Programmazione I
Prova di Programmazione – 15 Dicembre 2009 – 2ore 15 min

Partendo dal frammento di codice fornito, scrivere un programma di simulazione di una regione contigua di celle di memoria, ciascuna contenente un valore intero. Tale regione, soggetta a corruzione casuale delle informazioni in fase di scrittura, sarà d'ora in poi chiamata semplicemente *regione*. Le celle della regione sono nello stato *scritto* se sono state interessate da almeno una scrittura, altrimenti sono nello stato *non scritto* (abbreviato *ns*). All'inizio la regione ha dimensione 0. Il programma deve fornire le seguenti funzionalità:

1. **reinizializza(n)** Reinizializzazione della regione. La regione assume dimensioni **n**, con tutte le celle nello stato *non scritto*. Il precedente contenuto della regione è perso.
2. **scrivi(indirizzo_iniz, sequenza_valori)** Scrive nella regione la sequenza di valori a partire dall'indirizzo iniziale passato. Ciascun valore deve essere compreso tra 1 e 20. La scrittura in ciascuna delle celle può essere corretta o corrotta. Si decide quale delle due possibilità si verifica invocando ed usando il valore di ritorno della funzione **da_corrompere** fornita. Se la scrittura è corrotta, il valore corrotto da memorizzare nella cella si ottiene invocando la funzione **valore_casuale** fornita e sommando al valore corretto il valore casuale ritornato da tale funzione. Ad esempio, se si scrive la sequenza [3, 6] a partire dalla seconda cella di una regione di 8 celle in cui si è ancora mai scritto prima, la regione potrebbe poi contenere i valori: **`ns, (5, 3), (6, 6), ns, ns, ns, ns, ns.`** Il primo valore di ciascuna coppia **`(x, y)`** è il valore effettivamente scritto, mentre il secondo è quello che andava scritto. Se una cella viene sovrascritta, il precedente contenuto è perso. Ad esempio, scrivendo ulteriormente la sequenza [7, 1, 5] a partire dalla terza cella nella precedente regione, si potrebbe ottenere: **`ns, (5, 3), (7, 7), (2, 1), (5, 5), ns, ns, ns.`**
3. Stampa del contenuto della regione e dei valori corretti nel formato degli ultimi esempi: le informazioni relative a ciascuna cella sono separate da virgole e terminate da un punto.
4. Salvataggio del contenuto della regione in un file di testo dal nome definito a tempo di scrittura del programma. Punteggio massimo se si salvano solo i dati delle celle in stato *scritto*.
5. Caricamento del contenuto della regione dal file di testo. Il precedente contenuto è perso.
6. **scrivi_con_somma(sequenza_valori)** Scrive la sequenza di valori a partire dalla prima cella della regione e forza nello stato *non scritto* le restanti celle. Ciascun valore della sequenza deve essere compreso tra 1 e 20. Oltre ai valori veri e propri della sequenza, si inserisce (scrive) ogni *M* valori un valore aggiuntivo contenente la somma di tali *M* valori. La scrittura in ciascuna cella è corrotta in modo casuale con le stesse identiche regole del punto 2. Se si fossero scritti con M=2 i valori [11, 5, 3] nella regione dell'ultimo esempio, si sarebbe potuto ottenere: **`(16, 11), (5, 5), (16, 16), (3, 3), ns, ns, ns, ns.`**
   Inserendo invece i valori [8, 2, 3, 6, 5], si sarebbe potuto ottenere: **`(8, 8), (3, 2), (10, 10), (7, 3), (6, 6), (11, 9), (5, 5), ns.`**

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 essere perfettamente funzionante almeno nelle parti 1, 2 e 3. 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