Programmazione I
Prova di Programmazione – 19 gennaio 2016 – 2 ore 15 min

Partendo dal frammento di codice fornito, realizzare un gioco il cui obiettivo è ottenere il punteggio massimo muovendo una pedina *a salti in avanti* su una sequenza di caselle. Ciascuna casella contiene un numero intero positivo, tranne la prima, che contiene il valore 0. Un esempio di sequenza potrebbe essere:

```
0 1 3 4 2 1 2 3
```

All'inizio del gioco , la pedina è posizionata nella casella `0`, e può essere spostata (solo) *a salti in avanti*, ossia saltando in avanti di una o più caselle. Il punteggio all'inizio è 0, ed ogni volta che si salta su una nuova casella, il punteggio incrementa del valore contenuto nella casella. Però tale valore vincola anche l'estensione del prossimo salto: si devono saltare un numero di caselle almeno pari a tale valore. Il gioco finisce quando il prossimo salto porta sopra, oppure oltre, l'ultima casella. Nell'esempio precedente, la serie di caselle da visitare per ottenere il punteggio massimo è: `1 4 3`.

All'avvio del programma la sequenza di caselle è vuota. Realizzare le seguenti funzionalità.

1. **inizia_partita()** Genera una sequenza di caselle di lunghezza casuale compresa tra 5 e 50, e con valori casuali, nelle caselle, compresi tra 1 e 5 (tranne ovviamente la prima che contiene il valore 0). Inoltre posiziona la pedina all'inizio delle sequenza ed azzera il punteggio. L'eventuale precedente contenuto della sequenza, nonché l'eventuale precedente punteggio, sono persi. Realizzare la funzionalità in modo tale da occupare la memoria minima possibile per la sequenza di caselle.
2. **stampa_stato()** Stampa lo stato del gioco, ossia: la sequenza di caselle, l'indicazione della posizione della pedina, ed il punteggio. Il formato di stampa è illustrato col seguente esempio, relativo all'inizio del gioco per il precedente esempio di sequenza di caselle (la pedina è sulla prima casella):

   ```
   0 1 3 4 2 1 2 3
   ^
   Punteggio: 0
   ```
3. **salva_stato()** Salva l'intero stato della partita in corso, su un file di testo dal nome definito a tempo di scrittura del programma.
4. **carica_stato()** Ricarica lo stato della partita dal file. L'eventuale stato precedente è perso
5. **sposta_pedina(n)** Fa saltare la pedina in avanti di un numero di caselle pari al massimo tra **n** ed il valore della casella su cui si trova attualmente la pedina. Inoltre incrementa il punteggio di un valore pari a quello della nuova casella su cui viene collocata la pedina, dopodiché stampa il nuovo stato del gioco. Se il salto fa anche finire la partita, allora comunica che la partita è finita, azzera il punteggio e colloca di nuovo la pedina sulla prima casella.
6. **punteggio_massimo()** Ritorna il punteggio massimo che si potrebbe ottenere se, per ogni casella visitata, tranne la prima, il numero di caselle da saltare fosse esattamente uguale al valore della casella visitata. Dalla prima casella rimarrebbe invece la libertà di saltare in avanti di qualsiasi numero di caselle. Per il precedente esempio, questa funzionalità ritornerebbe 7.

I parametri di ingresso delle funzionalità sono solo indicativi. Gestire opportunamente le situazioni di errore, tranne l'inserimento di dati in formato errato e di messaggi troppo lunghi da *stdin*.

REGOLE

- Si può utilizzare ogni genere di manuale o di materiale didattico di altra natura
- Per superare la prova, il programma 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