POLITECNICO DI TORINO

e-Lite

Poli@Home

12BHD Informatica

Rudimenti di programmazione grafica in C con la libreria WinBGIm

Informatica (DELS-FEQ) – F. Corno – A.A. 2010/2011

Argomenti

- Librerie grafiche
- La libreria WinBGIm
- Software necessario
- Modello di funzionamento
- Principali funzioni
- Esempi

12BHD Informatica

Librerie grafiche

Programmazione grafica con libreria WinBGIm

Computer graphics, oggi

- Sistemi operativi basati su interfacce grafiche
  - Alte risoluzioni, full color
  - Applicazioni a più finestre
  - "Widget" standard (menu, toolbar, bottoni, campi testo, …)
- Campi applicativi
  - Applicazioni "office"
  - Applicazioni grafiche (immagini, filmati, …)
  - Video giochi
  - Programmi scientifici (matematica, fisica, ingegneria, …)
- Però… elevata complessità di programmazione

Complessità di programmazione

- Necessità di interagire con il sistema operativo attraverso opportune API
- Solitamente le API sono pensate per linguaggi object-oriented (C++, Java, C#, VB.Net, python, …)
- Programmazione di tipo asincrono ed event-driven
  - L'ordine di esecuzione delle operazioni dipende dalle azioni dell'utente e non è sotto il diretto controllo del programma
- Portabilità: API specifiche per un sistema operativo, o API "generiche" disponibili per più sistemi operativi
- Funzionalità della GPU: accelerazione, trasparenza, 3D, buffering, …

Librerie grafiche (API)

- A livello di Sistema operativo
  - Windows GDI, Unix X-11, MacOs Aqua
- Soluzioni accelerate non portabili
  - DirectX, Direct3D su Windows
- Librerie 2D/3D accelerate portabili
  - OpenGL e decine di toolkit basati su OpenGl
  - OpenCV
- Librerie per "applicazioni windowing"
  - Portabili: GTK, Qt, wxWindows, …
  - Windows: WPF (XAML+.NET)
- Librerie di più alto livello

…ma si può fare in C?

- Molte librerie espongono anche API per programmi scritti in C. Però:
  - Forte uso di puntatori e memoria dinamica
  - Funzioni spesso complesse e basate su strutture dati (struct, matrici, …) che richiedono un certo studio
  - Paradigma di programmazione event-driven, da studiare e capire
  - Hello world → 100-200 righe di programma
- …nulla che si possa imparare in ~un'ora
  - …a meno di tornare indietro di 20 anni!

12BHD Informatica

La libreria WinBGIm

Programmazione grafica con libreria WinBGIm

DOS e Borland

- Ai tempi di MS-DOS (prima di Windows 3.0)
- Non esistevano interfacce a finestre
  - Console
  - Grafica bitmapped
- La software house Borland vendeva favolosi compilatori
  - TurboPascal, TurboC/TurboC++
- Nei compilatori Borland era inclusa una libreria grafica
  - BGI: Borland Graphics Interface
  - Semplice da usare, anche se poco potente (per gli standard odierni)
  - Programmazione procedurale in C

BGI oggi

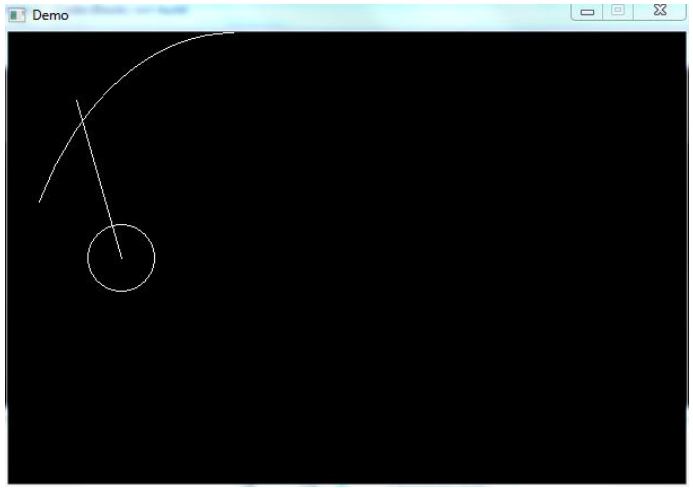
- La libreria BGI originale funziona:
  - Con compilatori a 16 bit
  - Con schede grafiche solo EGA/VGA/SVGA
  - Nel sistema operativo DOS (non Windows)
- Esiste un "porting" su windows
  - **WinBGIm**: Windows BGI con mouse
  - Compatibile con il compilatore mingw
  - Estende la "vecchia" BGI con funzioni più utili in ambiente windows (es. controllo del mouse)
  - Mantiene la stessa modalità di programmazione
  - Purtroppo non molto documentata e/o aggiornata
- Non funziona su Linux o Mac OS-X

