Fondamenti di Informatica - Compito A

Prof. Marco Gavanelli

17 gennaio 2018

1 Esercizio (punti 16)

Le immagini con sfumature di grigio possono essere salvate in formato PGM. Nel formato PGM, le immagini sono rappresentate come file di testo. Un file PGM contiene, separati da spazi o "a capo":

- la stringa `"P2"`
- la larghezza W dell'immagine
- l'altezza H dell'immagine
- il numero S di sfumature (al massimo 255)

I dati successivi contengono dei numeri interi compresi fra 0 ed S; i numeri rappresentano l'intensità dei punti (o pixel):

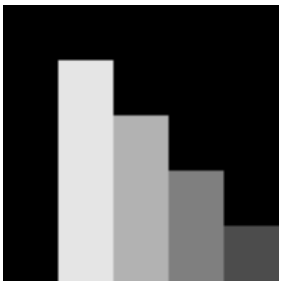
- 0 rappresenta il colore nero,
- S rappresenta il bianco
- e i numeri intermedi rappresentano le varie tonalità di grigio; numeri più alti sono associati a tonalità più chiare e numeri più bassi a tonalità più scure.

I primi W numeri rappresentano la prima riga dell'immagine, poi seguono ulteriori W numeri che rappresentano la seconda riga, e così via.

Ad esempio, il seguente file:

```
P2  5  5  9
 0  0  0  0  0
 0  9  0  0  0
 0  9  7  0  0
 0  9  7  5  0
 0  9  7  5  3
```

rappresenta l'immagine



Si desidera visualizzare sullo schermo l'immagine contenuta nel file `immag.pgm` utilizzando la cosiddetta ASCII-art, in cui si usa un carattere per visualizzare ciascun pixel dell'immagine. Si utilizzano solo due valori: un pixel con luminosità compresa fra 0 e $S/2$ viene visualizzato con un carattere spazio, mentre un pixel con luminosità superiore a $S/2$ è visualizzato con un carattere asterisco.

Nell'esempio riportato sopra, il programma dovrà visualizzare:

```
* ** ** **
```

Per semplicità, si può considerare, in questo esercizio, che l'immagine abbia dimensioni esattamente $W = 79$,$H = 66$, ovvero è necessario che il programma funzioni solo per immagini 79×66 (se il programma non funziona per immagini di dimensioni diverse da 79×66, va bene lo stesso).

Si organizzi il programma come segue

1. **main:** nel `main`, si invochi una funzione di lettura del file `immag.pgm` (da realizzare al punto 2), e una di visualizzazione (da realizzare al punto 3)

2. **lettura `immag.pgm`:** la funzione deve leggere il file `immag.pgm` e portarne il contenuto in una opportuna matrice di interi

3. **visualizzazione:** la funzione di visualizzazione ha come parametri

   - la matrice letta al punto 2
   - il numero di sfumature S che ci sono nell'immagine

   più, eventualmente, altri parametri e visualizza sullo schermo in forma di ASCII art la matrice letta al punto 2. Per fare questo, per ogni elemento della matrice

   - invoca una funzione di calcolo del carattere (da realizzare al punto 4), che fornisce il carattere da visualizzare
   - visualizza sullo schermo il carattere

4. **calcolo carattere:** la funzione riceve come parametri

- il valore di luminosità di un pixel dell'immagine (intero)
- il numero di sfumature S

più, eventualmente, altri parametri e fornisce il carattere spazio o il carattere asterisco a seconda se il valore di luminosità è superiore o inferiore a $S/2$.

È indispensabile organizzare opportunamente il programma in procedure e funzioni; è quindi altamente consigliabile aggiungere altre procedure/funzioni oltre a quelle indicate esplicitamente nel testo.

2 Esercizio (punti 4)

Per migliorare la visualizzazione dell'immagine, invece di avere due soli livelli di luminosità (visualizzati con i caratteri spazio e asterisco) si vogliono utilizzare più caratteri diversi: caratteri più "pieni" rappresenteranno sfumature più chiare e caratteri più "vuoti" rappresenteranno sfumature più scure.

Per sapere quale carattere visualizzare in corrispondenza di ciascun valore di luminosità, viene dato il file binario `scala.bin`; esso contiene una sequenza di dati (al massimo 255), in cui ciascun dato contiene:

- *inf*, *sup*: due interi che rappresentano i valori di luminosità per cui viene usato il carattere
- *car*: un valore di tipo `char` che rappresenta il carattere da visualizzare.

Qualora il valore di luminosità di un pixel sia compreso nell'intervallo da *inf* a *sup* (compresi), allora il carattere da visualizzare sarà *car*.

Ad esempio, se il file `scala.bin` contiene:

0	2	' '
3	5	'.'
6	8	':'
9	15	'*'

allora il programma, considerando l'immagine di esempio data sopra, dovrà visualizzare:

```
* *: *:. *:..
```

Si modifichi inoltre il programma in modo che funzioni per immagini che hanno dimensione qualunque, fino ad un massimo di 255 righe e 255 colonne.

È indispensabile aggiungere un numero significativo di funzioni rispetto all'esercizio 1, altrimenti verranno tolti dei punti.

È *indispensabile* organizzare il programma in opportune procedure e/o funzioni.

Qualora si svolga solo l'esercizio 1, si può svolgere tutto l'esercizio su un unico file.

Qualora si svolga anche l'esercizio 2, si consegnino i file:

- un file `COGNOME.c` (dove COGNOME va sostituito col cognome dello studente) che contiene il `main` e le funzioni usate solo nell'esercizio 1
- un file `facoltativo.c` che contiene il `main` e le funzioni usate solo nell'esercizio 2
- un file `funzioni.c` che contiene le funzioni comuni

più tutti i file header ritenuti necessari.

Nella correzione, il docente creerà due progetti:

- in uno, inserirà i file `COGNOME.c` e `funzioni.c`. L'eseguibile che viene creato dovrà risolvere l'esercizio 1.
- nell'altro, inserirà i file `facoltativo.c` e `funzioni.c`. L'eseguibile che viene creato dovrà risolvere l'esercizio 2.