

TestiGara - OttavaEdizione

Esercizi di Gara della VIII edizione

AVVISI:

- Se non specificato altrimenti negli esercizi, le sequenze iniziali su nastro si intendono *non vuote*, ovvero contenenti almeno un simbolo.
- Per numero decimale si intende un numero positivo o nullo rappresentato con le cifre 0, 1, 2, ..., 9, senza zeri iniziali non significativi; per esempio 0 e 19 sono numeri validi, mentre 0032 deve essere scritto come 32.
- Nel fornire le soluzioni, ricordarsi di pulire il nastro finale da ogni simbolo che non costituisca la risposta!
- Ogni volta che si salva la soluzione di un esercizio con il simulatore della macchina di Turing, il "timestamp" dell'esercizio viene aggiornato con il tempo trascorso fino a quel momento.

Esercizio 1: Quadristringhe [Punti 15] Una *quadristringa* è una stringa su un alfabeto dato che ha la forma XXXX, ovvero, che è composta di quattro parti uguali ripetute. Per esempio, abcabcabcab è una quadristringa. Si scriva un programma che, data in ingresso una stringa sull'alfabeto a,b,c, lasci sul nastro SI se la stringa di ingresso è una quadristringa, NO altrimenti. Esempi:

NASTRO INIZIALE	NASTRO FINALE
abababab	SI
bacbacbacbac	SI
cbabcbcb	NO
cccc	SI
aaaaa	NO

Esercizio 2: Tristringhe [Punti 25]. Analogamente all'esercizio precedente, una *tristringa* è una stringa che ha la forma XXX, ovvero, che è composta di tre parti uguali ripetute. Per esempio, abcabcab è una tristringa. Si scriva un programma che, data in ingresso una stringa sull'alfabeto a,b,c, lasci sul nastro SI se la stringa di ingresso è una tristringa, NO altrimenti. Esempi:

NASTRO INIZIALE	NASTRO FINALE
aaa	SI
bbbbbb	SI
abababab	NO
babacbabacbabac	SI

Esercizio 3: I multidispari I --- riconoscimento di espressioni [punti 15]. I numeri *multidispari* sono quelli che hanno la forma $(2(2(2(\dots(2+1)\dots)+1)+1)+1)$. Sono multidispari $2+1=3$, $2(2+1)+1=7$, $2(2(2+1)+1)+1=15$, ecc. Si scriva un programma che, ricevuta in ingresso una stringa sull'alfabeto {1,2,+,[,] (in cui le parentesi quadre prendono il posto delle tonde), lasci sul nastro SI se la stringa ricevuta è l'espressione di un numero multidispari racchiusa fra parentesi, NO altrimenti. Esempi:

NASTRO INIZIALE	NASTRO FINALE
-----------------	---------------

[2+1]	SI
[[1]+2]	NO
2+2	NO
2[2+2]	NO
[2[2[2[2+1]+1]+1]+1]	SI
2[2[2[2+1]+1]+1]+1	NO (non è racchiusa fra parentesi)

Esercizio 4: I multidispari II --- calcolo [Punti 25]. Con riferimento all'esercizio precedente, si scriva un programma che, ricevuta in ingresso una espressione valida per un numero multidispari (sempre racchiusa fra parentesi), lasci sul nastro il numero decimale corrispondente al risultato del calcolo. Esempi:

NASTRO INIZIALE	NASTRO FINALE
[2+1]	3
[2[2+1]+1]	7
[2[2[2[2[2[2+1]+1]+1]+1]+1]+1]+1]	255

Esercizio 5: I multidispari III --- riconoscimento dei valori [Punti 20]. Con riferimento agli esercizi precedenti sui multidispari, si scriva un programma che, ricevuto in ingresso un numero decimale, lasci sul nastro SI se il numero in questione è un multidispari, NO altrimenti. Esempi:

NASTRO INIZIALE	NASTRO FINALE
3	SI
7	SI
35	NO
511	SI
2041	NO

Esercizio 6: Conversione esadecimale-binario [Punti 15]. I numeri in base 16, o *esadecimali*, utilizzano come cifre le consuete cifre decimali 0..9, più le lettere A, B, C, D, E, F che rappresentano, rispettivamente, 10, 11, 12, 13, 14, 15. Per esempio, il numero esadecimale 2B vale $2 \times 16^1 + 11 \times 16^0$, ovvero 43 in decimale. I numeri in base 2, o *binari*, utilizzano invece esclusivamente le cifre 0 e 1. Per esempio, 1001 in binario vale $1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$, ovvero 9 in decimale. Si scriva un programma che, ricevuto in ingresso un numero esadecimale, lasci sul nastro lo stesso numero espresso in notazione binaria. **Suggerimento:** si noti che una cifra esadecimale corrisponde al più a 4 cifre binarie. Esempi:

NASTRO INIZIALE	NASTRO FINALE
1	1
A	1010
F1ABA	11110001101010111010
B8	10111000
20	100000
0	0

Esercizio 7: Vettori fagici [Punti 30]. Il DNA degli esseri viventi è costituito da una lunga catena di *basi* (adenina, timina, guanina, citosina, indicate rispettivamente con A,T,G,C). Su una sequenza di basi operano degli *enzimi*, che per via chimica possono operare trasformazioni sulla sequenza di DNA. In particolare, i *vettori fagici* sono in grado di eliminare certe sotto-sequenze di DNA, ricongiungendo le estremità del frammento dopo la rimozione. Si scriva un programma che simuli il funzionamento di un vettore fagico. Il programma, ricevute in ingresso due stringhe sull'alfabeto {A,T,G,C} separate da X, rimuove tutte le occorrenze disgiunte della prima stringa all'interno della seconda stringa, e lascia sul

nastro la sola stringa risultante. Nel caso di più occorrenze sovrapposte (per esempio: se si vuole rimuovere ATA da CATATAG), si rimuova soltanto l'occorrenza più a sinistra (producendo come risultato CTAG). Esempi:

