TestiGara - SedicesimaEdizione

Esercizi di Gara della XVI 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: Calendario AT. [Punti 3] Alan Turing nacque il 23 giugno 1912. Si consideri il 1912AD (Anno Domini) come anno 0AT (Anno Turingi), e si scriva un programma per Macchina di Turing che, ricevuto sul nastro un intero rappresentante un anno nella convenzionale notazione Gregoriana (fra il 1912 e il 9999), lasci sul nastro il corrispondente anno in notazione AT.

NASTRO INIZIALE	*NASTRO FINALE*
1912AD	0AT
1913AD	1AT
2012AD	100AT
2013AD	101AT
4120AD	2208AT

Esercizio 2: Compleanno di Turing. [Punti 2] Si scriva un programma per macchina di Turing che, ricevuta in ingresso una data nel tradizionale formato *gg*/*mm*/*aaaa* (con *gg* fra 01 e 31, *mm* fra 01 e 12, *aaaa* fra 1000 e 9999), lasci sul nastro la scritta HBAT (abbreviazione di Happy Birthday Alan Turing) se la data cade nel "compleanno" di Alan Turing, o NO altrimenti. Si considera "compleanno" anche una data precedente la nascita di AT.

NASTRO INIZIALE	*NASTRO FINALE*
23/06/1912	HBAT
22/06/1912	NO
23/11/1912	NO
23/06/1012	HBAT
23/06/2012	HBAT
05/12/2012	NO

Esercizio 3: Calcolo dell'età. [Punti 5] Si scriva un programma per macchina di Turing che, ricevuto in ingresso una data nel formato dell'esercizio precedente, successiva o uguale alla data di nascita di AT, lasci sul nastro l'età che AT ebbe o avrebbe avuto nella data indicata, secondo la consueta convenzione per

cui si incrementa l'età nel giorno stesso del compleanno.

NASTRO INIZIALE	*NASTRO FINALE*
23/06/1912	0
14/11/1922	10
14/02/2012	99
15/08/2012	100

Esercizio 4: La firma completa. [Punti 4] Il nome completo di AT era *Alan Mathison Turing*. Si scriva un programma per macchina di Turing che, ricevuta sul nostra una forma abbreviata o meno del nome di Turing, lasci sul nastro la forma espansa, in cui un segmento “A.” è sostituito da “Alan”, un segmento “M.” da “Mathison”, e un segmento “T.” da “Turing”. La stringa in ingresso può contenere lettere alfabetiche, spazi (al più uno consecutivo), punti, virgole. Tutti i caratteri non espansi devono essere copiati identici nel risultato

NASTRO INIZIALE	*NASTRO FINALE*
A. M. Turing	Alan Mathison Turing
Dr. A. Turing	Dr. Alan Turing
Turing, A. M.	Turing, Alan Mathison
Prof. Turing	Prof. Turing
A. T.	Alan Turing

Esercizio 5: Suona la campanella! [Punti 7] Nel 1926, a 14 anni, AT iniziò a frequentare le scuole superiori presso la Sherborne School, nel Dorset, a una certa distanza da Southampton, dove allora viveva. Disgraziatamente, il primo giorno di lezioni coincise con il grande sciopero generale del 1926 che paralizzò la Gran Bretagna. Il giovane AT era però particolarmente ostinato, per cui partì il giorno precedente per percorrere in bicicletta, da solo, i 97Km che separavano Southampton da Sherborne (fermandosi a dormire la notte in un ostello lungo la via), in modo da essere comunque presente al primo giorno di lezione. Si scriva un programma per macchina di Turing che, ricevuto sul nastro un intero rappresentante un numero di giorni di lezione nel periodo dello sciopero (che durò in tutto 9 giorni), lasci sul nastro un altro intero rappresentante il numero di chilometri da percorrere (97 all'andata, 97 al ritorno) per frequentare le lezioni. *Nota*: non risulta che AT abbia ripetuto l'exploit tutti i giorni!

NASTRO INIZIALE	*NASTRO FINALE*
1	194
3	582
9	1746

Esercizio 6: Battaglia fluviale. [Punti 11] Com'è ormai noto, durante la seconda guerra mondiale AT lavorò per il servizio segreto di Sua Maestà Britannica, curando in particolare la decodifica dei messaggi cifrati della marina tedesca. Possiamo immaginare che, nel tempo libero, si dilettasse nel gioco della *Battaglia fluviale.* La Battaglia fluviale segue le regole della più nota battaglia navale, ma il gioco si svolge su un fiume (quindi, su una sequenza di caselle) anziché in mare (quindi, su una griglia di caselle). In particolare, rappresenteremo con il carattere “X” una casella occupata da una nave, e con uno spazio una casella occupata da acqua; le caselle sono numerate da sinistra a destra, partendo da 1.

