Programmazione I
Prova di programmazione – 13 Luglio 2023 – 2 ore

Partendo dal frammento di codice fornito, scrivere un programma di gestione di etichette contenenti codici a barre. Ogni codice è una sequenza di 8 coppie di posizioni consecutive. In ogni coppia di posizioni è contenuta una barra verticale. In particolare, la coppia di posizioni è costituita da una posizione sinistra ed una posizione destra, e la barra può essere presente a sinistra o a destra. Ecco un esempio di codice, in cui le posizioni senza barra sono rappresentate con un puntino:

```
0 1 2 3 4 5 6 7
.|.|..|.|.|.|.
```

nella prima coppia di posizioni (la coppia 0) la barra è presente a destra, nella seconda e terza coppia la barra è presente a sinistra, nella quarta è a destra, e così via. Il programma gestisce una etichetta alla volta, ed ogni etichetta contiene *N* codici, con *N* noto solo a tempo di esecuzione del programma.

All'avvio del programma l'etichetta non contiene alcun codice. Il programma fornisce le seguenti funzionalità.

1. **[+2, +2] inizializza_etichetta(N)** Inizializza l'etichetta a contenere **N** codici, leggendo i codici da standard input, ed assumendo che i codici siano rappresentati da un parola di 16 caratteri, come nell'esempio riportato nell'intestazione di questa traccia. L'eventuale precedente contenuto dell'etichetta è perso. Si ottiene il punteggio primo aggiuntivo massimo se si evitano deallocazioni e riallocazioni di memoria non necessarie. Per il collaudo, se **N**>0 allora le **N** stringhe vanno lette tutte per intero, anche se contengono caratteri scorretti, o se hanno lunghezza diversa da 8. Per ciascuna stringa, se la stringa rappresenta correttamente un codice, allora il corrispondente codice viene aggiornato a contenere il valore rappresentato dalla stringa. Altrimenti il codice viene inizializzato a contenere solo barre a sinistra. Si ottiene il secondo punteggio aggiuntivo se si memorizzano i codici occupando meno spazio possibile (assumiamo che il tipo **`char`** stia su 8 bit).

2. **stampa_etichetta** Stampa l'etichetta su *stdout*, con il formato mostrato nell'esempio seguente per una etichetta contenente due codici:

```
0 1 2 3 4 5 6 7       0 1 2 3 4 5 6 7
.|.|..|.|.|.|.       |.|.|..|.|.|.||.
```

3. **[2] salva_etichetta** Salva l'etichetta in un file **binario**.

4. **[2, +2] carica_etichetta** Carica l'etichetta dal file. L'eventuale precedente etichetta è persa. Si ottiene il punteggio aggiuntivo massimo se si evitano deallocazioni e riallocazioni di memoria non necessarie e si evita replicazione del codice.

5. **[3] inizializza_etichetta2(N)** Identica alla funzionalità 1, a parte il formato in cui vengono letti i codici. Per ciascun codice, non legge una sequenza di caratteri, ma legge il numero naturale corrispondente alla rappresentazione in base 2 di tale codice. In tale rappresentazione in base 2, la cifra binaria i-esima rappresenta la posizione i-esima del codice, e, in particolare, una barra in posizione, rispettivamente, a sinistra o a destra è rappresentata dal valore 0 o 1. Quindi, ad esempio, il valore 0 rappresenta il codice

```
|.|.|.|.|.|.|.|.
```

mentre il valore 1 rappresenta il codice

```
|.|.|.|.|.|.|..|
```

6. **[2] stampa_etichetta2** Data la rappresentazione in base 2 dei codici descritta nel precedente punto, stampa i numeri naturali corrispondenti a tale rappresentazione in base 2.

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

Per il collaudo: se fate stampare messaggi per invitare l'utente ad inserire dell'input, ricordate di aggiungere la stampa di caratteri accapo. Altrimenti nel puro output del programma vi saranno delle righe fuse, e di fatto tale output non sarà quello che credete (le righe fuse non le vedete quando usate il programma da terminale, perché inserite voi l'accapo da utenti).

REGOLE

- Si può utilizzare ogni genere di manuale e di materiale didattico
- Per superare la prova, bisogna svolgere almeno i punti 1 e 2. Se si svolgono solo tali punti, il programma deve essere perfettamente funzionante. Il voto ottenuto in questo caso è 18.

- Ciascuna funzionalità DEVE essere implementata mediante almeno una funzione.
- Il voto massimo 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
  c) sono state seguite eventuali altre indicazioni presenti nella traccia in merito al voto finale