Esempio programma winBGIm

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <graphics.h>
4
5  int main(int argc, char *argv[])
6  {
7      initwindow(600, 400, "Demo", 100, 50) ;
8
10     circle(100,200,30);
11     line (60, 60, 100, 200);
12     ellipse (200, 300, 90, 150, 200,300);
13
14     while( !kbhit() )
15         /*nop*/;
16
17     getch() ;
18
19     closegraph( );
20     exit(0) ;
21 }
```

Esempio programma winBGIm

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <graphics.h>
4
5  int main(int argc, char *argv[])
6  {
7      initwindow(600, 400, "Demo", 100, 50) ;
8
10     circle(100,200,30);
11     line (60, 60, 100, 200);
12     ellipse (200, 300, 90, 150, 200,300);
13
14     while( !kbhit() )
15         /*nop*/;
16
17     getch() ;
18
19     closegraph( );
20     exit(0) ;
21 }
```



Software necessario

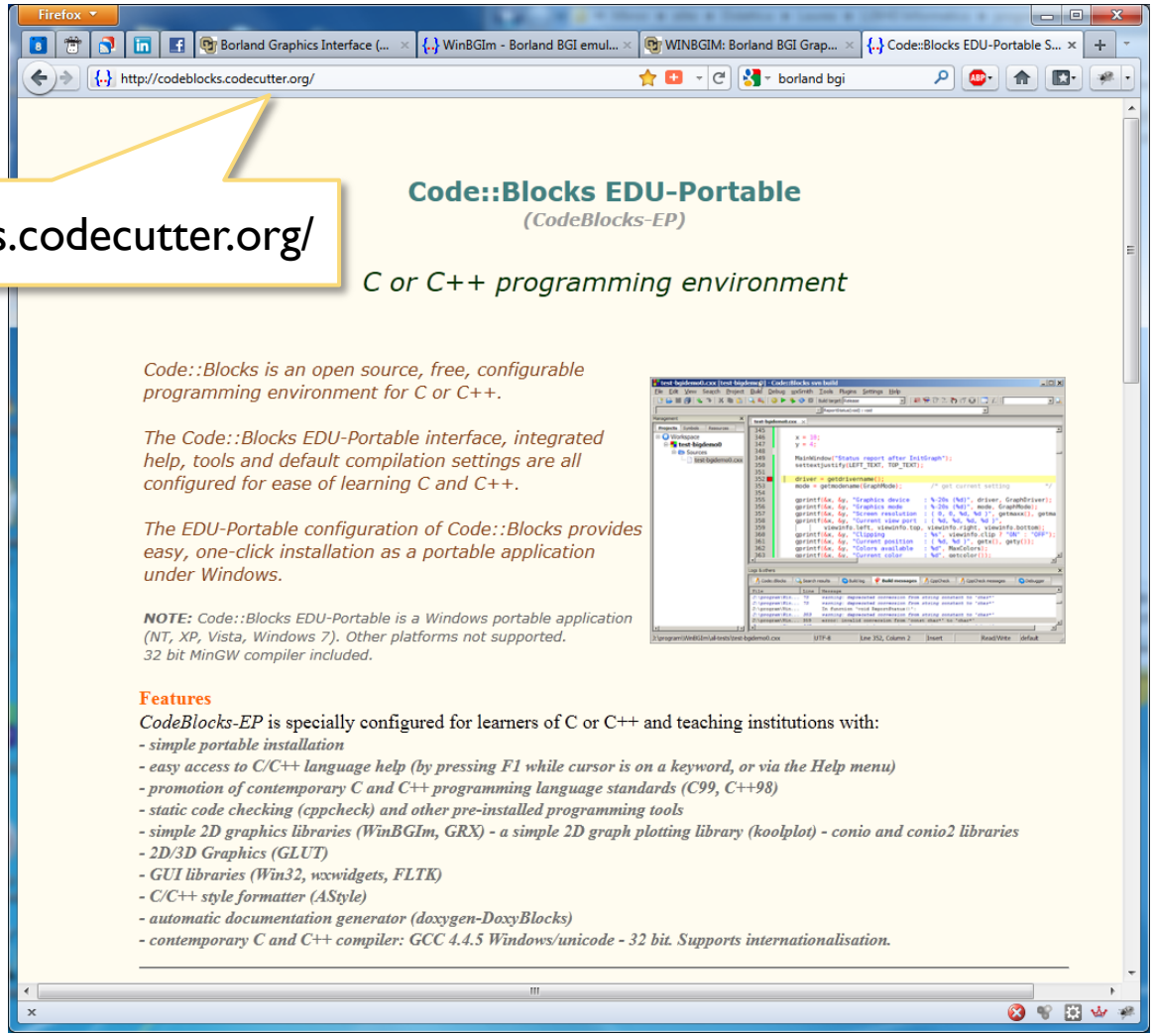
Programmazione grafica con libreria WinBGIm

