**Problemi assegnati nelle gare precedenti**

**Esercizi di Gara della III edizione**

1. Programmare una macchina di Turing che, dato un nastro iniziale contenente un numero intero *n* compreso tra 1 e 9, termina la sua esecuzione lasciando sul nastro *n* A consecutive.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| 1 | A |
| 5 | AAAAA |
| 9 | aaaaaaaaa |

1. Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza di *n* A consecutive (con *n*>0), termina la sua esecuzione lasciando sul nastro il numero *n*.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| A | 1 |
| AAAAAA | 6 |
| AAAAAAAAAAA | 11 |

1. Programmare una macchina di Turing che, dato un nastro iniziale contenente due numeri interi positivi *x* e *y* separati da una cella vuota tali che *x*>*y* e 9>=*y*>0, termina la sua esecuzione lasciando sul nastro soltanto la differenza tra *x* e *y*.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| 9-4 | 5 |
| 13-6 | 7 |
| 302-3 | 299 |

1. Indichiamo con *S* una sequenza formata da A, B o C ed indichiamo con *x* e *y* un simbolo che sia A o B. Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza del tipo *xyS* termina la sua esecuzione lasciando sul nastro la sequenza ottenuta da *S* rimpiazzando tutte le occorrenze di *x* con *y*.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| ABcAB | CBB |
| BBABC | ABC |
| BA |  |

1. Programmare una macchina di Turing che, dato un nastro iniziale contenente due sequenze di A separate da una D, termina la sua esecuzione lasciando sul nastro la sequenza che contiene il maggior numero di A.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| AADA | AA |
| AADAAA | AAA |
| AADAA  DA | AA  A |

1. Indichiamo con *S* e *T* due sequenze non vuote e della stessa lunghezza formate da A o B. Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza del tipo *S*D*T*, termina la sua esecuzione lasciando sul nastro nastro la sola sequenza SI se *T* è un anagramma di *S*, la sola sequenza NO altrimenti.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| ABADBAA | SI |
| BABADABBA | SI |
| ABBDBAA  ABADABB | NO  NO |

1. Programmare una macchina di Turing che, dato un nastro iniziale contenente un numero intero (arbitrariamente grande), termina la sua esecuzione lasciando sul nastro la sola sequenza SI se il numero è divisibile per 3, la sola sequenza NO altrimenti.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| 3 | SI |
| 27 | SI |
| 81 | SI |
| 20 | NO |
| 7676585 | NO |

1. Una sequenza di parentesi si dice *bilanciata* secondo la seguente definizione induttiva:
   1. la sequenza vuota è bilanciata,
   2. se *S* e *T* sono sequenze bilanciate allora anche la sequenza **(** *S* **)** *T* è bilanciata.

Rappresentando **(** con B e **)** con E, programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza di B ed E, termina la sua esecuzione lasciando sul nastro la sola sequenza SI se la sequenza è bilanciata, la sola sequenza NO altrimenti.

**Esempi**

|  |  |
| --- | --- |
| nastro iniziale | nastro finale |
| BEBE | SI |
| BBBEEE | SI |
| BBEBEE  BBBEBEEBEE | SI  SI |
| BEE | NO |
| BBEEEB | NO |

**Esercizi di Gara della V 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!

**Esercizio 1 [Sostituzione di caratteri].** Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza arbitraria di simboli A e B, *sostituisca* ogni occorrenza di due simboli consecutivi AB con due simboli CD. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| AABABBBAABAAABAABAA | ACDCDBBACDAACDACDAA |
| BBBBAAA | BBBBAAA |
| AABB | ACDB |

**Esercizio 2 [Parentesi bilanciate].** Una sequenza di parentesi quadre e graffe annidate si dice *bilanciata* secondo la seguente definizione: (i) la sequenza vuota è bilanciata; (ii) se *S* e *T* sono sequenze bilanciate allora anche le due sequenze **{** *S* **}** *T* e **[** *S* **]** *T* sono bilanciate. Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza (non vuota) di parentesi quadre e graffe, termini la sua esecuzione lasciando sul nastro la *sola* sequenza SI se la sequenza iniziale è bilanciata e la *sola* sequenza NO altrimenti. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| []{[]} | SI |
| [{]} | NO |
| [[{}][]{}] | SI |

