Università
Ca'Foscari
Venezia

Documentazione progetto

TheMazeInF#

Corso di “Introduzione alla programmazione”

Anno Accademico
2019/2020

Docente
Alvise Spanò

Gruppo
LabProg2019 10

Studenti
Elia Bertapelle (881359@stud.unive.it)
Leonardo Piccolo (882351@stud.unive.it)

Indice

Requisiti

Il progetto consiste nella realizzazione di un semplice videogioco in stile ASCII ART. In particolare, viene richiesto di sviluppare un programma che permetta la creazione di labirinti (maze) dove poter giocare in modalità interattiva o in modalità automatica.

I requisiti minimi per questo progetto sono:

- implementazione dell'algoritmo di generazione di un labirinto casuale w x h; è inoltre richiesta la sua corretta visualizzazione su schermo, mediante l'utilizzo del motore grafico messo a disposizione, oppure attraverso altri metodi concordati con il docente;
- sviluppo della modalità di gioco interattiva: un giocatore può dare comandi interattivi mediante l'utilizzo di un dispositivo di input data (es: tastiera) al fine di poter risolvere il labirinto generato. L'interazione deve essere facilmente visualizzabile per l'utente finale tramite la console;
- sviluppo della modalità automatica di risoluzione: il programma deve poter risolvere automaticamente il labirinto generato e mostrare graficamente la soluzione al giocatore. L'utente utilizzatore del programma deve poter scegliere tra le diverse modalità di gioco, possibilmente tramite un menù oppure una qualche altra forma interattiva di interfaccia grafica.

Linguaggi utilizzati

Come da richiesta, il codice del nostro programma è scritto in f#. Le variabili, i tipi e le firme delle funzioni sono in genere autoesplicative e in inglese. Abbiamo deciso però di scrivere i commenti in italiano, perché abbiamo considerato chi effettivamente sarebbe andato a lavorare sul progetto e, avendo entrambi come lingua madre l'italiano, abbiamo deciso che scrivere i commenti in inglese avrebbe potuto portarci a perdite di tempo e incomprensioni in modo particolare nel caso dovessimo mettere mano al codice in futuro.

Descrizione file di progetto

Motore grafico

Il motore grafico su cui il nostro programma è basato ci è stato fornito come base di partenza su cui strutturare tutto il nostro progetto.
Su di esso, abbiamo effettuato alcune modifiche perchè potesse risolvere alcune nostre necessità. In particolare abbiamo aggiunto nel file Engine.fs una funzione *refresh* che quando richiamata esegue un aggiornamento della schermata. Questo metodo verrà utilizzato durante la risoluzione automatica, in quanto abbiamo la necessità di eseguire il refresh dello schermo ad ogni spostamento del nostro risolutore e non solo alla pressione di un tasto.
I sorgenti e la documentazione del motore si possono trovare ai seguenti link:

- Sorgenti
- Documentazione e consegna del progetto

Main.fs

Questo file ci è stato assegnato quasi pronto, al suo interno conteneva già alcune delle funzioni principali del programma, come la creazione della finestra di log. A questo abbiamo aggiunto le chiamate alle nostre funzioni, prima il main di Menu.fs. Poi a seconda delle scelte effettuate nel menu, viene richiamata la funzione che avvia la corrispondente modalità di gioco. Al main di Maze.fs passo modalità di gioco e risoluzione scelta.

Menu.fs

Questo file contiene la gestione del menù del gioco. Questo menù consente di scegliere prima di tutto il gioco a cui si vuole giocare e poi, la grandezza del labirinto.
Per memorizzare questa scelta ci siamo affidati a due variabili mutable "gameMod" e "gameRes" che contengono rispettivamente la modalità di gioco (definite nel file di configurazione) e le 5 risoluzioni che abbiamo scelto per il nostro labirinto (15*15, 25*25, 51*51, 99*99, 99*51).

Le varie scelte potranno essere effettuate spostando il cursore rosso verso l'alto o verso il basso premendo 'w' e 's'. In qualsiasi momento premendo il tasto 'q' il giocatore potrà tornare al menù precedente o uscire dal gioco a seconda di dove si trova.

Maze.fs

Modalità di gioco

Le diverse modalità di gioco vengono avviate in base al valore passato come parametro al *main* e sono le seguenti:

Interattiva ~ single player

Descrizione

In questa modalità di gioco può giocare solo un utente per volta. Lo scopo del gioco è quello di raggiungere l'angolo in basso a destra partendo da quello opposto.

Istruzioni di gioco

L'utente potrà muoversi utilizzando 4 tasti:

- 'w' spostamento di una posizione verso l'alto
- 'a' spostamento di una posizione verso sinistra
- 's' spostamento di una posizione verso il basso
- 'd' spostamento di una posizione verso il destra

La pressione del tasto 'q' effettuerà l'uscita dal gioco e il ritorno al menù

Interattiva ~ multi player

Descrizione

Possono giocare due giocatori in contemporanea. I giocatori inizieranno in alto a lati opposti e dovranno raggiungere l'angolo in basso opposto a loro.