Installazione WinBGIm

- Composta da 2 file
  - Libreria compilata: libbgi.a
  - File da includere: graphics.h
- Necessario:
  - Copiare tali file nelle directory di installazione di mingw (all'interno di codeblocks)
  - Impostare opzioni di compilazione particolari nel progetto
    - -x c++ -lbgi
- Oppure:
  - Scaricare una versione "modificata" di Code::Blocks che include già tali modifiche

Distribuzione modificata di Code::Blocks

http://codeblocks.codecutter.org/



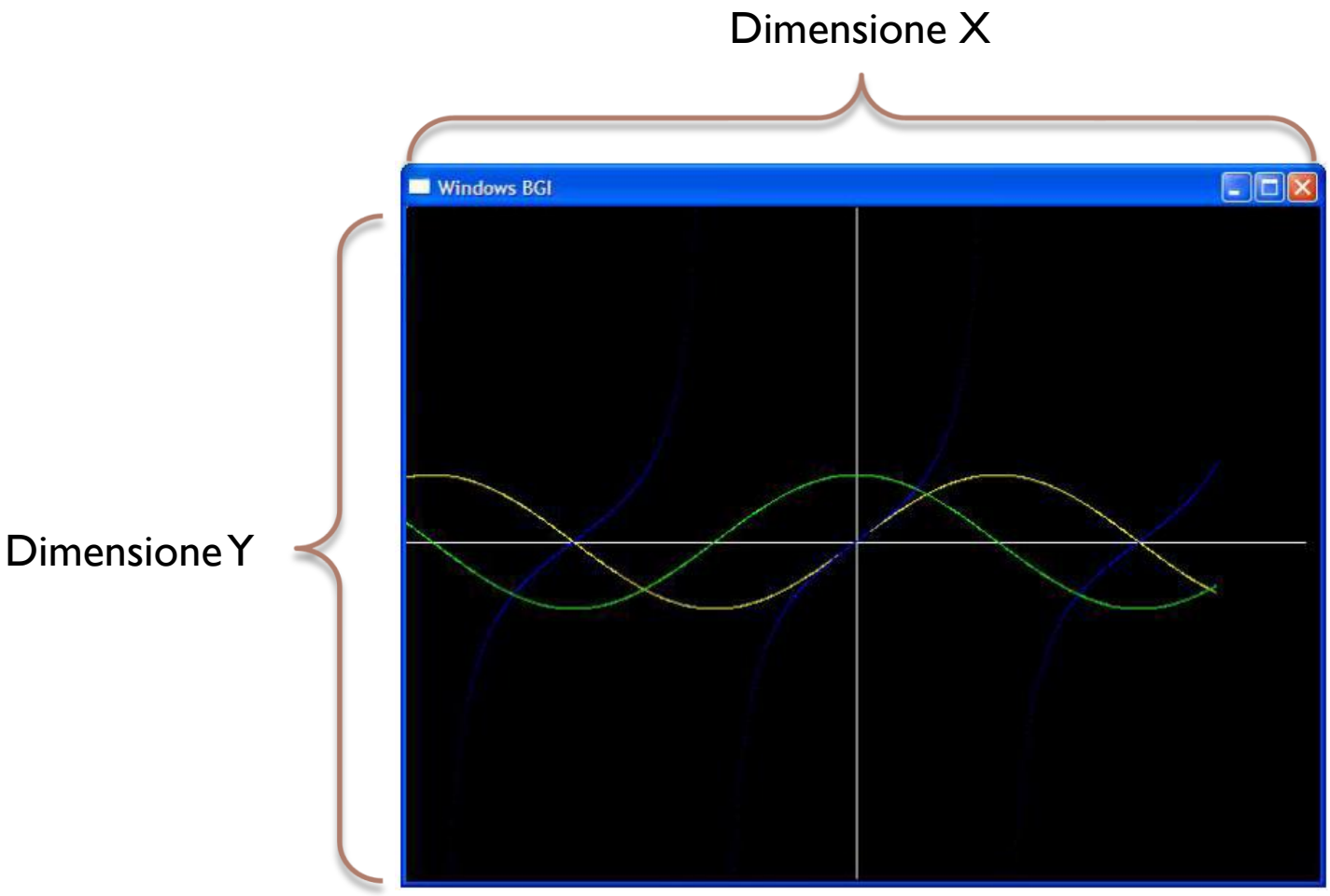
Modello di funzionamento

Programmazione grafica con libreria WinBGIm

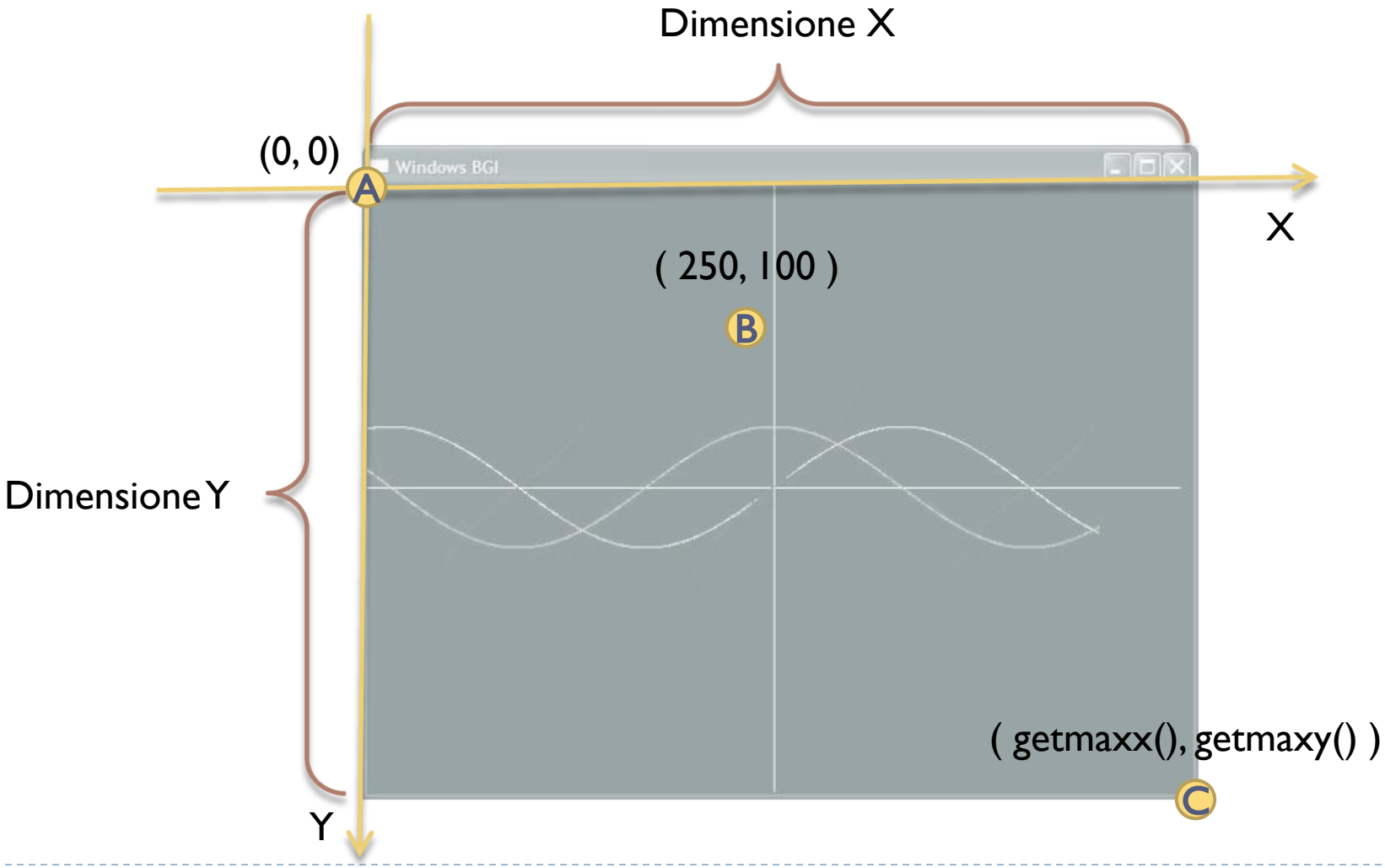
Modello di funzionamento

- Il programma può aprire una (o più) **finestre grafiche**
- Ogni finestra ha un **sistema di coordinate** in cui lavorare, corrispondente ai pixel dell'immagine
