PROGETTAZIONE GIOCO DA TAVOLO “MONOPOLY”

DESCRIZIONE: Programma che permette di simulare una partita con più giocatori locali a monopoly.

-REGOLE:

- Si può giocare fino a 4 giocatori

- Vince l’ultimo che ha soldi disponibili o il più ricco dopo un certo numero di turni

- La quantità di soldi iniziale di ogni giocatore viene decisa in base al numero di partecipanti

- I giocatori iniziano senza alcuna proprietà

- Durante il proprio turno, il giocatore può richiedere uno scambio ad un altro giocatore

- Tutte le azioni che il giocatore può eseguire devono essere fatte prima del tiro dei dadi

-FUNZIONALITA’:

- I giocatori possono essere sia umani che controllati dal computer

- L’utente può decidere se la partita possa continuare finchè rimane un solo giocatore con dei soldi, o se la partita finisca dopo un determinato numero di turni

- La posizione iniziale della pedina è già definita e non può essere cambiata

- L’ordine del turno è in senso antiorario (Non viene scelto tirando un dado all’inizio)

- Le pedine si muovono automaticamente a seconda del punteggio dei dadi ottenuto

- Una volta tirati i dadi, il turno del giocatore corrente finisce e si passa a quello successivo

Classi

**Giocatore**: E’ distinto dal proprio colore, una lista di tutte le proprietà che possiede e un numero di soldi, che servono anche a verificare se l’utente può continuare a giocare o ha perso.

+ Giocatore(C: brush, S: int) - Costruttore della classe giocatore, assegna il colore allo stesso e il numero di soldi iniziali, che variano a seconda del totale di partecipanti.

+ Compra(C: Casella): bool - Metodo che viene richiamato quando il giocatore vuole acquistare la casella su cui è. Restituisce TRUE se il giocatore ha abbastanza soldi per comprare la proprietà, FALSE il contrario.

+ Vendi (C: Casella): bool – Metodo che viene richiamato quando il giocatore decide di vendere (ipotecare) una proprietà. Se la proprietà appartiene allo stesso giocatore che la vuole vendere, ritorna TRUE. Se si sta cercando di vendere una proprietà di qualcun altro, non verrà venduta (FALSE).

+ Vendi (S: Struttura, C: Casella) – Questo metodo serve per vendere le strutture (case o alberghi che siano) costruite sulla proprietà. Allo stesso modo, il giocatore può sempre vendere strutture (ammesso che ce le abbia).

**Struttura : Image**: Questa classe serve solamente per distinguere il tipo di struttura che si vuole utilizzare per compiere un’operazione ( Casa o albergo ).

+ Struttura(T: bool) – Costruttore della classe che ritorna TRUE se la struttura è una casa, FALSE se è un albergo

**Casella**: Classe astratta, ereditata per il colore delle caselle.

+ Casella (Colore: brush) – Costruttore, assegna il colore alla casella.

**Proprieta : Casella**: Classe che serve per istanziare una proprietà normale con un costo. La proprietà potrà essere ipotecata dal proprietario, e quest’ultimo potrà costruirci sopra le strutture.

+ Proprieta(Cost: int, Color: brush) – Costruttore, viene assegnato il prezzo d’acquisto base della proprietà e il suo colore.

**Speciali : Casella**: Classe che serve per istanziare le proprietà speciali non acquistabili, come la prigione e le carte di imprevisti o probabilità. Avrà un tipo di casella speciale (come quelli riportati prima) e, se si tratta di una tassa, avrà anche l’importo da pagare.

+ Speciali(Tipo: Tipo\_Speciali, Color: brush): Costruttore, viene assegnato il tipo di proprietà speciale non acquistabile e il suo colore.

**Proprieta\_Speciali : Casella**: Classe che serve per istanziare le proprietà speciali acquistabili, come le stazioni, società elettrica e società idrica. Queste proprietà avranno un costo base, e potranno essere ipotecate dal proprietario.

+ Proprieta\_Speciali(Cost: int, Color: brush) – Costruttore, viene assegnato il costo alla proprietà e il suo colore.

**Tipo\_Speciali**: Enumeratore che definisce il tipo di proprietà speciale (Prigione, imprevisti, ecc.)