Si scriva un programma per macchina di Turing che, ricevuto sul nastro un intero *k* che rappresenta un numero di casella, seguito da un simbolo “:” e una rappresentazione del fiume (composta da “X” e spazi), di lunghezza non determinata, verifichi il contenuto della *k*-esima casella del fiume. Se questa era

occupata da una nave, il programma sovrascrive la casella con uno spazio, e lascia sul nastro una “C” (per “colpito!”) seguita da “:” e dalla rappresentazione del fiume. Se invece la casella era occupata da acqua, lascia sul nastro una “H” (per “H_2O”) seguita da “:” e dalla rappresentazione del fiume.

NASTRO INIZIALE (per chiarezza, indichiamo gli spazi con -)	*NASTRO FINALE (per chiarezza, indichiamo gli spazi con -)*
4:-XX--XXX	H:-XX--XXX
3:-XX--XXX	C:-X---XXX
2:-X---XXX	C:-----XXX
12:-XX--XXX------XX	H:-XX--XXX------XX
16:-XX--XXX------XX	C:-XX--XXX------X
8:	H:

Esercizio 7: La Battaglia dell'Atlantico. [Punti 18] La grande Battaglia Navale a cui AT diede un impulso risolutivo fu la Battaglia dell'Atlantico, combattuta fra gli U-Boot del grand'ammiraglio Karl Dönitz (i cosiddetti “branchi di lupi” dell'Atlantico) e i convogli di navi militari e trasporti organizzati dall'Ammiragliato Britannico.

La vera Battaglia dell'Atlantico vide coinvolti circa 1200 sommergibili tedeschi (e qualche italiano) contro molte migliaia di navi di superficie alleate. Per la nostra versione, però, ci accontenteremo di 8 unità, disposte (in maniera “segreta”) su una plancia di 8x8 caselle. Le unità sono composte da: 1 portarei (occupa 5 caselle, contenenti il codice “5”), 1 corazzata (occupa 4 caselle, contenenti il codice “4”), 1 cacciatorpediniere (occupa 3 caselle, contenenti il codice “3”), 5 sommergibili (occupano 1 casella, contenente il codice “1”). Le caselle sono identificate da un sistema di coordinate, in cui le righe sono denotate da una lettera (fra A e H) e le colonne da un numero (fra 1 e 8). Sul nastro, l'Atlantico è rappresentato mettendo di seguito le 64 celle della plancia, riga per riga. Negli esempi sottostanti, per chiarezza, andremo “a capo” ad ogni riga; sul nastro ovviamente non ci saranno “a capo”. Inoltre, rappresentiamo con uno sfondo colorato la casella indicata dalle coordinate.

Si scriva un programma per macchina di Turing che, ricevuto sul nastro l'indicazione di una casella, seguita da “:” e da una rappresentazione della plancia, verifichi il contenuto della casella indicata; se essa conteneva acqua, lasci sul nastro “H” seguito da “:” e dalla plancia; se invece si trattava di una nave, lasci sul nastro “A” (per “affondata”) se il colpo ha completamente eliminato l'unità colpita, oppure “C” (per colpita) se l'unità è stata colpita, ma non affondata; in entrambi i casi, nella casella indicata viene scritto uno spazio per indicare che ora contiene acqua, e la risposta è seguita dal codice del tipo di unità (1-5), da “:” e dal nuovo stato della plancia.

NASTRO INIZIALE	*NASTRO FINALE*
B4:	H: (acqua)
------4-	------4-
333---4-	333---4-
---1--4-	---1--4-
2-----4-	2-----4-
2-1-----	2-1-----
----1--1	----1--1

1-------
-55555--
D1:
------4-
333---4-
---1--4-
2-----4-
2-1-----
----1--1
1-------
-55555--
E1:
------4-
333---4-
---1--4-
------4-
2-1-----
----1--1
1-------
-55555--
C4:
------4-
333---4-
---1--4-
------4-
--1-----
----1--1
1-------
-55555--
H8:
------4-

1-------
-55555--
C2: (colpita unità da 2)
------4-
333---4-
---1--4-
------4-
2-1-----
----1--1
1-------
-55555--
A2: (affondata unità da 2)
------4-
333---4-
---1--4-
------4-
--1-----
----1--1
1-------
-55555--
A1: (affondata unità da 1)
------4-
333---4-
------4-
------4-
--1-----
----1--1
1-------
-55555--
H: (acqua)
------4-

333---4-	333---4-
------4-	------4-
------4-	------4-
--1-----	--1-----
----1--1	----1--1
1-------	1-------
-55555--	-55555--

Esercizio 8: La Bomba. [Punti 30] Uno dei contributi principali di Turing alla decodifica del codice tedesco Enigma consistette nel perfezionamento della *Bomba*, una macchina elettromeccanica (originariamente concepita dai servizi segreti polacchi, anch'essi dotati di eccellenti matematici). Il funzionamento della macchina era basato sulla disponibilità di un *crib* – ovvero, un frammento che si supponeva dovesse comparire nel testo decrittato. Un esempio famoso di *crib* era "*Keine besonderen Ereignisse*" (equivalente a "niente da segnalare"), che come si immagina facilmente, compariva spesso nei più vari report militari.

