Introduzione alla Programmazione (a.a. 2012/13)

Martedì 27 e giovedì 29 - Esercitazione 9 con valutazione - Parsing di file in formato vettoriale

Parsing di file in formato vettoriale

Questa è una esercitazione con valutazione e quindi deve essere svolta individualmente (le consegne uguali potranno essere penalizzate sul punteggio).

Quali sono gli argomenti principali di questa esercitazione:

- struct
- vector
- array bidimensionali
- realizzazione di codice modulare

Obiettivo dell'esercitazione

L'obiettivo dell'esercitazione è quello di scrivere un programma in grado di leggere e interpretare il contenuto di file in un semplice formato vettoriale e tradurlo in un formato raster (ossia in un'immagine digitale).

Il formato vettoriale

Il formato vettoriale che vogliamo gestire è vagamente riminiscente del formato SVG. SVG su wikipedia

Questo formato ci permette di descrivere in modo compatto forme geometriche piane e trasformazioni di forme. In questa esercitazione considereremo

1. **rettangoli pieni** descritti da un punto principale (il vertice in alto a sinistra), dall'altezza e dalla larghezza

2. **cerchi pieni** descritti da un punto di centro e da un raggio
3. **poligoni** vuoti, di N vertici (descritti da una lista di punti)

Un tipo di dato che sarà utile realizzare è il `point` (che conterrà le coordinate x e y di un punto)

Ogni forma può subire trasformazioni, noi considereremo le seguenti:

1. **move:** sposto la forma specificando lo spostamento delle righe e delle colonne
2. **rotate:** ruoto la forma intorno al vertice principale (il primo)
3. **scale:** scalo la forma rispetto al vertice principale (il primo)

Una forma è descritta dal suo tipo, i valori iniziali e (l'eventuale) lista delle trasformazioni che applichiamo; tutto questo viene inserito tra parentesi acute < >

La sintassi del nostro formato vettoriale è descritta nel seguente esempio

```
< rect 0 0 10 10 rotate 90 >
```

```
< circle 50 50 10 move 20 20 scale 0.5 >
```

la prima riga descrive un rettangolo che parte dal punto (0,0) di altezza e larghezza pari a 10 ruotato di 90 gradi intorno al suo centro

la seconda riga descrive un cerchio centrato sul punto (50 50) di raggio 10 che spostiamo di (20, 20) (ossia alla fine di questo spostamento il centro del cerchio sarà nel punto (70,70)) e riscaliamo di 0.5 (ossia alla fine della scalatura il cerchio avrà raggio 5).

In questo nostro formato vettoriale possiamo anche effettuare copie multiple (eventualmente trasformate) di una forma "base". Per fare questo dobbiamo prima di tutto assegnare un identificatore ad una forma:

```
< rect 0 0 10 10 rotate 90 id rettangolo1 >
```

dopo di che possiamo richiamare la forma associata ad un identificatore esistente nel seguente modo

```
< repeat rettangolo1 15 move 10 30 scale 0.8 >
```

in questo modo, per esempio, possiamo ripetere il rettangolo1 15 volte spostando ogni volta il rettangolo di (10,30) pixel e riscalandolo ogni volta di 0.8

Il formato raster

Le immagini digitali (come quelle viste nel laboratorio 7) realizzano la cosiddetta grafica raster. In un'immagine digitale una forma viene descritta in modo meno compatto, semplicemente andando ad assegnare un valore fissato (per esempio 255) a tutti gli elementi che appartengono alla forma stessa.

Per avere un'idea della relazione tra grafica vettoriale e grafica raster potete dare un'occhiata a questo tool online

Il programma

Il nostro programma dovrà essere in grado di produrre un'immagine digitale in formato PGM (si veda l'esercitazione 7) a partire da un file vettoriale. Per fare questo il programma dovrà:

- leggere il file vettoriale riga per riga, nell'ipotesi che ogni riga contenga un solo comando compreso tra parentesi acute < >.
- interpretare i simboli di una riga e produrre una descrizione astratta della forma
- tradurre questa forma in un'immagine digitale (raster), andando ad assegnare un valore fissato (per esempio 255) ad ogni elemento dell'immagine digitale che appartiene alla forma.

Importante:

- assumiamo che il file vettoriale sia sintatticamente corretto e non ci interessa se le forme generate vengono sovrapposte nell'immagine raster risultante.
- invece dobbiamo stare attenti che le forme non escano dall'immagine.

