Programmazione I
Prova di Programmazione – 7 luglio 2010 – 2 ore 30 min

Partendo dal frammento di codice fornito, scrivere un programma di gestione del seguente gioco di carte, in cui il mazzo è costituito da due insiemi identici, di *N* carte ciascuno, con *N* definito a tempo di scrittura del programma. Su ciascuna carta di ciascun insieme è stampato uno dei numeri da 1 ad *N*. Lo scopo del giocatore è trovare tutte le coppie di carte con lo stesso numero, vedendo le carte capovolte, ossia una X per ogni carta, e scoprendo le carte due alla volta. Il programma deve fornire le seguenti funzionalità:

1. **inizia_partita** Chiede all'utente di inserire il valore di ciascuna delle carte che saranno stampate capovolte nel prossimo punto. Se per esempio *N* = 3, una possibile sequenza potrebbe essere 1 3 1 2 3 2, il che vuol dire che la carta di valore 1 è in posizione 1, quella di valore 3 in posizione 2, e così via. Tralasciare il controllo del fatto che l'utente inserisca ciascun valore esattamente due volte. Un'eventuale precedente partita in corso è persa.
2. **stampa_stato** Solo se vi è una partita in corso, stampa: 1) le carte ancora da trovare, una dopo l'altra lungo una riga, e capovolte; 2) il punteggio del giocatore, inizialmente uguale a 0. Per esempio, all'inizio della partita dell'esempio al punto 1, stamperebbe:
   `X X X X X X --- Punteggio: 0`
3. **confronta_coppia(x, y)** Ritorna vero solo se vi è un partita in corso e le carte nelle posizioni **x** ed **y** sono uguali. Ritornerebbe per esempio vero per le posizioni (4, 6) nell'esempio al punto 1.
4. **salva_partita** Se vi è una partita in corso, ne salva lo stato su di un file dal nome stabilito a tempo di scrittura del programma.
5. **carica_partita** Ricarica la partita salvata dal file. L'eventuale partita in corso è persa.
6. **gioca_coppia(x, y)** Se non vi è nessuna partita in corso non fa nulla; altrimenti stampa il valore delle carte nelle posizioni **x** ed **y**, e poi, se le carte sono uguali, le elimina da quelle ancora da trovare, incrementa di 1 il punteggio del giocatore, e dichiara la partita terminata se non vi sono più carte da trovare. Se invece le carte nelle posizioni **x** ed **y** sono diverse, decrementa di 1 il punteggio del giocatore. Partendo di nuovo dall'esempio al punto 1, se il giocatore scopre le posizioni (1, 2), il suo punteggio è decrementato di 1, mentre se scopre le posizioni (1, 3) le carte in tali posizioni sono eliminate ed il punteggio è incrementato di 1. La sequenza di carte si compatta dopo l'estrazione di due carte uguali. Quindi, se invocata dopo questo secondo caso, la **stampa_stato** stamperebbe la sequenza di carte 3 2 3 2 capovolte, ossia
   `X X X X --- Punteggio: 1`
   Le posizioni valide ora diventano quindi quelle da 1 a 4. Per trovare per esempio le carte di valore 2, si dovrebbe ora scoprire la coppia di posizioni (2, 4), e così via.

I parametri di ingresso delle funzionalità sono solo indicativi. Gestire opportunamente le situazioni di errore, tranne gli *overflow* dovuti a valori assoluti troppo elevati e l'inserimento di dati in formato errato da *stdin.*

REGOLE

- Si può utilizzare ogni genere di manuale o di altro materiale didattico
- Per superare la prova, il programma deve contenere l'implementazione delle funzionalità 1, 2 e 3, e deve essere perfettamente funzionante nelle parti 1 e 2. Il voto ottenuto in questo caso è 18.
- Ciascuna funzionalità DEVE essere implementata mediante almeno una funzione.
- Il voto massimo (almeno 30) si ottiene se
  a) il programma è perfettamente funzionante in ogni sua parte
  b) tutti i principi di ingegneria del codice visti nel corso sono stati applicati