Istruzioni di gioco

L'utente 1 potrà muoversi utilizzando 4 tasti:

- 'w' spostamento di una posizione verso l'alto
- 'a' spostamento di una posizione verso sinistra
- 's' spostamento di una posizione verso il basso
- 'd' spostamento verso destra di una posizione

L'utente 2 potrà muoversi utilizzando 4 tasti:

- 'i' spostamento di una posizione verso l'alto
- 'j' spostamento di una posizione verso sinistra
- 'k' spostamento di una posizione verso il basso
- 'l' spostamento verso destra di una posizione

La pressione del tasto 'q' effettuerà l'uscita dal gioco e il ritorno al menù

Automatica

Descrizione

In seguito alla pressione del tasto ‘s’ il labirinto verrà risolto automaticamente dal computer mostrando il percorso fatto.
Sfrutta la variabile “stop” per fermare la ricerca e ignorare i risultati della ricorsione.

Istruzioni di gioco

- ‘s’ inizio ricerca automatica del percorso corretto
- ’q’ uscita dal programma (non effettuabile durante l’esecuzione del risolutore)

Interattiva + ~ single player

Descrizione

Questa modalità di gioco rispetto alla modalità interattiva single player aggiunge solamente la valorizzazione del killer point che farà terminare precocemente il programma.

Implementazione

Abbiamo cercato il più possibile di evitare di riscrivere codice, motivo per cui ci sono molte funzioni in comune tra tutte le varie modalità.

Generazione del labirinto

Per la generazione del labirinto, ci siamo affidati ad un algoritmo trovato online che abbiamo implementato e modificato per ottenere una matrice valorizzata con i valori “wall” e “path” che indicano rispettivamente il muro e il percorso.
Useremo questa matrice in diverse occasioni:

- Generazione di un array di pixel per la visualizzazione, tramite il motore grafico, del labirinto
- Controllo durante il gioco della presenza dei muri per permettere o negare lo spostamento

Algoritmo

L’algoritmo di generazione funziona al seguente modo:

- Inizializzazione della matrice a “wall”
- Scelta randomica di un punto da cui partire, settandolo a “path”
- Estensione ricorsiva del “path” a partire dal punto ottenuto, collegandolo di volta in volta a uno a caso dei suoi vicini (celle con distanza di 2)

My_update

Questa funzione è quella che gestisce i movimenti dello sprite effettuando anche i vari controlli del caso. Viene passata al motore grafico nella modalità interattiva, in quella automatica viene richiamata direttamente da una nostra funzione, mentre in quella multigiocatore viene utilizzata da una funzione supplementare, passata al motore grafico, in modo da gestire i due diversi giocatori.

Ricerca del percorso

Per effettuare la ricerca del percorso, abbiamo pensato di utilizzare un algoritmo di ricerca in profondità, provando prima il percorso di destra e in basso per ottimizzarne la ricerca.
Le istruzioni eseguite sono:

1. Controllo della possibilità di spostamento tramite una funzione *trymove* che esegue un controllo sulla struttura dati in cui è memorizzato il labirinto
2. Controllo che la posizione di destinazione non sia già stata visitata accedendo alla matrice *Visited* di tipo *Visit*
3. Se entrambi i controlli vanno a buon fine eseguo lo spostamento attraverso la funzione *my_update*
4. Quando nessuna delle quattro vie è percorribile torno indietro colorando di rosso la strada.

Avvio del programma

Il main accetta come parametro la modalità di gioco e la risoluzione, una volta ottenute inizializza le variabili necessarie al funzionamento del programma e istanzia l'engine. In base al GameMod ottenuto lancia l'esecuzione della porzione di programma desiderata.

KillerPoint

È un punto invisibile che viene inizializzato all'avvio del programma e che solo nella modalità interattiva+ prende un valore all'interno del labirinto. Questo punto quando attraversato blocca il programma e ritorna "Game Over" al player.

Visualizzazione

Per garantire una visualizzazione quadrata del labirinto, abbiamo deciso di raddoppiare tutte le misure in larghezza senza intaccare però le modalità di memorizzazione.
La strada percorsa dagli utenti sarà visualizzabile durante il gioco attraverso la creazione di uno sprite colorato per ogni pixel percorso dall'utente. Nella modalità automatica, segnaliamo di due colori diversi la strada corretta e quella errata.

Conclusioni

Il software finale è funzionante, collaudato e testato. Risponde a tutti i requisiti, con l'aggiunta di alcune funzionalità extra come la modalità multigiocatore e la modalità giocatore interattiva+.
La struttura del programma inoltre, è predisposta per una facile implementazione futura di nuovi giochi e modalità di gioco.