Progetto Programmazione 2021

Gianpaolo Roversi - Mattia Maranzana - Lorenzo Pinelli

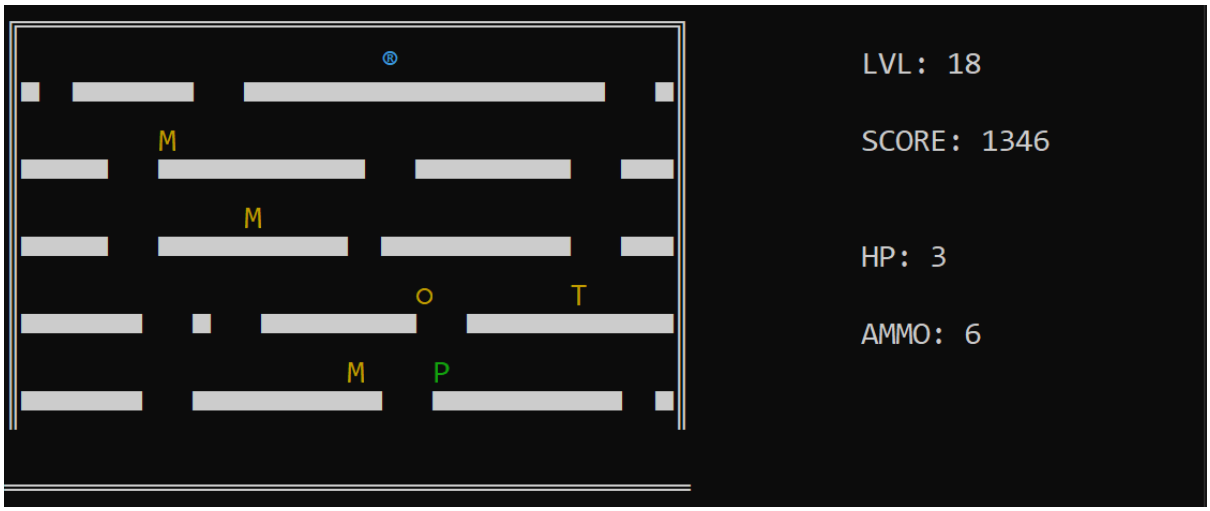
Il Gioco: Trama

> _>Player?
> _>...
> _>Player mi senti?
> _>...
> _>Player siamo vicini alla base nemica 211, ma la bufera non ci permette di recuperarti. Abbiamo sottovalutato l’Antartide.
> _>...
> _>Mi dispiace ma da ora sei solo, e sappiamo che in giro sono rimaste numerose truppe Maziste. Fatti valere, buona fortuna soldato.
> _>...

Il Gioco: Comandi

- WASD: Movimento
- J: sparo a sinistra
- K: sparo a destra
- P: pausa

Il Gioco: Meccaniche



Player avanza nelle varie stanze attraverso le porte affrontando i diversi Nemici che incontra e raccogliendo Bonus.

I Nemici sono di due tipi:

- M: Mazista ordinario, pattuglia la sua zona, a volte si muovono in coppia.
- T: Mazista armato di Torretta, è sempre solo e continua a sparare nella direzione di Player.

I Bonus sono di tre tipi:

- ♥: Kit Medico, Player ha un massimo di tre vite, che perde imbattendosi nei Mazisti e venendo colpito da un proiettile, e recupera con i Kit Medici.
- ©: Coin, Player ottiene punti aggiuntivi trovando Segreti Militari nascosti in giro per la base 211.
- ®: Magazine, Player recupera munizioni.

Player ha due modi di attaccare i Mazisti: sparare e colpire dall'alto.

I Bonus colpiti dai proiettili sono distrutti per sempre senza ottenere la ricompensa.

Codice: Struttura

Tutti i files .hpp tranne Constants sono classi.

La classe **Cast** è la classe padre di tutti gli elementi che popolano i livelli: **Hero, Enemy, Item, Bullet**.

La classe **Room** descrive il singolo livello tramite l'attributo **view**: un array di caratteri contenente muri, piattaforme e le sottoclassi di Cast.
Room possiede tre liste che memorizzano le varie istanze di Enemy, Item e Bonus, ciascuna con il corrispondente puntatore alla testa: currentMonsters, currentAmmo, currentBonus.
Room ha un array chiamato **freeRow** di elementi di tipo **Control**: una struct ausiliaria per gestire le righe di view.

La classe **Game** gestisce la visualizzazione a schermo di view e delle informazioni sulla partita.
Contiene la lista delle Room create dinamicamente ogni volta che Player avanza di livello.

Constants contiene e definisce le costanti che abbiamo sfruttato nel progetto, come le dimensioni della console e gli alias dei caratteri usati in view.

Codice: Funzioni Principali

logic()

Funzione principale chiamata nel main(), si occupa delle scritte iniziali e finali e tutto ciò che avviene nel gioco, come il movimento di proiettili e nemici, la cattura degli input da tastiera, la visualizzazione a schermo e il cambio di stanza.
Contiene il ciclo while che determina la fine della partita in caso di vite o score che raggiungono lo zero.

bulletMove(), enemyMove()

Proiettili e Nemici sono gestiti tramite liste monodirezionali. bulletMove() e enemyMove() scorrono la lista e aggiornano le posizioni.

move(getch())

Funzione che sfrutta la funzione **getch()** di ‘conio.h’ e controlla che l’input da tastiera sia un comando valido, quindi agisce di conseguenza.

toCharInfo()

Funzione che copia l’array view in un array di Char Info e aggiorna le scritte che mostrano le informazione della partita.
Come ultima cosa chiama la funzione **stampView()** per mostrare a schermo le modifiche.

prevRoom(), nextRoom()

Quando Player raggiunge una porta viene visualizzato il cambio di livello, tornando indietro nella lista di Room se si tratta della porta sinistra, o generando una nuova Room se si tratta di quella di destra.

roomGenerator()

Funzione che controllando il livello raggiunto, genera le piattaforme, inizializza e visualizza bonus e nemici, infine crea i buchi nelle piattaforme.

drillRow()

Funzione che genera i buchi in base a tre situazioni:
-riga sovrastante non popolata, genero un numero di buchi in base al livello in posizioni casuali;
-riga sovrastante contenente un elemento (Bonus o Enemy), divido la riga in quattro sezioni e buco nelle tre sezioni in cui non si trova l’elemento;
-riga sovrastante contenente due nemici, divido la riga in quattro sezioni e buco nelle due sezioni in cui non si trovano i nemici nel caso in cui si trovino in due sezioni diverse, mentre se si trovano nella stessa sezione si considera la metà riga in cui si trovano, e si buca l’altra.