TestiGara - DecimaEdizione

Esercizi di Gara della X 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: Addizione unaria [Punti 7]. Nei sistemi di numerazione posizionale, il valore denotato da un numero è ottenuto moltiplicando ogni cifra per la corrispondente potenza di una base. Nel caso della notazione decimale, la base è 10; la cifra più a destra è moltiplicata per 10^0=1, quella successiva per 10^1=10, la terza da destra per 10^2=100, e così via. Nella numerazione con base 1 (unaria), ciascuna cifra (si usa il carattere "1" per convenzione) viene moltiplicata per 1^n=1, qualunque sia *n*. Per esempio, 12_{10} = 111111111111_1. Si noti che lo 0 si esprime in unario con l'assenza completa del numero. Si scriva un programma per Macchina di Turing che, ricevuta sul nastro una addizione unaria (composta da due numeri unari separati da "+"), lasci sul nastro il risultato dell'addizione, sempre espresso in unario.

NASTRO INIZIALE	*NASTRO FINALE*
11+111	11111
1111111+1	11111111
1+1	11
1+	1
+	

Esercizio 2: Addizione binaria [Punti 10]. La notazione binaria è un sistema di numerazione posizionale con base 2. In binario le cifre (0 o 1) vengono moltiplicate per 2^0=1, 2^1=2, 2^2=4, e così via, in base alla loro posizione. Per esempio, 12_{10} = 1100_2. Si scriva un programma per Macchina di Turing che, ricevuta sul nastro una addizione binaria (composta da due numeri binari separati da "+"), lasci sul nastro il risultato

dell'addizione, sempre espresso in binario.

NASTRO INIZIALE	*NASTRO FINALE*
11+111	1010
1+1	10
0+0	0
10001+1101001	1111010
11011001101+1	11011001110

Esercizio 3: Conversione romana [Punti 28]. La notazione romana è un sistema di numerazione non-posizionale. I romani lo adattarono dal sistema Etrusco, e la sua forma moderna si stabilizzò intorno al XIII secolo. In questo sistema si usano alcune lettere per indicare determinati numeri; quando una lettera di valore minore segue una di valore maggiore, si intende che i due valori vadano sommati; quando invece una lettera di valore minore precede una di valore maggiore, si intende che il valore della prima vada sottratto da quello della seconda. Le lettere usate generalmente sono: I=1, V=5; X=10; L=50; C=100; D=500; M=1000 (per esprimere numeri maggiori venivano posti vari segni moltiplicatori sopra o accanto alle lettere usuali). Si noti che i Romani non avevano un segno per lo 0, e che il sistema è ambiguo: per esempio, 99 si può scrivere sia IC che XCIX (entrambe le forme sono attestate in iscrizioni). Noi adotteremo quest'ultima forma, in cui ogni cifra della rappresentazione decimale è codificata indipendentemente: 99 = 90 + 9 = XC IX. Si scriva un programma per macchina di Turing che, ricevuto sul nastro un numero romano fra 1 e 3000, lasci sul nastro il corrispondente numero decimale.

NASTRO INIZIALE	*NASTRO FINALE*
I	1
IV	4
IX	9
XIX	19
XCIX	99
DCLXVI	666
MCMXLV	1945
MMVI	2006

Esercizio 4: Addizione romana [Punti 20]. Con riferimento alla notazione romana introdotta nell'esercizio

precedente, si scriva un programma per macchina di Turing che, ricevuta sul nastro una *addizione romana* formata da due numeri romani compresi fra 1 e 10 separati dalla locuzione “ et “ (uno spazio, 'e', 't', uno spazio), lasci sul nastro il risultato della loro addizione, sempre espresso in notazione romana.

