Università
Ca'Foscari
Venezia

Corso di Laurea in Informatica

INTRODUZIONE ALLA PROGRAMMAZIONE

Progetto X-Tetris

In linguaggio ANSI-C

Gruppo #026

Bettiol Luca 890424
Brognera Enrico 890406

A.A. 2021/2022

1) Progetto

Come progetto di Introduzione alla programmazione, abbiamo dovuto sviluppare in linguaggio ANSI C, X-Tetris, una variante del gioco Arcade Tetris.

Rispetto al gioco originale, X-Tetris prevede:

- La scelta della mossa, ovvero il giocatore può scegliere il pezzo da giocare, dove farlo cadere e con quale rotazione
- Il numero di tetramini disponibili per ciascuno dei 7 tipi diversi di blocchi sono 20 per la modalità Single Player e 40 per la modalità Multi Player.

Per lo sviluppo il gruppo ha usufruito solamente di funzioni appartenenti alla libreria `stdio.h` e `string.h`, utilizzando alcune stringe speciali per colorare il terminale, di conseguenza per la visualizzazione di questi colori occorre usare il terminale Linux, oppure il WSL in Windows.

Sono presenti tre modalità di gioco:

- Single Player, che permette di giocare fino a quando l'utente non termina i pezzi o non riesce più a posizionarli
- Multi Player, che permette a due utenti di giocare alternando l'inserimento delle proprie mosse
- Player vs CPU, che permette ad un utente di giocare contro una strategia della CPU

2) Istruzioni e specifiche di gioco

A. La scelta della modalità desiderata si può selezionare inserendo il corrispettivo valore numerico indicato a schermo, inserendo 9 invece, si potrà terminare e uscire dall'esecuzione del gioco.
B. Ogni turno di gioco è composto da due fasi:
    - La stampa degli elementi di gioco, quindi la lista dei blocchi e delle sue caratteristiche e successivamente la stampa del piano di gioco dell'utente
    - La richiesta della mossa dell'utente che si suddivide ulteriormente in:
        - Richiesta del pezzo da giocare
        - Richiesta della sua rotazione che può essere di 0°/90°/180°/270°
        - Richiesta della posizione in cui farlo cadere, questa corrisponde quindi al numero della colonna in cui inserire il primo pezzo più a sinistra del blocco scelto
C. Al termine della partita si potranno visualizzare i punteggi ottenuti dai giocatori, tornare al menù di gioco dove si potrà eventualmente intraprendere una nuova partita, o uscire dal gioco.

3) Struttura del progetto

Il progetto è stato suddiviso in diversi file che gestiscono le principali funzionalità del gioco.

I file si possono suddividere a coppie, ciascuna formata dal file.c, contenente le implementazioni delle funzioni, e dal file.h contenente le dichiarazioni e i relativi commenti per la documentazione.

I file presenti nel progetto sono i seguenti:

- `main.c` e `main.h`: contengono la struttura iniziale del gioco, quindi il menù e le modalità a disposizione per l'utente, quindi Single Player, Multi Player e Player vs CPU.
- `tetris_components.c` e `tetris_components.h`: contengono le strutture di base per il gioco e le funzioni per la loro inizializzazione.
- `tetris_print.c` e `tetris_print.h`: contengono le funzioni per la gestione della grafica.
- `input_control.c` e `input_control.h`: contengono le funzioni per la gestione e il controllo delle scelte effettuate dal giocatore.
- `tetris_operations.c` e `tetris_operations.h`: contengono tutte le funzioni per la gestione delle operazioni di gioco.
- `NPC_tetris.c` e `NPC_tetris.h`: contengono le funzioni per la gestione della strategia della CPU.

Per la manipolazione di tipi di dato, come i tetramini o il piano di gioco, abbiamo definito delle strutture (`struct`). Questa scelta deriva dai vantaggi che ne concedono queste strutture come la generalità o la scalabilità del codice permettendo in futuro di avere nuovi blocchi.

Le funzioni sono state sviluppate seguendo il principio del riuso del codice. Abbiamo quindi tentato di evitare di scrivere blocchi lunghi e molto simili tra loro, quando era possibile, definendo invece funzioni più brevi che potessero essere riutilizzate. Abbiamo infine fatto uso di costanti globali per la generalizzazione delle regole come la dimensione del piano di gioco, la quantità di blocchi e i colori, rendendo così veloce una possibile modifica di esse in un futuro.

La strategia scelta per lo sviluppo della CPU è basata sul calcolo e la ricerca della mossa con la qualità maggiore. Tale qualità viene calcolata grazie alla somma dei punteggi ottenuti da una funzione ricorsiva che percorre l'albero di gioco che si genera dalla mossa iniziale. Quindi la profondità dell'albero che andiamo a percorrere determina la qualità della mossa. Maggiore è la profondità maggiore sarà la precisione della scelta, ma allo stesso tempo maggiore sarà il tempo impiegato dal calcolatore per processare tale decisione.

4) Organizzazione del lavoro tra i componenti

L'organizzazione del lavoro è risultata semplice e chiara sia grazie ad una facile comunicazione tra i componenti sia grazie alla medesima formazione che ha permesso quindi un continuo confronto tecnico privo di discrepanze nelle conoscenze.

L'organizzazione del tempo si è quindi divisa in diverse fasi:

1. Problem-solving, nella quale abbiamo analizzato le caratteristiche del gioco e deciso le principali strutture da utilizzare.
2. Inizio della creazione di una prima versione semplice del gioco composta principalmente dalle funzioni per la modalità Single Player e con una interfaccia base verso l'utente.
3. Per ottimizzare meglio i tempi a disposizione abbiamo deciso di dividere il lavoro sviluppando contemporaneamente sia l'implementazione del Multi Player sia il miglioramento dell'interfaccia con l'utente.
4. Terminate quindi le funzioni principali abbiamo provveduto a riordinare il codice suddividendolo in più file.
5. Sviluppo della CPU, che si è suddivisa in una prima fase di problem-solving per la scelta della strategia e una seconda per la sua stesura.
6. Ottimizzazione, Clean Up del codice e stesura della documentazione è stata l'ultima fase.

Per quanto possibile abbiamo sempre cercato di lavorare cooperativamente al codice e anche nei momenti in cui la suddivisione del lavoro era necessaria c'è stata una continua comunicazione per affrontare le problematiche e le difficoltà che si incontravano.

5) Principali difficoltà riscontrate

Le principali difficoltà riscontrate sono state:

- La comprensione di tutte le regole e delle caratteristiche del gioco e l'opportuno adattamento delle strutture dati di cui avevamo bisogno.
- Una condivisione facile e veloce del codice tra i componenti, che è stata risolta con l'utilizzo della piattaforma "Github" che ha permesso anche un continuo backup delle varie versioni prodotte.
- La ricerca di come colorare i caratteri che vengono stampati a terminale senza l'utilizzo di librerie aggiuntive. Questo è stato risolto grazie alla scoperta di alcuni caratteri speciali che rendono possibile questa operazione.
- La ricerca di un sistema per la pulizia del terminale che però non richiedesse l'utilizzo di librerie specifiche per il sistema operativo su cui veniva eseguito il programma. Questo, però, non è ancora stato possibile risolverlo al meglio.
- L'ottimizzazione della strategia implementata dalla CPU, poiché la quantità di possibili combinazioni di mosse che si possono verificare richiedevano un tempo di attesa troppo lungo. Questo è stato quasi risolto grazie ad un compromesso tra il tempo di attesa e la qualità della mossa scelta.