- All'interno delle finestre, è possibile **disegnare** con delle primitive grafiche (linee, punti, cerchi, scritte, …) in diversi **colori**
- È possibile leggere i caratteri della tastiera in modalità *non bloccante*

Finestra grafica



Finestra grafica



Creare una finestra

- La funzione initwindow(…) crea una nuova finestra grafica
  - Width, height: dimensioni della finestra in pixel
  - Title: titolo (opzionale)
  - Left, top: posizione iniziale (opzionale)
- Per chiudere la finestra usare closegraph()

- int initwindow(int width, int height, const char* title="...", int left=0, int top=0);
- void closegraph(void);

Colori

- 16 colori predefiniti
  - Costanti da 0 a 15:
  - BLACK, BLUE, GREEN, CYAN, RED, MAGENTA, BROWN, LIGHTGRAY, DARKGRAY, LIGHTBLUE, LIGHTGREEN, LIGHTCYAN, LIGHTRED, LIGHTMAGENTA, YELLOW WHITE
- Un qualsiasi colore RGB
  - COLOR(r, g, b) con r, g, b, compresi tra 0 e 255
  - COLOR(255,100,0)
- void setcolor(int color);
  - setcolor(BLUE); // Change drawing color to BLUE.
  - setcolor(COLOR(255,100, 0)); // Change drawing color to reddish-green.
- void setbkcolor(int color);

Principali funzioni

Programmazione grafica con libreria WinBGIm

Disegnare

- Punti
  - void putpixel(int x, int y, int color);
- Linee
  - void line(int x1, int y1, int x2, int y2);
    - Segmento da (x1, y1) a (x2, y2) con il colore corrente
  - void lineto(int x, int y);
    - Segmento dall'ultimo punto tracciato a (x, y)
  - void linerel(int dx, int dy);
    - Segmento dall'ultimo punto, spostandosi di un vettore (±dx, ±dy)