<i>NASTRO INIZIALE</i>	<i>NASTRO FINALE</i>
AXCTAAGAC	CTGC
ACCXCACTACCTACCAG	CACTTAG
CTTAXGATTACA	GATTACA
ATAXCATATAG	CTAG
ATTAXATTAATTA	

Esercizio 8: XOR esadecimale [Punti 20]. Si scriva un programma che, dati in ingresso due numeri esadecimali (si veda l'esercizio 6) separati da un punto, lasci sul nastro il risultato dell'*or esclusivo* (o *xor*) applicato ai due numeri. L'operatore xor è definito come segue: se nella rappresentazione binaria dei due operandi le cifre nella stessa posizione hanno lo stesso valore, allora la cifra corrispondente del risultato sarà 0; altrimenti, essa sarà 1. Per esempio, A2 xor 1F = DD, come si può vedere dal seguente schema:

A2 =	1	0	1	0	0	0	1	0	
1F =	0	0	0	1	1	1	1	1	
A0 xor 1F	1	0	1	1	1	1	0	1	= BD
=									

Si assuma che i numeri esadecimali passati in ingresso abbiano la stessa lunghezza, e si produca un risultato della stessa lunghezza. Per questo esercizio, sono ammessi gli 0 iniziali. Esempi:

<i>NASTRO INIZIALE</i>	<i>NASTRO FINALE</i>
A5.BC	19
1B4.020	194
1.1	0
0A224.55555	5F771
00.00	00

Esercizio 9: Gruppi ciclici [Punti 30]. Un *gruppo ciclico* $G(n,k)$, con $0 < n < k$, è una *sequenza* di interi tali che il primo elemento è sempre 0, e quelli successivi sono calcolati aggiungendo n al precedente e calcolando il risultato modulo k . La sequenza si interrompe quando viene generato un numero già presente. Per esempio, $G(2,5) = \{0,2,4,1,3\}$; $G(3,6) = \{0,3\}$; $G(1,5) = \{0,1,2,3,4\}$. Si scriva un programma che, presi in ingresso n e k , con $1 < k < 10$, lasci sul nastro il gruppo ciclico $G(n,k)$, nell'ordine corretto. Esempi:

<i>NASTRO INIZIALE</i>	<i>NASTRO FINALE</i>
25	02413
36	03
49	048372615
39	036
14	0123
28	0246

Esercizio 10: Il lambda-calcolo [Punti 35]. Il **lambda**-calcolo è un formalismo per la definizione e l'applicazione di funzioni. Una *lambda-espressione* è costituita da una dichiarazione di *parametri*, preceduti dalla lettera L, seguita da una espressione; le due parti sono separate da un punto. Per esempio,

la lambda-espressione $Lx.x$ rappresenta la funzione $f(x)=x$ (si noti però che le lambda-espressioni non introducono nomi per le funzioni: nel primo caso, non compare il nome f). L'applicazione di una funzione a un argomento viene denotata scrivendo l'argomento a destra della funzione: per esempio, $(Lx.x)3$ rappresenta la funzione $f(x)=x$ applicata a 3, ovvero $f(3)$, e dunque vale 3). Le applicazioni possono anche essere multiple: in questo caso, si calcola prima l'applicazione più a sinistra e più interna rispetto alle parentesi. Per esempio, $(Lx.x+1)((Lx.2x)3) = (Lx.x+1)6 = 7$, mentre $(Lx.xx)(Lx.x)3 = (Lx.x)(Lx.x)3 = (Lx.x)3 = 3$.

Useremo un lambda-calcolo semplificato con una sola variabile (x), e un solo termine base (e). Una lambda-espressione è una sequenza di elementi, ciascuno dei quali può essere un termine oppure una applicazione. Un termine può essere una x oppure una e . Una applicazione ha la forma $[lambda-espressione]termine$ oppure $[lambda-espressione]applicazione$. Si noti che, visto che abbiamo una sola variabile, non occorre avere un simbolo distinto per " Lx ": è sufficiente la parentesi quadra.

Si scriva un programma che, ricevuta in ingresso una lambda-espressione, lasci sul nastro il risultato della sua valutazione secondo le regole sopra espone.

Suggerimento: si scriva il programma in modo che esso applichi ripetutamente una riscrittura di una applicazione (la più interna a sinistra). Per esempio:

$[xx]ex \rightarrow eex$

$[x[xx]e]x \rightarrow [xee]x \rightarrow xee$

$[[xex]e]x \rightarrow [eee]x \rightarrow eee$

$[[xe]x][ex]e \rightarrow [xe][ex]e \rightarrow [xe]ee \rightarrow eee$

Esempi:

<i>NASTRO INIZIALE</i>	<i>NASTRO FINALE</i>
e	e
$[x]e$	e
$[xx][x]x$	xx
$[xx][x]e$	ee
$[x[xx]e]x$	xee
x	x
$[ex[x]e]x$	exe