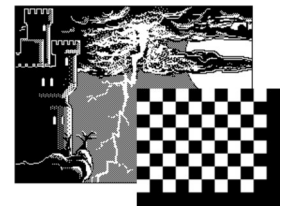


Nome: _____ Cognome: _____ Matr: _____ Postazione _____

Scrivere un programma in linguaggio C (chiamare il progetto con la propria <matricola>) che abbia il comportamento descritto nel seguito. Il tempo a disposizione è di **2 ore**. Al termine del tempo gli elaborati salvati su **U:** verranno raccolti automaticamente dal sistema. Eventuali documenti sono disponibili in **T:\Bertozzi**, usare **wordpad** per leggere i file di testo.

Una immagine digitale può essere vista come una matrice di punti. Nelle immagini in bianco e nero questi punti possono essere rappresentati con un singolo byte che assumerà i valori di 0 (nero) e 255 (bianco). Diversi sono i formati con cui una immagine può essere salvata su file.

Nel caso del **formato Compuserve RLE**, i file “binari” che contengono una immagine hanno 3 byte di intestazione che valgono: 27 (ESC), ‘G’ e a seguire ‘M’ per immagini con risoluzione 128×96 (larghezza × altezza in punti) o ‘H’ per immagini con risoluzione 256×192. I byte successivi, **decrementati di 32**, indicano, **alternativamente**, quanti punti trovo a seguire neri e bianchi partendo sempre con il nero. Ad esempio se il primo valore dopo la ‘M’/‘H’ vale 57 vuol dire che nella mia immagine, partendo dall’alto a sinistra la prima riga conterrà 57-32=25 punti neri di seguito. Il file termina con 3 byte che hanno come valori 27, ‘G’ e ‘N’.



Nel caso del **formato PGM**, i file binari che contengono i punti dell’immagine iniziano con una **intestazione ASCII** che contiene: nella prima riga “P5”, nella seconda riga le dimensioni dell’immagine (larghezza e altezza nell’ordine) separate da uno spazio, nella terza riga “255”. Queste tre righe terminano con un “a capo”. A seguire ci sono tutti i byte della matrice dell’immagine riga per riga. Si sviluppi un programma che:

1. (2) chiede all’utente il nome di un file di tipo RLE
2. (3-5) apre il file indicato in lettura, verifica che l’intestazione sia corretta per il formato RLE (altrimenti termina il programma) e ne calcola la risoluzione
3. (9-14) alloca dinamicamente opportuna struttura dati per memorizzare i punti dell’immagine RLE e li legge da file
4. (3-4) verifica che il file RLE termini correttamente (altrimenti termina il programma)
5. (2) chiede all’utente il nome di un file PGM
6. (4-6) apre il file indicato al punto precedente in scrittura e vi salva l’immagine letta al punto #3 in formato PGM

La definizione e l’uso di funzioni per i punti 2, 3, 4 e 6 permette di ottenere un punteggio aggiuntivo (i numeri tra parentesi indicano il punteggio massimo ottenibile senza e con l’uso di funzioni rispettivamente).

Suggerimenti:

- Funzioni di I/O viste per i file ASCII come fprintf() e altre possono essere comunque usate anche con file binari
- Sui PC di laboratorio, per visualizzare i file immagine potete usare Apri con → Infanview

Il codice va sviluppato nell’ordine indicato. Per ogni punto implementato si verifichi il corretto funzionamento. La correzione termina al primo punto non implementato correttamente.