Progetto di programmazione

(Anno accademico 2016/2017)

***HALLOWEEN NIGHT***

Storia:

È la notte del 31 ottobre, il cielo si è fatto grigio cupo e la città silenziosa; tutti si stanno preparando alla notte più spaventosa di tutti gli anni. C’è chi per troppa paura si chiude in casa, e chi come te, coraggioso e un po’ spavaldo decide di uscire per le strade della città.

E se l’antica leggenda diventasse realtà?!

Stai attento ragazzo/a non sai cosa ti aspetta là fuori, mostri(M) di ogni tipo sono pronti ad attaccarti, guardati sempre alle spalle, non ti fidare di nessuno e cerca di tornare a casa vivo stanotte.

C’è solo un modo per sopravvivere e mille per morire, nessuno ti aiuterà pensa a raccogliere più dolcetti che puoi, impara scherzi sempre più spaventosi per riuscire a combattere i mostri, e mi raccomando se dovessi trovare una lanterna prendila, ma usala solo quando sarà davvero necessario, perché potrebbe essere l’unica che troverai sul tuo cammino.

I dolcetti (o) saranno le tue energie, ciò che ti manterrà in vita.

Gli scherzi(a) saranno per te armi per combattere i mostri che cercheranno di ucciderti.

Con la lanterna(v) che ti illumina la strada, avrai meno paura dei mostri.

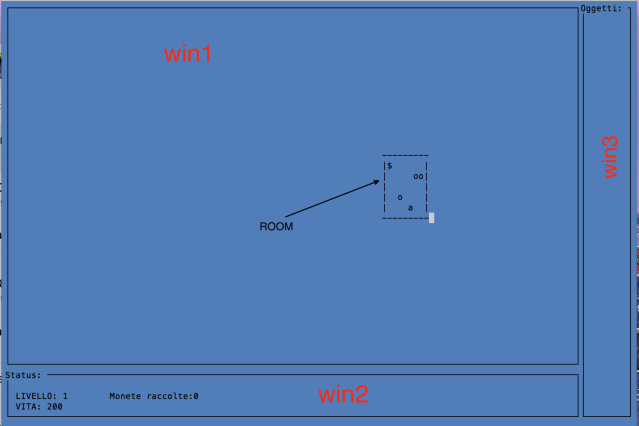
Vuoi sopravvivere?! …. Raggiungi casa (c) tua, sempre se riuscirai a ricordarti la strada.

Implementazione:

Il terminale:

Abbiamo scelto di suddividere il terminale in tre sotto-finestre:

* la finestra win1 dedicata al gioco, alla visualizzazione della mappa
* la finestra win2, “Status” usata per indicare all’utente il livello corrente, la propria vita e in caso di attacco la vita del mostro, quanti dolcetti ha raccolto
* la finestra win3 che contiene una lista di oggetti che l’utente raccoglie e che deve ancora



La mappa:

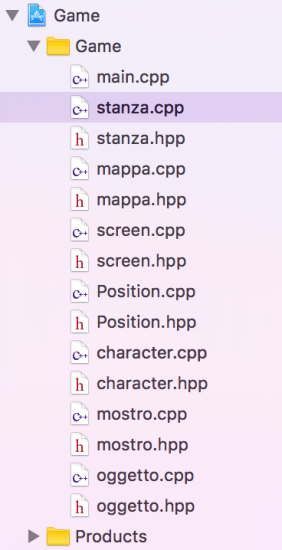
Abbiamo costruito la mappa tramite un vettore bidimensionale, che ci permette di avere infiniti livelli.

Il vettore bidimensionale contiene un puntatore alla classe stanza, dove all’interno troviamo un membro room che è una matrice di carattere che identifica una singola stanza.

Il codice:

L’intero codice è stato suddiviso in classi, per rendere la programmazione più semplice e più comprensibile.

Abbiamo create le seguenti classi:



ognuna di esse composta dal file .cpp e dal header file .h.

Spiegazione diverse classi:

* stanza: rappresenta la stanza già definita(7 righe e 9 colonne) che viene richiamata nella mappa. Al suo interno sono presenti le funzioni: creastanza() dove viene implementata e wprintroom() nel quale stampiamo nella finestra principale la stanza già creata.
* mappa: come già detto prima è formata da una lista bidimensionale (vector<vector>) dove le stanze istanziate verranno memorizzate. Le funzioni sono: creamappa() dove si inseriscono nel vettore 2d le stanze; esistestanza() controlla l’esistenza delle stanze e memorizza in una matrice 1 se è presente e 0 altrimenti(funzione che serve soprattutto per l’implementazione del nostro codice); stanzasucc() crea una stanza adiacente a una già creata( la posizione di questa stanza viene decisa tramite un numero random da 1 a 4); istanzia() istanzia una stanza all'interno della mappa nella posizione x,y presa in input e controllo() funzione che toglie la porta nel caso in cui la stanza succesiva non viene istanziata perchè tocca il bordo della finestra.
* screen: classe per gestire la stampa su finestra tramite libreria ncurses/curses. Funzioni: inizializzafinestra() che inizializza le diverse finestre win1,win2,win3; statuslivello() dove c’è scritto in che livello sei; statusmonete() incrementa il numero di monete ogni volta che vengono raccolte; statusvita() per sapere il valore della vita che cambia a seconda degli avvenimenti; statusmostro() si può leggere la vita del mostro.
* Position: classe creata per gestire principalmente il movimento del personaggio principale. Funzioni sono: posmove() che permette al personaggio di muoversi con le frecce; passaporta() se c’è una porta la può attraversare; crealivello() se si supera un obiettivo si incrementa il numero di livello; generachiave() dove aver superato l’obiettivo compare una chiave che permette di passare al livello successivo; generamostri() aumenta la potenza del mostro a seconda del livello; generaoggetti() genera oggetti.
* Character: gestisce i personaggi.Funzioni: vivo() che se la vita diventa 0 il gioco termina; raccogli oggetti() raccoglie gli oggetti quando li trova; usa oggetti() gli oggetti vengono usati; stampa oggetti() si stampano gli oggetti;
* mostro:sottoclasse di character e serve a implementare i mostri. Funzioni: movimento() fa muovere il mostro tramite una generazione casuale di un numero; charactervsmostro() quando il personaggio si avvicina al mostro lo attacca e mostrovscharacter() che fa il contrario.
* oggetto: classe in cui generiamo i diversi oggetti.

Il gioco termina quando il personaggio principale muore o quando si raggiunge la lettera ‘c’ che rappresenta la fine del gioco e quindi la vittoria.