Bantumi 2 del 1/4

Consegnare corretto entro il 30/4 compreso

Questo esercizio riguarda il gioco del Bantumi. Si tratta di scrivere un programma capace di leggere da "input" 14 interi che costituiscono una configurazione iniziale del gioco e che quindi dovranno essere inseriti opportunamente nell'array B (vedi Bantumi 1). Dopo queste operazioni, il programma leggerà da "input" una sequenza di coppie costituite da 2 interi p e b, entrambi maggiori o uguali a 0, dove p=0/1 specifica se si tratta di una mossa del giocatore 0 o del giocatore 1, mentre b (0<=b<=5) specifica quale buca del giocatore p viene scelta per la mossa. L'ultima coppia deve avere b=-1 a segnalare che il gioco va immediatamente interrotto.

Eseguire la mossa (p, b) significa:

i) prelevare i fagioli contenuti nella buca b del player p (vuotando quindi quella buca) e

ii) inserire uno dei fagioli prelevati in ciascuna buca che si incontra procedendo in senso anti-orario dalla buca successiva alla b fino all'esaurimento dei fagioli. Facendo attenzione che, in caso durante questo percorso si incontrasse la buca grande dell'avversario del player p (che sta muovendo), la si deve saltare senza inserirvi alcun fagiolo.

Nel seguito un tale percorso verrà chiamato una semina. Il gioco prevede questi 2 casi speciali:

iii) se una semina termina in nella buca grande del giocatore che ha mosso allora lo stesso giocatore può fare subito un'altra mossa;

iv) se una semina termina in una buca piccola vuota, allorache viene messo in questa buca, assieme a quelli che si trovano nella buca opposta a questa nel tavolo di gioco, vanno aggiunte alla buca grande del giocatore che ha mosso.

Per il momento non è richiesto di occuparsi del caso di vittoria di uno dei giocatori. Questo viene lasciato all'esercizio Bantumi 3.

Esempio: Supponiamo che "input" contenga i seguenti valori:

0 0 3 1 11 2 4 2 5 2 4 10 3 1
1 1
0 4
0 -1
Allora la situazione iniziale di gioco è la seguente:
player 0

4	2	11	1	3	0	0	
	2	5	2	4	10	3	1

player 1

dove il primo numero (4) a sinistra rappresenta il contenuto della buca grande del player 0, mentre il numero più a destra (1) è il contenuto della buca grande del player 1.

Eseguire la mossa (1,1) significa che la semina deve iniziare dalla buca 1 del player 1. La buca 1 è quella che contiene 5. Quindi la configurazione raggiunta dopo questa mossa sarà:

4	2	11	1	3	0	0	
	2	0	3	5	11	4	2

Dato che la semina termina nella buca grande del giocatore 1, questo giocatore avrebbe diritto di rigiocare una seconda volta, ma al momento questo fatto non va implementato, ma va eseguita semplicemente la mossa rappresentata dalla coppia successiva che nell'esempio è (0,4) che porterebbe alla seguente configurazione:

5	3	0	1	4	1	1	
	3	1	4	6	12	5	2

La semina termina nella buca 2 del player 0. Si osservi che la semina appena effettuata attraversa la buca grande del player 1 ma non ci inserisce fagioli.

La terza coppia di "input" è (0, -1) e quindi il gioco deve terminare. Se su "input", dopo la prima coppia (1,1) ci fosse di nuovo la coppia (1,1), ovviamente essa sarebbe inutile in quanto la buca 1 del giocatore 1 ora è vuota. In caso di mosse inutili come questa, il programma non deve fare nulla e passare alla prossima coppia.

Esercizio: Il programma da realizzare deve assumere la seguente precondizione:

PRE=("input" contiene 14 interi non negativi seguiti da coppie di interi non negativi che contiene anche una coppia con seconda componente uguale a -1)

Quindi il programma deve leggere i primi 14 valori in un array B che modella la situazione di partenza del gioco (nel modo richiesto in Bantumi 1). Dopo di che deve leggere una coppia (p,b) alla volta e, nel caso b=-1, dovrà subito interrompere l'esecuzione, mentre se b è diverso da -1, allora, deve eseguire la mossa (p,b) modificando di conseguenza la situazione del gioco (cioè l'array B). La postcondizione del programma è la seguente:

POST="output" contiene se seguenti cose:

i) la configurazione di gioco iniziale,

ii) per ogni coppia (p,b) letta da "input" e tale che b !=-1, i 2 interi p e b vanno scritti su "output" e, nel caso la buca b del giocatore p non sia vuota, "output" deve contenere la nuova configurazione raggiunta eseguendo la mossa (p,b). Se invece la buca b del giocatore p è vuota, la configurazione non cambia e quindi non va riscritta su "output";

iii) dopo ciascuna nuova configurazione va scritta anche la frase "il giocatore cambia" oppure "il giocatore non cambia" a seconda che la semina corrispondente alla mossa effettuata finisca nel caso (iii) oppure no. Va ricordato che in entrambi i casi il prossimo giocatore che deve muovere è deciso sempre dalla prossima coppia che viene letta da "input";

iv) "output" termina sempre con la stringa "fine".

Il programma deve usare una funzione semina che soddisfa la seguente specifica:

PRE=(B contiene 14 valori non negativi, player=0/1, buca in [0..5], fagioli>0. Chiamiamo old_B il valore di B all'invocazione della funzione)

bool semina(int B[][7], int player, int buca, int fagioli)

POST=(B è ottenuta da old_B eseguendo la mossa (player,buca) secondo le regole del gioco descritte prima. Restituisce true se e solo se alla fine della semina si verifica il caso (iii))

Esempio: Il contenuto di "output" richiesto per rappresentare la sequenza di mosse dell'esempio precedente è:

0 0 3 1 11 2 4 2 5 2 4 10 3 1
1 1
0 0 3 1 11 2 4 2 0 3 5 11 4 2

il giocatore non cambia
0 4
1 1 0 1 11 2 4 3 0 3 5 11 4 2
il giocatore cambia
fine

NOTA: si osservi che il programma deve funzionare per qualunque configurazione iniziale. Non è quindi ammesso fare ipotesi sul numero di fagioli che in questa configurazione potranno trovarsi nelle buche.

Correttezza: associate un invariante al ciclo del main. Dimostrate la correttezza della funzione semina rispetto alla PRE e POST indicate.