**Esercizio 3 [Schedina]**. Una colonna di schedina *S* è una sequenza simboli 1, 2 e X. Data la colonna vincente *V*, anch'essa costituita da una sequenza di altrettanti simboli scelti tra 1, 2 e X, si vuole verificare che *S* sia vincente, ovvero ci siano *almeno 12* risultati indovinati tra quelli riportati in *V*. Programmare una macchina di Turing che, dato un nastro iniziale contenente le sequenze *S* e *V* separate dal simbolo \*, termini la sua esecuzione lasciando sul nastro la *sola* sequenza SI se *S* è vincente e la *sola* sequenza NO altrimenti. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 1X1X2X21X1X12\*1X1X2X21X1X12 | SI |
| 1X1X2X21X1X12\*1X1X2X21X1X21 | NO |
| 1X1X2X21X1X12\*1X1X2X21X1112 | SI |

**Esercizio 4 [Divisione per due]**. Programmare una macchina di Turing che, dato un nastro iniziale contenente un numero pari decimale *N*, termini la sua esecuzione lasciando sul nastro il risultato della divisione di *N* per 2. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 1234 | 617 |
| 130 | 65 |
| 0 | 0 |

**Esercizio 5 [Raddoppio di sequenza]**. Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza arbitraria *S* di simboli A, B e C, termini la sua esecuzione lasciando sul nastro la sequenza *SS*, cioè la sequenza originale duplicata. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| ABACB | ABACBABACB |
| AB | ABAB |
| C | CC |

**Esercizio 6 [Divisibilità per sei]**. Programmare una macchina di Turing che, dato un nastro iniziale contenente un numero decimale *N*, termini la sua esecuzione lasciando sul nastro la *sola* sequenza SI se *N* è divisibile per 6 e la *sola* sequenza NO altrimenti. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 30 | SI |
| 16 | NO |
| 0 | SI |

**Esercizio 7 [Espressioni booleane]**. Si vogliono applicare ripetutamente le seguenti regole di sostituzione, dove la sequenza di due o tre simboli (in neretto) a sinistra di ogni freccia va sostituita con il simbolo corrispondente a destra della freccia:

* Sostituzioni NOT: **!0** -> 1, **!1** -> 0
* Sostituzioni AND: **\*00** -> 0, **\*01** -> 0, **\*10** -> 0, **\*11** -> 1
* Sostituzioni OR: **+00** -> 0, **+01** -> 1, **+10** -> 1, **+11** -> 1

Una sequenza *S* di simboli 0, 1, !, \* e + si dice *risolvibile* se applicando ripetutamente le sostituzioni suddette, in qualunque ordine, si ottiene alla fine un unico simbolo, chiamato *soluzione*, ovvero il simbolo 0 oppure il simbolo 1. Per esempio, se *S* è la sequenza +\*1+01\*0!\*01, si possono applicare le sostituzioni riportate sotto, ottenendo 1 come soluzione (si noti che, nel caso in cui più sostituzioni siano applicabili, l'ordine di applicazione non è rilevante):

+\*1**+01**\*0!\*01 -> +\*11\*0!\*01

+**\*11**\*0!\*01 -> +1\*0!\*01

+1\*0!**\*01** -> +1\*0!0

+1\*0**!0** -> +1\*01

+1**\*01** -> +10

**+10** -> 1

Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza *S* risolvibile, termini la sua esecuzione lasciando sul nastro la soluzione di *S*. Non importa come le sostituzioni vengano realizzate ed eseguite sulla macchina di Turing; è sufficiente che la soluzione finale calcolata (0 oppure 1) sia corretta. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 1 | 1 |
| \*1!\*1+01 | 0 |
| !!1 | 1 |

**Esercizio 8 [Somma]**. Programmare una macchina di Turing che, dato un nastro iniziale contenente due numeri decimali *N* e *M*, separati dal simbolo +, termini la sua esecuzione lasciando sul nastro la somma di *N* e *M*. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 30+85 | 115 |
| 23+0 | 23 |
| 0+0 | 0 |

