Informatica Generale
Homework 1/2017: Ordinatamente in Successione

Ivano Salvo – Sapienza Università di Roma – 10/4/2017

Esercizio 1 Considerate il problema del "$3x + 1$ *problem*" (*Homework di Prova*, Esercizio **2**). Scrivere una funzione C di prototipo:

```
void treXPUnoSucc(int n, int txpu[])
  /* PREC: n>0 */
```

che ricevendo un numero n carica il vettore $txpu$ in modo che $txpu[i]$ ($0 \leq i < n$) sia il numero di passi necessari affinché la sequenza che comincia con i raggiunga 1 (attenzione: $txpu[0]$ non è definito, essendo 0 un punto fisso della trasformazione, che non porta a 1).

Potete, banalmente, limitarvi ad invocare opportunamente la funzione *treXPUno* scritta nell'Homework precedente, oppure, con un po' di fatica, scrivere una funzione più efficiente che evita di ricalcolare sequenze già incontrate.

Esempio: Se $n = 20$, la funzione carica il vettore $txpu$ con i valori {0, 1, 7, 2, 5, 8, 16, 3, 19, 6, 14, 9, 9, 17, 17, 4, 12, 20, 20}.

Esercizio 2. Considerare il problema delle k-uple *ordinate* $\langle p_1, \ldots, p_k \rangle$ di interi positivi ($1 \leq p_1 \leq p_2 \leq \ldots \leq p_k \leq n$) il cui prodotto è n (vedi *Homework di Prova*, Esercizio **3**).

Se ci sono m k-uple di prodotto n, immaginatele ordinate secondo l'ordine lessicografico e numerate in tale ordine da 0 a $m - 1$. Scrivere una funzione C di prototipo:

```
int kuplaPM(int n, int k, int p, int kupla[])
```

che carica nel vettore *kupla* la k-upla p-esima, se $p < m$ (m numero di k-uple di n) ritornando 1 come risultato. Se, viceversa $p >= m$ torna semplicemente 0.

Esempi: Per ogni k, se n è primo l'unica k-upla che dà come prodotto n è $\langle \underbrace{1, 1, 1, \ldots 1}_{k-1}, n\rangle$. Quindi la funzione *kuplaPM* caricherà il vettore rispondendo 1 solo se il parametro p ha valore 0.

Se n fosse 12, k fosse 4, e p fosse 2, la funzione risponde 1 e carica il vettore *kupla* con i valori $\{1, 1, 3, 4\}$. Infatti ho che 12 può essere ottenuto come prodotto delle seguenti 4 k-uple (scritte nell'ordine lessicografico): $\langle 1, 1, 1, 12\rangle$, $\langle 1, 1, 2, 6\rangle$, $\langle 1, 1, 3, 4\rangle$, $\langle 1, 2, 2, 3\rangle$.

Esercizio 3. Dato un insieme P di $k > 0$ numeri naturali $\{p_0, \ldots p_{k-1}\}$, definiamo *induttivamente* l'insieme C_P dei numeri naturali che sono composti dei soli elementi in P, come il *minimo* insieme che soddisfa le seguenti:

$$1 \in C_P$$
$$n \in C_P \Rightarrow n \cdot p_j \in C_P \quad (p_j \in P)$$

Alternativamente, potete definire C_P come l'insieme di tutti i numeri naturali nella forma $p_0^{n_0} \cdot p_1^{n_1} \cdot \ldots \cdot p_{k-1}^{n_{k-1}}$, con $n_j \geq 0$ per ogni $j \in [0, k)$ (notate che, in accordo con la definizione induttiva, 1 appartiene sempre a C_P per ogni insieme P).

Considerate il problema di generare la successione *ordinata* dei numeri naturali in C_P e scrivere una funzione C di prototipo:

```
int * generaNComposti(int n, int p[], int k)
   /* PREC: n,k>0 */
```

che presi come parametri di ingresso un numero n, e un vettore p di k numeri distinti *alloca* un vettore *dinamico* di lunghezza n e lo carica con i primi n numeri naturali in S, ordinati in modo crescente.

Esempio: Se i parametri fossero $n = 25$, $k = 3$ e il vettore $p = \{2, 3, 5\}$, occorrerebbe caricare il vettore risultato con i numeri:

$$\{1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 16, 18, 20, 24, 25, 27, 30, 32, 36, 40, 45, 48, 50, 54\}$$

Note Pratiche: vedi note pratiche dell'Homework di Prova.