Disegnare

- Curve
  - void **circle**(int x, int y, int radius);
  - void **arc**(int x, int y, int stangle, int endangle, int radius);
    - /* angoli da 0 a 360 */
  - void pieslice(int x, int y, int stangle, int endangle, int radius);
    - /* arco ripieno */
  - void **ellipse**(int x, int y, int stangle, int endangle, int xradius, int yradius);
  - void fillellipse(int x, int y, int xradius, int yradius); /* pieno */
  - void sector(int x, int y, int stangle, int endangle, int xradius, int yradius);
    - /* arco di ellisse ripieno */

Disegnare

- Rettangoli
  - void rectangle(int left, int top, int right, int bottom);
  - void bar(int left, int top, int right, int bottom); /* riempita */
- Poligoni
  - void drawpoly(int numpoints, int *polypoints);
    - Polypoints è un vettore di 2N interi, in cui per ciascuno degli N punti sono riportare le coordinate X e Y
  - void fillpoly(int numpoints, int *polypoints); /* riempito */
- Riempimento
  - void floodfill(int x, int y, int border);
    - Riempie del colore corrente l'area intorno al punto (x,y), fermandosi ai bordi di colore border

Scrivere

- "Disegnano" il testo specificato
  - void outtextxy(int x, int y, char *textstring);
  - void outtext(char *textstring);
- Il carattere si può speficicare con:
  - void settextstyle(int font, int direction, int charsize);
    - font = 0…10 (font predefiniti dalla BGI)
    - direction = HORIZ_DIR
    - charsize = 1 o 2

Tastiera

- Leggi un carattere senza aspettare l'invio
  - int getch(void);
- L'utente ha premuto un tasto?
  - int kbhit(void);
  - Funzione non bloccante!
    - if(kbhit()) { ch = getch() ; move_user(ch) ;} else { move_monsters(); }

Mouse

- Coordinate correnti del mouse
  - int mousex(void);
  - int mousey(void);
- C'è stato un click?
  - bool ismouseclick(int kind);
- Dove è stato il click?
  - voud getmouseclick(int kind, int& x, int& y);
- Pulisci il buffer dei click precedenti
  - void clearmouseclick(int kind);
- Parametro kind: tipo di evento che vogliamo interrogare
  - WM_MOUSEMOVE
    WM_LBUTTONDOWN
    WM_LBUTTONUP
    WM_RBUTTONDOWN
    WM_RBUTTONUP …

Controllo del programma

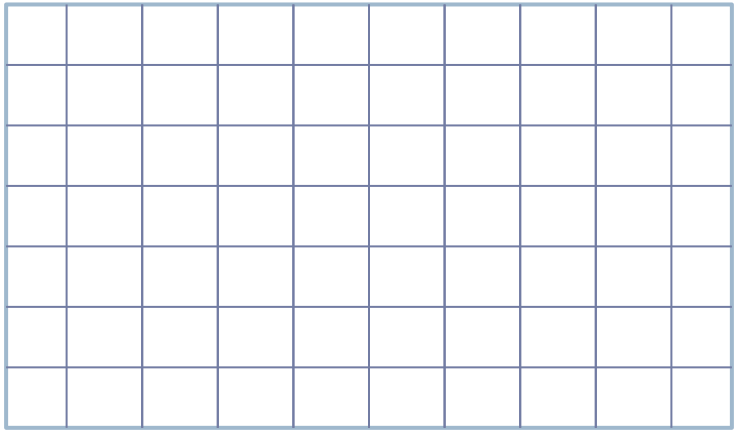
- Piccolo ritardo di elaborazione
  - void delay(int millisec);

