

# Corso di Laurea in Informatica

INTRODUZIONE ALLA PROGRAMMAZIONE

Progetto X-Tetris

In linguaggio ANSI C

Bettiol Luca 890424

Brognera Enrico 890406

A.A. 2021/2022

## Progetto

Gli studenti sono stati chiamati a sviluppare, in linguaggio ANSI C, X-Tetris, una variante del gioco originale Tetris (<https://it.wikipedia.org/wiki/Tetris).>

Rispetto al gioco originale, X-Tetris prevede:

* + La scelta della mossa, che prevede la scelta del 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 il doppio per la modalità Multi Player.

Fondamentale citare il fatto che durante la fase di sviluppo il gruppo ha usufruito solo di funzioni appartenenti alla libreria stdio.h e string.h; di conseguenza la fase di compilazione può essere effettuata in qualsiasi sistema operativo, quindi, sia su Windows che su Linux e MacOS.

Sono presenti tre modalità di gioco, Single Player (permette di giocare fino a quando l’utente non termina i pezzi o non riesce più a posizionarli), Multi Player (permette a due utenti di giocare alternando l’inserimento delle proprie mosse) e Player vs CPU (permette ad un utente di giocare contro una strategia della CPU).

## Istruzioni e specifiche di gioco

1. La scelta della modalità desiderata si potrà selezionare inserendo il corrispettivo valore numerico, inserendo 9 invece si potrà terminare e uscire dall’esecuzione del gioco
2. Ogni turno di gioco sarà composto da due fasi:
   1. 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
   2. La richiesta della mossa dell’utente che è divisa 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
3. Al termine della partita si potranno visualizzare i punteggi ottenuti dai giocatori e tornare al menù di gioco dove si potrà eventualmente intraprendere una nuova partita.

## 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, contenete le implementazioni delle funzioni, e nel file.h contenete le dichiarazioni e i relativi commenti per la documentazione. I file contengono:

* main.c e main.h: contengono la struttura iniziale del gioco, quindi il menù e le modalità a disposizione per l’utente, quindi Sigle 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 rendendo quindi più facile la condivisione di esso tra i membri del gruppo.

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 o la quantità di blocchi, rendendo così veloce una possibile modifica di esse in un futuro.

## Organizzazione del lavoro tra i componenti

Il gruppo che presenta questo progetto è frutto di una fusione di due gruppi, il Gruppo 16 e il Gruppo 28. Originariamente in entrambi i gruppi si sono riscontrate, nei membri non dotati di una formazione tecnica, delle difficoltà circa le conoscenze e la capacità di problem-solving; è stata pertanto necessaria una fase, modestamente lunga, ma doverosa, di formazione circa determinate tematiche come, a titolo esemplificativo ma non esaustivo, strutture dati, quali vettori e matrici, e la definizione ed uso di *structure* (struct).

Al termine del primo semestre, i membri di entrambi i gruppi apprendono che dei loro componenti intendono ritirarsi dagli studi, non consentendo quindi lo sviluppo in teamwork del progetto; contestualmente la nostra comunicazione e richiesta, poi approvata dai docenti, di unire i due gruppi.

Per quanto concerne l’organizzazione del lavoro tra i componenti, si riporta che Jacopo Cini ha curato principalmente la parte grafica, mentre Matteo Grazioso e Italo Albano hanno curato la strategia di gioco. Naturalmente si tratta di una suddivisione relativa in quanto i membri del gruppo si sono aiutati a vicenda in caso di problematiche e difficoltà.

Il lavoro di gruppo ha avuto un eccellente impatto su ognuno dei membri, non solamente per l’introduzione, almeno per un membro del gruppo, allo sviluppo di software non individuale, ma anche e soprattutto perché le differenze in conoscenze, capacità specifiche e background dei membri hanno fatto sì che ognuno colmasse delle lacune altrui, come un ferro ne affila un altro.

## 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 rendevano 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 richiedevo 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.