Programmazione I / Informatica Generale
Prova di Programmazione - 27 Febbraio 2008

Partendo dal frammento di codice fornito, scrivere un gioco in cui 2 giocatori hanno una pedina ciascuno. Lanciando un dado che restituisce valori discreti uniformemente distribuiti tra 1 e 6, i giocatori muovono a turno la propria pedina su *M* caselle consecutive. Il numero *M* di caselle è prefissato. Vince il giocatore che raggiunge o supera per primo l'ultima casella. In particolare, realizzare le seguenti funzionalità:

1. *Inizio nuova partita*: la pedina di ciascun giocatore è posizionata nella prima casella, il giocatore di turno è il giocatore 1. Un'eventuale partita in corso viene persa.
2. *Stampa stato partita*: se c'è una partita in corso, stampa, per ciascun giocatore, la casella in cui il giocatore si trova; quindi stampa il prossimo giocatore di turno, o l'indicazione di partita finita se la partita è già finita. Se, ad esempio le pedine si trovano nelle caselle 8 e 10 ed il giocatore di turno è il secondo, si stampa: `Giocatore 1: 8, Giocatore 2: 10`
   `Turno: Giocatore 2`
3. *Prossimo turno*: se la partita è già finita non fa nulla; altrimenti il giocatore di turno lancia il dado ed avanza di un numero di caselle corrispondente al valore restituito dal dado. Se il giocatore raggiunge o supera l'ultima casella vince, e la partita finisce. Altrimenti il turno passa al prossimo giocatore.
4. *Salva stato partita*: salva in un file di testo (dal nome definito a tempo di scrittura del programma) lo stato della partita.
5. *Carica stato partita*: carica da un file di testo (dal nome definito a tempo di scrittura del programma) lo stato della partita. La partita riprende da dove era stata salvata, ed un'eventuale partita in corso è persa.

Estendere il gioco supponendo che alcune caselle causino un ulteriore avanzamento o arretramento:

6. Definire il tipo di ciascuna casella a tempo di scrittura del programma.
7. Aggiungere una funzionalità *stampa caselle*. Ad esempio, supponendo *M*=7, bisogna stampare `[0, 2, -1, -3, 1, -4, 0]`, per indicare una sequenza costituita da caselle normali (0), che fanno avanzare (2, 1) e che fanno arretrare (-1, -3,-4).
8. Estendere la funzionalità *prossimo turno* affinché una pedina che finisce su una casella di avanzamento/arretramento sia ulteriormente spostata in avanti/indietro del numero di caselle indicate. Anche se questo facesse finire di nuovo la pedina su una casella di avanzamento/arretramento, non si effettuano comunque altri spostamenti fino al prossimo turno.

Gestire opportunamente le situazioni di errore.

REGOLE

- Per superare la prova, il programma deve essere perfettamente funzionante nelle parti 1, 2, 3, 4. Il voto ottenuto in questo caso è 18 (per superare poi l'intero esame bisognerà raggiungere 18 come voto finale).
- Ciascuna delle funzionalità DEVE essere implementata mediante una funzione dedicata.
- Si possono fare ritornare anche informazioni superflue per un dato punto dalla funzione o dalle funzioni ad esso relative, se questo semplifica la soluzione dei punti successivi.
- Il voto pieno (30) si ottiene se
  a) il programma è perfettamente funzionante in tutte le sue parti
  b) lo scopo di ogni funzione ed i suoi valori di ingresso ed uscita sono descritti brevemente mediante commenti
  c) lo scopo di eventuali punti complicati del codice è commentato (molto brevemente)