Esempi

Programmazione grafica con libreria WinBGIm

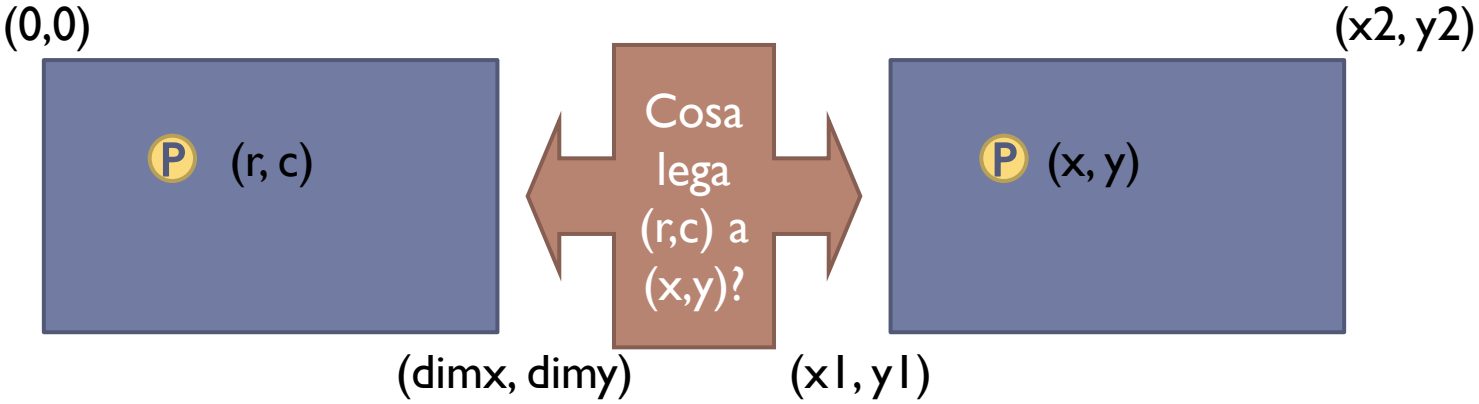
Esercizio

- Creare una finestra grafica 400x300 e disegnare una "griglia" di linee a distanza regolare di 20 pixel
- Opzionalmente, le linee dovranno avere diversi colori
- Opzionalmente, i singoli quadretti dovranno essere riempiti con un motivo a scacchiera



Esercizio

- Disegnare il grafico della funzione y=x^2 (con x ∈ [-2,2]) in una finestra di dimensione 400x400
- Opzionalmente, disegnare anche gli assi cartesiani
- Nota: occorre rimappare il sistema di coordinate della finestra nel sistema di coordinate del grafico



Esercizio

- Realizzare un gioco scemo nel quale l'utente può spostare un quadrato, spostandolo con i tasti freccia.
- I movimenti del quadrato dovranno essere vincolati all'interno della finestra
- Se l'utente preme H (hyperspace), il quadrato scompare e ricompare in un punto casuale.

Riferimenti e link

- Code::Blocks EDU-Portable
  - http://codeblocks.codecutter.org/
- Documentazione funzioni WinBGIm
  - Introduzione: http://www.cs.colorado.edu/~main/cs1300-old/cs1300/doc/bgi/bgi.html
  - Consultazione on-line: http://www.cs.colorado.edu/~main/cs1300/doc/bgi/index.html
  - In formato PDF: http://www.stefansundin.se/programmering/proga/WINBGIM-%20MANUAL.pdf
- Dispensa sull'uso delle funzioni BGI
  - http://web.tiscali.it/cervelli/dispense/Programmazione%20Grafica%20V%200.3.pdf

Licenza d'uso

- Queste diapositive sono distribuite con licenza Creative Commons "Attribuzione - Non commerciale - Condividi allo stesso modo 2.5 Italia (CC BY-NC-SA 2.5)"
- Sei libero:
  - di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera
  - di modificare quest'opera
- Alle seguenti condizioni:
  - **Attribuzione** — Devi attribuire la paternità dell'opera agli autori originali e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.
  - **Non commerciale** — Non puoi usare quest'opera per fini commerciali.
  - **Condividi allo stesso modo** — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.
- http://creativecommons.org/licenses/by-nc-sa/2.5/it/