**Esercizio 9 [Sequenza prefissa]**. Una sequenza di simboli A, B, e C si dice *prefissa* secondo la seguente definizione: (i) la sequenza composta di un solo simbolo (A, B oppure C) è prefissa; (ii) se *S* è una sequenza prefissa allora anche le sequenze *SS*A, *SS*B e *SS*C, costruite duplicando *S* e aggiungendo un simbolo in fondo, sono sequenze prefisse. Per esempio, A, AAA, AAC, AACAACC e AACAACCAACAACCA sono prefisse, mentre AA, ABA, AABA e ABAABAC non lo sono (ABAABAC non è prefissa perché ABA non è prefissa). Programmare una macchina di Turing che, dato un nastro iniziale contenente una sequenza di simboli A, B, e C, termini la sua esecuzione lasciando sul nastro la *sola* sequenza SI se la sequenza è prefissa e la *sola* sequenza NO altrimenti. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| B | SI |
| AB | NO |
| AABAABC | SI |

**Esercizio 10 [Crivello di Eratostene]**. Un intero *q* > 1 si dice *primo* se è divisibile solo per 1 e per se stesso. Per esempio, 2, 3, 5, 7, 11, 13, 17 e 19 sono primi. Dato un numero decimale *M*, si vogliono individuare tutti i numeri primi *q* <= *M* usando il seguente algoritmo, che rappresenta una versione semplificata del "crivello di Eratostene". Si marcano inizialmente come primi tutti i numeri da 2 a *M*. Sia *q* l'ultimo numero primo trovato (inizialmente *q* = 2). Si marcano come "non primi" tutti i numeri maggiori di *q* che sono multipli di *q*. Per esempio, se *q* = 2, si marcano 4, 6, 8, 10, 12, 14, ecc. Quindi, si pone *q* uguale al successivo numero che risulta marcato come primo, e si ripete la marcatura finché non ci sono ulteriori primi da esaminare. Usando il simbolo P per marcare un primo e il simbolo N per un "non primo", i numeri che rimangono marcati con P alla fine del crivello sono i numeri primi. Per esempio, per *M* = 20, il crivello esegue i seguenti passi:

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

N P P P P P P P P  P  P  P  P  P  P  P  P  P  P  P

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20      q = 2

N P P N P N P N P  N  P  N  P  N  P  N  P  N  P  N

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20       q = 3

N P P N P N P N N  N  P  N  P  N  N  N  P  N  P  N

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20       q = 5,7,11,13,17,19

N P P N P N P N N  N  P  N  P  N  N  N  P  N  P  N

Per *M* >= 2, sia data una sequenza formata da un simbolo N seguito da *M* - 1 simboli P, dove l'*i*-esimo simbolo (P o N) corrisponde al numero 1 <= *i* <= *M*. Programmare una macchina di Turing che, dato un nastro iniziale contenente la suddetta sequenza di *M* simboli NPPPPPPPPPPPPPPP..., esegua il crivello di Eratostene e termini l'esecuzione lasciando sul nastro la sequenza di *M* simboli NPPNPNPNNNPNPNNNPNPN... in cui ciascuna P corrisponde a un numero *primo* *q* <= *M*. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| NPPPPPPPPPPP | NPPNPNPNNNPN |
| NPPPPPPPPPPPPPPP | NPPNPNPNNNPNPNNN |
| NP | NP |

**Esercizi di Gara della VII edizione**

*AVVISI:*

         Se non specificato altrimenti negli esercizi, le sequenze iniziali su nastro si intendono *non vuote*, ovvero contenenti almeno un simbolo. La difficoltà degli esercizi è direttamente proporzionale al loro punteggio.