Organizzazione del programma

Il programma che realizzate deve essere strutturato in modo modulare e attenersi strettamente alle seguenti specifiche:

- le strutture dati e le funzioni atte a descrivere le forme e le loro trasformazioni saranno incluse in un modulo `shapes` (comprensivo di due file, shapes.h e shapes.cpp)
- le strutture dati e le funzioni relative alla produzione di un'immagine raster, all'inizializzazione della stessa e alla conversione di forme da formato vettoriale a raster saranno parte di un modulo `raster` (comprendivo di due file, raster.h e raster.cpp)
- le funzioni di parsing dell'input (delle righe del file) saranno contenute in un modulo `parse` (comprensivo di due file, parse.h e parse.cpp)

Ecco un file make che vi può aiutare nella compilazione in linea (copiate il file nella directory del codice e scrivete su terminale "make")

Ecco il file main.cpp che dovrete utilizzare e un file di test

Per la gestione dinamica di array bidimensionali potete usare il modulo matrix

Dettagli

Disegnare le forme sull'immagine

- Disegnare il rettangolo pieno e' facile
- Per disegnare il cerchio pieno e' sufficiente ricordare l'equazione del cerchio
- Per disegnare un poligono con N vertici dobbiamo riuscire a disegnare le rette passanti per due punti e ripetere l'operazione sulle N-1 coppie di vertici adiacenti.

Rotazione

- ruotare un cerchio intorno al suo centro è facile!

- per ruotare un rettangolo pieno senza fare troppa fatica fissiamo un solo angolo di rotazione possibile: 90 gradi
- per ruotare di un generico angolo theta un poligono con N vertici vuoto possiamo applicare le seguenti trasformazioni
  - applichiamo ad ogni punto (x,y) una rotazione secondo le seguenti formule x_new = cos(theta)*x- sin(theta)*y; y_new = sin(theta)*x+cos(theta)*y;
  - State attenti a fare sempre i cast a intero alla fine

Scalatura

- scalare un cerchio o un rettangolo è facile, basta aggiustare il raggio e altezza e larghezza
- scalare un poligono con N vertici è un pochino più complicato. Per ogni vertice tranne il primo dovete calcolare la distanza dal primo e posizionare il nuovo vertice dopo avere scalato questa distanza.

Identificatori

Quando associamo un identificatore ad una forma abbiamo in mente di riusarla in futuro. Occorre quindi memorizzarla da qualche parte, per esempio in un vector di forme dello stesso tipo.

Se il costrutto repeat fa riferimento ad un identificatore non esistente occorre ritornare un messaggio di errore e passare oltre.

Valutazione

Il programma che consegnate verrà compilato e testato su uno o più file di prova. Il codice che non compila potrebbe non venire corretto.

Il punteggio che otterrete sarà nel range [0,4].

- Per ottenere il punteggio massimo dovete completare tutta l'esercitazione, e produrre codice leggibile ben strutturato, ben commentato, che rispetti tutte le specifiche e sia in grado di gestire in modo opportuno tutti i casi indicati nel testo.
- Per ottenere un punteggio buono (2 o 3) dovete produrre codice leggibile ben strutturato, ben commentato, che rispetti le specifiche e sia in grado di gestire in modo opportuno i casi indicati nel testo. Se, per motivi di tempo, non riuscite a completare tutte le richieste, confinate la parte mancante in modo che il vostro lavoro sia completo nella parte che consegnate (per esempio potete lasciare da parte una specifica forma, ma quelle che considerate devono essere complete e funzionanti).

Stato consegna

Stato consegna	Consegnato per la valutazione
Stato valutazione	Non valutata
Termine consegne	lunedì, 10 dicembre 2012, 08:00
Tempo rimasto	Il compito è stato consegnato 6 ore in anticipo

Ultima modifica	lunedì, 10 dicembre 2012, 01:59
Consegna file	
Commenti alle consegne	☒ Commenti Commenti (0)

Modifica consegna

Modifica la tua consegna

NAVIGAZIONE

☐ Nascondi blocco Navigazione ☒ Sposta nel dock

Dibris

- Dashboard

Dibris

Corso in uso

IP-1213

Partecipanti

Badge

Introduzione

ESAMI

Prove d'esame degli anni precedenti

24 settembre - 30 settembre

1 ottobre - 7 ottobre

8 ottobre - 14 ottobre

15 ottobre - 21 ottobre

22 ottobre - 28 ottobre

29 ottobre - 4 novembre

5 novembre - 11 novembre

12 novembre - 18 novembre

19 novembre - 25 novembre

26 novembre - 2 dicembre

- **Martedì 27 e giovedì 29 - Esercitazione 9 con valu...**

3 dicembre - 9 dicembre

10 dicembre - 16 dicembre

17 dicembre - 23 dicembre

ESAMI

I miei corsi

Corsi

AMMINISTRAZIONE

Amministrazione del corso

Nascondi blocco Amministrazione Sposta nel dock

Referenti
Attivazione insegnamento

Policy

UNIVERSITÀ DEGLI STUDI DI GENOVA

Aul@web
Facebook
Youtube

Sei collegato come Leonardo Siri. (Esci)
IP-1213