NASTRO INIZIALE	*NASTRO FINALE*
I ET I	II
III ET I	IV
IX ET X	XIX
IV ET IV	VIII
X ET V	XV

Esercizio 5: Conversione babilonese [Punti 35]. Il sistema di numerazione babilonese, stabilizzatosi fra il 1900 e il 1800 a.C., era un sistema posizionale con base 60. La scelta di questa base rendeva semplici le divisioni per 2, 3, 5, 6, 10, 12, 15, 20 e 30 (tutti fattori di 60), nonché quelle per i loro prodotti (4, 6, 8, 18, 24, ecc.). D'altro canto, l'uso di una base 60 obbliga a usare 59 simboli diversi per rappresentare le cifre (più lo spazio, che rappresentava lo 0). Fortunatamente, i Babilonesi usavano il sistema cuneiforme, e il simbolo di ciascuna cifra era ottenuto incidendo sulle tavolette un certo numero di simboli "<" (che indicavano le decine) e "Y" (che indicavano le unità). Per esempio, la cifra 43 era rappresentata dal simbolo 43 nella tabella accanto, e che noi indicheremo con "<<<<YYY". Naturalmente, se tale simbolo compariva nella posizione più a destra di un numero il suo valore era moltiplicato per 60^0=1; in quella successiva era moltiplicato per 60, poi per 3600, ecc. Noi rappresenteremo le cifre babilonesi separandole con un ".", e usando il simbolo "=" al posto dello spazio per denotare lo 0. Si scriva un programma che, ricevuto sul nastro in ingresso un numero in notazione babilonese, lasci sul nastro in uscita il corrispondente numero in notazione decimale.

NASTRO INIZIALE	*NASTRO FINALE*
YYYY	4

<	10
<<<YY	32
Y.<<YYY	83
<.=.YY	36002
<<<YYY.<<YYYYYY	2006
<.=	600

Esercizio 6: Sequenza di pari [Punti 10]. Si scriva il programma di una Macchina di Turing che, dato un numero sul nastro, stampi la sequenza dei numeri pari compresi fra 0 e il numero dato, separati da ".".

NASTRO INIZIALE	*NASTRO FINALE*
5	0.2.4
0	0
2	0.2
17	0.2.4.6.8.10.12.14.16

Esercizio 7: Densità di una sequenza [Punti 15]. Si scriva il programma di una macchina di Turing che, data una sequenza di numeri sul nastro, separati da '.' e compresi tra 0 e 20, lasci sul nastro S se questa contiene almeno 10 numeri distinti, N altrimenti.

NASTRO INIZIALE	*NASTRO FINALE*
0.2.5.6.7.8.10.11.19.11.13	S
0.0.1.5.1.0.11.10.20.20.20	N
10	N
0.0.1.2.3.4.5.6.7.8.9	S

Esercizio 8: Verso di una sequenza [Punti 18]. Una sequenza di interi $n_1...n_k$ (con $k \geq 2$) è detta *crescente* se ciascuno degli interi successivi al primo è maggiore del precedente (ovvero, $\partial i>1$, $n_i>n_{i-1}$); *decrescente* se ciascuno degli interi successivi al primo è minore del precedente (ovvero, $\partial i>1$, $n_i<n_{i-1}$), *incerta* in tutti gli altri casi. Si scriva il programma di una macchina di Turing che, data una sequenza di numeri sul nastro, separati da '.', lasci sul nastro C se la sequenza è crescente, D se è decrescente, I se è incerta.

NASTRO INIZIALE	*NASTRO FINALE*

0.2.4.5	C
10.15.23.20.6	I
10.5	D
25.20.3.3.0	I

Esercizio 9: Numeri perfetti [Punti 22]. Un numero si dice *perfetto* se la somma dei suoi divisori propri ha come risultato il numero stesso. Sono perfetti 6 (1+2+3=6), 28 (1+2+4+7+14=28), 496, ecc. Si scriva un programma per la macchina di Turing che, dati sul nastro un numero e la lista dei suoi divisori, lasci sul nastro P se il numero è perfetto, N altrimenti. Si assuma che il nastro inizialmente sia della forma *N*?*D*+*D*+*D*..., dove *N* è il numero e i *D* sono i suoi divisori.

NASTRO INIZIALE	*NASTRO FINALE*
6?1+2+3	P
7?1	N
15?1+3+5	N
28?1+2+4+7+14	P
30?1+2+3+5+6+10+15	N

Esercizio 10: Numeri di Padovan [Punti 30]. I *numeri di Padovan* sono definiti dalla seguente formula per ricorrenza:

I primi numeri di Padovan sono dunque, nell'ordine, 1, 1, 1, 2, 2, 3, 4, 5, 7, 9, 12, 16, 21, 28, ... Si scriva un programma per Macchina di Turing che, ricevuto sul nastro in ingresso un intero *n*, lasci sul nastro in uscita l'*n*-esimo numero di Padovan *P*(*n*).

NASTRO INIZIALE	*NASTRO FINALE*
1	1
8	5
14	28
27	1081
44	128801