         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: Musica, musica! [Punti 1.5]** Tradizionalmente, le sette note della scala musicale vengono denominate in Italia do, re, mi, fa, sol, la, si. Le stesse note nel sistema anglosassone vengono indicate con le prime lettere dell'alfabeto: A, B, ... G, con do=C, re=D, … sol=G, la=A, si=B. Si scriva un programma che, data una stringa di note nel sistema anglosassone, lasci sul nastro le note equivalenti nel sistema italiano, separate da un punto. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| (Tanti auguri) CCDCFE | DO.DO.RE.DO.FA.MI |
| (Fra' Martino) CDECCDECEFGEFG | DO.RE.MI.DO.DO.RE.MI.DO.MI.FA.SOL.MI.FA.SOL |
| (Zerlina cede alle lusinghe di Don Giovanni -- Mozart)  BACBAC | SI.LA.DO.SI.LA.DO |

**Esercizio 2: Sottostringa [Punti 3].** Data una stringa s di caratteri di lunghezza n, e due interi i e j, con 1<=i<=j<=n, si vuole estrarre la sottostringa di s contenente i caratteri dall'i-esimo al j-esimo, estremi inclusi. Si scriva un programma che, dati in input i e j (in notazione decimale) e una stringa, tutti separati da un punto, lasci sul nastro solo la sottostringa richiesta. Si assuma che la stringa s possa contenere tutti i caratteri MAIUSCOLI nell'alfabeto A...Z. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 1.5.BANANA | BANAN |
| 7.16.SUPERCALIFRAGILISTIC | ALIFRAGILI |
| 4.4.CAROTA | O |

**Esercizio 3: La dama lineare I [punti 2].** Il gioco della dama lineare si gioca fra due giocatori, che chiameremo Bianco e Nero, su una "scacchiera” di 12 caselle disposte in linea. Ogni giocatore ha inizialmente 4 pedine del proprio colore, che indicheremo con B (per il Bianco) e N (per il Nero). Useremo invece il carattere "S" per le caselle vuote. All'inizio della partita, le quattro pedine di ogni giocatore occupano un estremo della scacchiera, e ci sono quattro caselle vuote fra i due giocatori:

BBBBSSSSNNNN

Durante il gioco, le pedine possono spostarsi e possono essere "mangiate", come si vedrà più avanti. Ad ogni momento durante una partita, la scacchiera dovrà contenere sempre 12 caselle: alcune saranno libere, altre occupate da pedine. Ogni giocatore avrà da 0 a 4 pedine del proprio colore (se ne ha 0, ha perso e la partita è finita), ma deve sempre essere presente almeno una pedina di uno dei due giocatori.

Si scriva un programma che, data una stringa di simboli sull'alfabeto {B, S, N}, lasci sul nastro "SI" se la stringa rappresenta una configurazione valida della scacchiera, come definito sopra, o "NO" altrimenti. Suggerimento: si noti che una configurazione è considerata valida in base al numero delle caselle e delle pedine presenti, ignorando le loro posizioni. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| BBBBSSSSNNNN | SI |
| BBBNSSSSSNNN | SI |
| BSNSN | NO (troppo corta) |
| BSSSSBBBSSNNN | NO (troppo lunga) |
| BBSBBBNNNNSS | NO (5 B) |
| NSSSSSSSSSSS | SI |
| BNBSSSSSSNNN | SI |
| SSSSSSSSSSSS | NO (ci vuole almeno una pedina) |

**Esercizio 4: La dama lineare II [Punti 1]**. Durante una partita a dama, può essere utile poter "ruotare" la scacchiera, in modo da guardare le pedine dalla prospettiva dell'avversario. Si scriva un programma che, data in ingresso una stringa rappresentante una scacchiera valida, la restituisca vista dal punto di vista dell'avversario. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| BBSBBSSNSNNN | NNNSNSSBBSBB |
| BSBBSSSSBSNN | NNSBSSSSBBSB |
| NNNSNSSBBSBB | BBSBBSSNSNNN |

**Esercizio 5: La dama lineare III [Punti 3.5]**. I giocatori muovono a turno, similmente a quanto avviene nel gioco della dama; inizia sempre il Bianco. Al proprio turno, un giocatore può muovere una propria pedina avanti di una casella, se la casella di destinazione è  libera: per esempio, la prima mossa di una partita è obbligatoriamente

BBBBSSSSNNNN -> BBBSBSSSNNNN

oppure può "mangiare" una pedina avversaria saltandola, se se la trova davanti e se la casella successiva è libera, come fa il bianco in questo esempio:

BBBSSBNSSNNN -> BBBSSSSBSNNN

In questo caso, se nella posizione di destinazione è ancora possibile mangiare una pedina avversaria, essa deve essere mangiata nella stessa mossa (doppio o triplo salto):

BBBSBNSNSNNN -> BBBSSSSSBNNN

Si scriva un programma che, data in input una configurazione della scacchiera, esegua una mossa valida del bianco, lasciando sul nastro la situazione della scacchiera dopo la mossa. Si assuma che nella scacchiera di ingresso ci sia sempre esattamente una mossa possibile per il bianco. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| BBBBSSSSNNNN | BBBSBSSSNNNN |
| SBNSNSSNSSSN | SSSSSBSNSSSN |
| SSSBSSSSSSNN | SSSSBSSSSSNN |
| BBSSBNNSSSSS | BSBSBNNSSSSS |

**Esercizio 6: Notazione polacca inversa [Punti 2]**. Nella notazione polacca inversa per le espressioni aritmetiche, gli operatori seguono gli operandi (sono postfissi anziché infissi). 1+2 si scrive 1 2 +, e (1+2)\*(3+1) è 1 2 + 3 1 + \*. Si scriva un programma per macchina di Turing capace di calcolare il valore di semplici espressioni in notazione polacca inversa, composte da numeri decimali (a una sola cifra) e dagli operatori P (per +) e M (per -). Si assuma che tutti gli operandi in ingresso, i risultati parziali, e il risultato finale siano interi fra 0 e 9. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 14P | 5 |
| 51M | 4 |
| 611PM | 4 |
| 12P3M22PP | 4 |
| 53P11PM1M | 5 |

**Esercizio 7: Cifrario di Cesare [Punti 1.5]**. Durante le sue guerre in Gallia, Giulio Cesare usava un semplice cifrario per comunicare con i suoi generali. In questo cifrario, ogni lettera era traslata di un numero fisso di posizioni. Nel cifrario originale, questo numero era 13: si aveva dunque

in chiaro                 ABCDEFGHIJKLMNOPQRSTUVWXYZ

cifrato                     NOPQRSTUVWXYZABCDEFGHIJKLM

Si scriva un programma per macchina di Turing che implementi un cifrario di Cesare generalizzato. Il programma prende in ingresso un numero decimale (da 1 a 25) seguito da una stringa su A-Z, e deve lasciare sul nastro la stessa stringa, cifrata traslando ogni lettera del numero indicato di posizioni. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 3ABC | DEF |
| 13BACO | ONPB |
| 1BACO | CBDP |

**Esercizio 8: Domani è un altro giorno [Punti 5]**. Si scriva un programma che, data una data sul nastro di ingresso nel formato gg/mm/aaaa, lasci sul nastro la data incrementata di un giorno. Si assuma che tutti gli anni divisibili per 4 (anche quelli secolari divisibili per 400) siano bisestili. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 01/02/2001 | 02/02/2001 |
| 31/10/2003 | 01/11/2003 |
| 28/02/2000 | 29/02/2000 |
| 28/02/2001 | 01/03/2001 |
| 31/12/2003 | 01/01/2004 |

**Esercizio 9: Scambio di coppia [Punti 1]**. Scrivere un programma per macchina di Turing che scambi a due a due i caratteri di una stringa formata da A, B e C, fornita sul nastro in ingresso. Se la stringa contiene un numero dispari di caratteri, l’ultimo carattere rimane inalterato. Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| AB | BA |
| ABCB | BABC |
| ABBCA | BACBA |
| A | A |
| BCCABA | CBACAB |
| CCCC | CCCC |

**Esercizio 10: Il conto della serva [Punti 1.5]**. Si programmi una macchina di Turing che, data in ingresso una cifra espressa in euro (intera), lasci sul nastro la stessa cifra espressa in lire, con l'assunzione che 1 euro valga 2000 lire, più mille lire (la cresta della serva). Esempi:

|  |  |
| --- | --- |
| *NASTRO INIZIALE* | *NASTRO FINALE* |
| 1 | 3000 |
| 2 | 5000 |
| 15 | 31000 |
| 540 | 1081000 |
| 0 | 1000 |