La macchina Enigma utilizzata sostanzialmente un *cifrario a sostituzione polialfabetico*, ovvero un sistema in cui ogni lettera dell'alfabeto viene sostituita da un'altra, e la tabella di sostituzione è diversa in punti diversi del messaggio. In questo sistema, la lettere "A" può essere sostituita da "L" in prima posizione, ma da "Q" in seconda posizione, da "R" in terza posizione, e così via – apparentemente, senza regolarità. In realtà, la regolarità era data dal particolare intreccio di circuiti elettrici che si realizzava combinando vari componenti meccanici (rotori, tratti di cavi, ecc.) all'interno della macchina.

Noi illustreremo il funzionamento della Bomba affidandoci a un più semplice *cifrario a sostituzione monoalfabetico* in cui la tabella di sostituzione è fissata e uguale in tutte le posizioni: e anzi, come ulteriore semplificazione, useremo un cifrario ROT*n* in cui ogni lettera viene sostituita da quella che si trova *n* posizioni più avanti nell'alfabeto (modulo 26). Il particolare caso del cifrario ROT3 è noto anche come *Cifrario di Cesare*, in quanto utilizzato da Caio Giulio Cesare (secondo la testimonianza di Svetonio) nelle guerre galliche. In ROT3, ogni A è sostituita da D, ogni B da E, e così via.

Si realizzi dunque un programma per macchina di Turing che, ricevuto sul nastro un *crib* (testo in chiaro) seguito da "?=" e da un messaggio cifrato in ROT*n* (in cui *n* è ignoto, ma maggiore di 0), lasci sul nastro il testo decodificato secondo il più piccolo *n* per cui il *crib* compaia, in posizione qualunque, nel testo decodificato. Il programma deve terminare scrivendo "?" se nessuna codifica ROT*n* è in grado di decodificare il messaggio in modo che emerga il *crib*. Sia il *crib* che il messaggio cifrato usano l'alfabeto inglese a-z a 26 lettere.

NASTRO INIZIALE	*NASTRO FINALE*
alba?=dwwdffduhdoodoed	attaccareall**alba** (ROT3)
port?=azcelpcptityiazcez	**port**aereixinx**port**o (ROT11)
port?=pnzovnerlpvsenevb	?
cifra?=pnzovnerlpvsenevb	cambiarey**cifra**rio (ROT13)
mai?=xfytktzytpdlfctep	ucvqhqwvq**mai**czqbm (ROT3 – ko!)
oni?=xfytktzytpdlfctep	munizi**oni**esaurite (ROT11 – ok!)
arma?=qqwzvxcibaedfasfwerflkejx	?

Esercizio 9: Il problema della fermata. [Punti 26] Uno dei primi contributi teorici di Turing fu legato al *Problema della fermata* (problema di cui forse già si era occupato attendendo l'autobus nei giorni del grande sciopero generale di cui all'esercizio 5): esiste una procedura che, dato un generico programma e il suo input, determina se il programma terminerà la sua computazione, oppure se continuerà a "girare" per sempre? Turing dimostrò che il problema della fermata è indecidibile: non può esistere una tale procedura.

Tuttavia, se assumiamo un particolare meccanismo per eseguire un programma (per esempio: una macchina di Turing) e poniamo un limite *k* al numero di passi che esso può compiere, possiamo rispondere facilmente alla domanda correlata: il programma dato termina entro *k* passi? La procedura è in questo caso semplice: basta eseguire passo-passo la macchina (simulandola), tenere il conteggio del numero di passi, e fermarsi se il programma termina (in questo caso, la risposta è SI), oppure se si raggiunge il numero di passi fissato *k* (in questo caso, la risposta è NO). Notate che se un programma non è terminato entro *k* passi, potrebbe sempre terminare al passo *k*+1, se solo l'avessimo lasciato andare avanti ancora un po'... quindi, la non-terminazione entro *k* non dice nulla sulla non-terminazione *tout court*.

