Esempi: Presi in input 63 e 2, il programma deve stampare 5, in quanto $2^5 = 32 \leq 63 < 64 = 2^6$. Presi in input 81 e 3, il programma deve stampare 4, in quanto $3^4 = 81 < 243 = 3^5$.

Esercizio 3 (Facoltativo – vale un bonus di 1 punto sul voto finale)

Consideriamo la seguente successione di numeri naturali: $u_0 = 1, u_1 = 2$ e $u_n (n > 1)$ è il *minimo* numero naturale che si può scrivere in *modo unico* come somma di due precedenti numeri della successione. La successione sopra definita comincia con $1, 2, 3, 4, 6, 8, 11, 13, 16, 18, 26, 28, \ldots$.

Per capire meglio la definizione, osservate che u_4 non può essere 5, in quanto $5 = 2+3 = u_1+u_2$, ma anche $5 = 1+4 = u_0+u_3$. Viceversa solo $u_1+u_3 = 2+4$ danno come somma 6.

Scrivere un programma C che preso in input un numero naturale n scrive in output u_n. Ad esempio, se l'input è 4, il programma dovrà rispondere in output 6. Se l'input fosse 11, l'output dovrebbe essere 28.

Suggerimenti: ricevuto in input il numero n allocate un vettore `u` con $n+1$ posizioni (indicizzate da 0 a n), caricate i valori iniziali in `u[0]` e `u[1]` e calcolatevi iterativamente tutta la successione partendo da `u[2]` fino a `u[n]`.

Potete ovviamente seguire molte strade. Per facilitare la ricerca del $k+1$-esimo elemento u_{k+1} (una volta noti $u_0, u_1, \ldots, u_k$) osservate che necessariamente si ha che $u_k + 1 \leq u_{k+1} \leq u_{k-1} + u_k$. Infatti, se ci fosse qualche numero minore di u_k che si scrive in modo unico come somma di due precedenti, sarebbe già stato inserito nella successione. Inoltre, essendo la successione strettamente crescente, è ovvio che $u_{k-1} + u_k$ si scrive in modo unico come somma di due precedenti (in quanto questa somma è strettamente maggiore di ogni altra somma di due precedenti) e ciò, tra l'altro, dimostra che la successione è infinita.