Tetris

Relazione del progetto per l'insegnamento di Programmazione

Alberto Zuccari - 1029107

Università di Bologna
8 febbraio 2024

1 Funzionalità

Tetris è un videogioco di logica e ragionamento russo inventato da Aleksej Leonidovič Pažitnov nel 1984, mentre lavorava al centro di calcolo dell'Accademia delle scienze dell'URSS di Mosca.

1.1 Programmazione

Il programma, scritto usando `C++11` e fa grande uso delle classi per lavorare.
Le specifiche del progetto prevedono serie limitazioni sull'uso della *Standard Template Library*. I contenitori come `vector`, `list` e `map` e `string`, per esempio, sono vietati. Per far fronte a tali sono stati utilizzati gli array di caratteri al posto delle string.

2 Il menù

Alla chiamata della funzione `menu()` viene visualizzato il menù, questo semplicemente renderizzando uno schermo una finestra e rimanendo in un ciclo `while` chiamato dal metodo `scelte` (implementato per evitare la duplicazione di codice uguale) sempre vero finché non si preme il tasto `enter` che selezionerà anche cosa andare a vedere, se far partire il gioco oppure renderizzare la classifica.

3 Game

il metodo Game è quello chiamato per far partire il gioco ed è il pilastro portante del programma.
Come prima cosa controlla se il file della `classifica.txt` segue determinate regole per evitare che il costruttore della classe `Classifica` riporti un errore di `segmentation fault`. Viene poi rimossa l'ultima riga del file `classifica.txt` riservato per il salvataggio del nome e punteggio del giocatore.
Vengono poi caricate le finestre di gioco, quella delle informazioni e quella con il prossimo blocco. Poi viene creato l'oggetto game di tipo Game e viene richiesto all'utente il nome.
Il nome è limitato ad una dimensione di 3 caratteri per riprendere lo stile arcade in cui si limitavano i caratteri a 3 per questioni di memoria.

Dopo l'inizializzazione di alcune variabili necessarie per il corretto funzionamento del gioco parte il loop di gioco.
Durante tutte le iterazioni del loop di gioco vengono ricontrollate e ristampate a schermo le seguenti informazioni:

- se lo stato del gioco è in GameOver
- stampa della classifica nella barra a sinistra
- disegno del prossimo blocco nella barra a sinistra
- seguendo un timer di 200ms viene disegnato e fatto cadere il blocco corrente
- infine viene gestito il caso di GameOver
  - viene ordinata la classifica in ordine decrescente
  - viene salvato il punteggio del giocatore sul file `classifica.txt`
  - aspetta 1 secondo e poi torna al menù

3.1 La classe Game

La classe game è il cuore del gioco, essa sfrutta le classi `Grid` , `Classifica` e `Block` per dare la possibilità di creare un unico elemento game per ogni partita.
La classe è divista tra `private` e `public` in questo modo:

- `public:`
  - `Game();`
  - `Block getRandomBlock();`
  - `Grid grid;`
  - `Classifica classifica;`
  - `void getAllBlocks(int[], int);`
  - `void draw(WINDOW * win);`
  - `void handleInput(WINDOW * win);`
  - `void MoveBlockLeft();`
  - `void MoveBlockRight();`
  - `void MoveBlockDown();`
  - `bool isGameOver();`
  - `bool gameOver;`
  - `Block nextBlock;`
- `private:`
  - `int blocks[7];`
  - `Block currentBlock;`
  - `bool isBlockOutside();`
  - `void rotateBlock();`

- `void lockBlock();`
- `bool blockFits();`

Particolarmente rilevanti sono le classi di movimento del blocco e della gestione delle collisioni, per il movimento viene controllato sempre se non si è in gameOver e in caso affermativo si sposta il blocco delle coordinate rispettive $(\pm y, \pm x)$ attrraverso il metodo `Block::move()` della classe Block.
Subito dopo il movimento si controlla se il il movimento è invalido e se non si può spostare il blocco in quella direzione si reimpostano le coordinate allo stato iniziale prima della chiamata dell metodo di movimento.

Per le collisioni si utilizza il metodo `isBlockOutside` che crea una variabile di tipo `position` ovvero un vettore di 8 coordinate. Questo array verrà poi controllato tutto verificando se almeno una delle coordinate non rientri nei limiti della grid. Ciò sfruttando il metodo `isCellOutside` della classe `Grid`

3.2 La classe Grid

Viene utilizzata per rappresentare un tabellone di gioco basato su griglia, qua abbiamo una matrice `grid` di dimensione 20 x 20 che contiene dei numeri che vanno dall' 1 all'8 per disporre i colori dei blocchi.
Contiene anche i metodi che controllano se una riga viene riempita e nel caso la spostano in basso, e la gestione del punteggio, che aumenta ogni volta che si completano delle righe.

3.3 La classe Block

È usata per gestire: movimento e rotazione dei blocchi.
per spiegare come funzionano i blocchi è necessario introdurre due semplici strutture:

- `coordinate`
  contiene due valori interi rappresentanti la x e la y dei caratteri che compongono i blocchi
- `position`
  è semplicemente un vettore di 8 coordinate, perché ogni blocco è costituito da 4 quadrati e per formare un quadrato sono necessari 2 caratteri, quindi abbiamo $2\text{caratteri} \cdot 4\text{quadrati} = 8$

Nel costruttore viene poi inizializzato un vettore di `position` che contiene 4 elementi, questo perché ogni blocco ha 4 stati di rotazione, indicati dall'`id` della classe.
Abbiamo poi il rotation state, che indica quale tipo di rotazione ha assunto il blocco.
Gli ultimi elementi della classe sono gli offset, ovvero due valor x e y che rappresentano la distanza dall'origine degli assi del blocco, gli offset sono richiamati nella funzione `getPos` per indicare la posizione del blocco.

3.3.1 Le classi figlie di Block

Facendo uso dell'ereditarietà sono state costruite 7 classi figlie di `Block`, una per ogni tetramino. Nella definizione del costruttore delle classi si inizializzano subito tutte le coordinate dei caratteri che compongono i quadrati dei tetramini. E si imposta la posizione di partenza del blocco in per una corretta visualizzazione alla generazione del blocco.

3.4 La classe Classifica

Infine la classe classifica gestisce la lettura e la scrittura sul file `classifica.txt` Che, legge e salva su di un array gli elementi della classifica per evitare la continua apertura e chiusura del file. È poi presente il metodo che fa inserire al giocatore il nome, e quelli che gestiscono l'ordinamento della classifica.

4 Classifica

La classifica, viene visualizzata come un proseguimento del menù. Vengono visualizzati solamente i primi 10° Giocatori anche perché verranno salvati sul file `classifica.txt` solamente i 10 giocatori con il punteggio più alto.

Per prendere i dati dal file `classifica.txt` utilizziamo i metodi della libreria `<fstring>` per ragioni di specifiche del progetto. L'acquisizione dei caratteri avviene tramite un ciclo `while` che stampa utilizzando il metodo `mvwaddch()` per stampare carattere per carattere ogni elemento del file. Infine, viene anche utilizzato il metodo `mvwprintw()` per stampare la posizione in classifica dei nomi presenti nel file.