Sul simulatore troverete pre-caricato con il nome *Problema 9* un interprete per una macchina di Turing, esso stesso scritto come macchina di Turing: si tratta della cosiddetta *Macchina di Turing universale*. In questa macchina (semplificata), il programma è dato da quintuple scritte sul nastro nel formato <*stato*><*input*><*stato*><*output*><*dir*>. Gli <*stati*> sono indicati da sequenze di S (S è lo stato 1, SS è lo stato 2, e così via); l'<*input*> e l'<*output*> sono sull'alfabeto {0, 1, N} (in cui N rappresenta il blank); lo spostamento della testina <*dir*> è codificato con 0 (= sposta a sinistra), 1 (= sposta a destra), N (= stai fermo). Si può dimostrare che, anche con queste limitazioni, la macchina di Turing è in grado di calcolare qualunque cosa che la macchina senza limitazioni può calcolare: è una versione più complicata da programmare, ma non meno potente.

Per esempio, la tupla (sss, 0, s, 1, >) viene codificata in questo sistema con SSS0S11, mentre la tupla (sss, -, ss, 0, -) diventa SSSNSS0N. La macchina di Turing universale pre-caricata si attende sul nastro il programma e l'input, nel formato: B<*tupla*><*tupla*>...<*tupla*>IT<*input*> in cui B indica l'inizio del programma, I la sua fine, e la T rappresenta la posizione della testina (che "punta" al simbolo che la segue: quindi, all'inizio, è posizionata sul primo simbolo dell'input). Un esempio di input completo per la nostra MdT universale è dunque: BS0S11S1S01IT0100010100101 (per i curiosi: si tratta del programma che scambia "0" e "1" nell'input, con la stringa di input 0100010100101).

I commenti nel codice pre-caricato forniscono qualche indicazione su come la MdT universale funzioni: la si modifichi in modo che essa accetti un intero *k* prima del simbolo B iniziale del programma, che durante l'esecuzione del programma dato in input conti i passi della macchina simulata e che si arresti col messaggio "NONT" (per "non termina") se la macchina non ha ancora terminato l'esecuzione dopo *k* passi, oppure con "TERM" se la macchina termina l'esecuzione in un numero di passi minore o uguale a *k*.

NASTRO INIZIALE	*NASTRO FINALE*
1000BS0S11S1S01IT0100010100101	TERM
5BS0S11S1S01IT0100010100101	NONT
20BSNSS11SSNSSS01SSSNSSSS0NITN	TERM
2000BSNSS11SSNSSS01SSSNSSSS0NITN	TERM
2BSNSS11SSNSSS01SSSNSSSS0NITN	NONT
5000BS0SS0S1S1NIT0	TERM
5000BS0SS0S1S1NIT1	NONT

Esercizio 10: Filotassi Fibonacciana. [Punti 17] Come molti altri personaggi dotati di menti inquiete, AT amava occuparsi dei problemi più disparati. Oltre ai suoi lavori fondamentali per l'Informatica, ha lasciato importanti contributi in chimica e in genetica, quando ancora questa disciplina aveva appena

iniziato la sua fioritura. A proposito di fioritura, AT si occupò del rapporto fra la *filotassi* (la parte della Botanica che studia la disposizione geometrica delle piante) e la *successione di Fibonacci*. In molte piante, la successione degli angoli o delle distanza fra petali, foglie, steli, segue infatti i rapporti dei termini consecutivi della successione di Fibonacci (il limite del rapporto fra due termini consecutivi è noto come *rapporto aureo*).

La successione di Fibonacci è definita classicamente come segue: F(0) = 0, F(1) = 1, e per $k>1$ F(k) = F(k-1)+F(k-2).

La *Felce di Turing* (*Fibopteridoma Alanturingi*) è una pianta infestante che cresce esclusivamente sui nastri delle macchine di Turing (pare si nutra di tuple, ma gli esperti non sono unanimi al riguardo). Il seme della pianta è costituito da un numero intero *j*. Innaffiando leggermente il nastro, la felce cresce in senso longitudinale, e nascono dei rami secondari, alternativamente a destra e a sinistra, distanziati fra di loro di una lunghezza di stelo pari all'*h*-esimo numero di Fibonacci. La felce smette di crescere quando *h* è pari al seme iniziale *j*. Sul nastro, il simbolo "-" indica una unità di stelo, il simbolo "/" un ramo a sinistra, il simbolo "\" un ramo a destra. La cima della felce corrisponde a F(0), e corrisponde sempre a un ramo "\". La base della felce è sempre un ramo (che può essere "/" o "\" a seconda dei casi).

Si scriva un programma per macchina di Turing che, ricevuto sul nastro di ingresso il seme *j*, lasci sul nastro di uscita la felce pienamente sviluppata.

NASTRO INIZIALE	*NASTRO FINALE*
3	\--/-\-/\
6	/--------\-----/---\--/-\-/\
1	\-/\
7	/----------/--------\-----/---\--/-\-/\
0	/\

Hyperboloids of wondrous Light
Rolling for aye through Space and Time
Harbour those Waves which somehow Might
Play out God's holy pantomime.