GOP (Gioco dell'Oca Pazza)

Introduzione

Versione digitale del gioco dell'oca, con alcune rivisitazioni.

- Trello (gestione attività): https://trello.com/b/ccJxlR3O
- GitHub: https://github.com/Zimm1/GOP

Grafica

Per la realizzazione di questo gioco è stato prediletto l'approccio universale rispetto alla grafica, perciò l'unico strumento richiesto è il terminale e vengono unicamente utilizzate librerie multi-piattaforma.

Tema

La scelta è ricaduta su una passione in comune del team, la Musica.
Ogni membro ha infatti contribuito con il proprio genere musicale preferito.
Sono quindi presenti domande riguardanti: Rock, Metal, Musica Italiana, Pop, EDM.

Caratteristiche

- 1-4 giocatori
- Carte
  - Rispondere a una domanda
  - Vai avanti di n caselle
  - Torna indietro di n caselle
  - Scambia posizione con giocatore successivo
- Caselle azione
  - Pesca una carta
  - Vai avanti di n caselle
  - Torna indietro di n caselle
  - Salta 1 turno.
  - Torna all'inizio.

Scelte implementative

ANSI

ANSI è una sequenza di caratteri ASCII che permette di controllare le funzioni a video (Colore del carattere o posizione del cursore).
In generale l'escape sequence indica una sequenza di byte contenuta all'interno di un testo che i terminali cercano e interpretano come comandi.

In questo progetto sono state utilizzate solo le funzioni che controllano il colore del testo, la sequenza di caratteri è composta da `ESC[Value;...;Valuem`:

- ESC: "Escape" carattere `\033` nella codifica ASCII in base ottale.
- Value: Si possono concatenare vari valori separati da `;`, questi valori riguardano:
  - Colore del testo.
  - Colore dello sfondo.
  - Attributi del testo (sottolineato o grassetto).
- m: carattere finale che indica la fine della sequenza di caratteri ANSI.

Array di char

Per lo sviluppo di questo progetto abbiamo prediletto l'utilizzo degli array di char in quanto utilizzando principalmente la libreria **cstring**, che fornisce utili funzioni per la manipolazione di stringhe. Le funzioni appartenenti a questa libreria richiedono come parametri puntatori ad array di char (es `strcpy ( char * destination, const char * source );`). Inoltre un altro vantaggio degli array di char è quello di poter specificare la lunghezza massima che possono avere, evitando in questo modo di allocare spazio in più che non verrà utilizzato come invece accade con **string**.

Domande a tema

Le domande, create a tema Musica, propongono 4 possibili risposte, delle quali solamente una è corretta.
In base alla correttezza della risposta data, danno un beneficio o uno svantaggio al giocatore.
Sono organizzate in un array di strutture (struct) "Question".
Tramite una funzione, la quale ritorna un valore booleano, viene scelta in modo randomico una domanda, visualizzata, presa in input la risposta e valutata.

Programmazione ad oggetti

Al fine di rendere il progetto modulare e facilmente estendibile, è stato prediletto il paradigma ad oggetti.
Esso infatti permette di separare i componenti principali in blocchi semplici e autonomamente funzionanti, i quali sono utilizzati in seguito per implementare funzionalità più complesse.

Oggetti sviluppati:

Deck - *Mazzo di carte*

- Permette di pescare una carta, scelta in modo randomico in base alla percentuale associata ad ognuna di esse
- Potenzialmente estendibile, nel caso si vogliano implementare più mazzi di carte diversi fra loro

Card - *Carta*

- Classe base, estesa dagli oggetti più specifici
- Contiene un parametro "messaggio" (array di char)
- Espone un costruttore vuoto (per permettere l'ereditarietà), un costruttore con il messaggio, 2 metodi per leggere e modificare il messaggio, un metodo "effetto"
- Estesa da:
  - **Question Card** - *"Domanda"*
    - Possibilità di estrazione: 60%
    - Propone una domanda a tema
    - Si converte in Go Ahead Card o Go Back Card in base alla risposta
  - **Go Ahead Card** - *"Vai avanti"*
    - Possibilità di estrazione: 0% (15% considerando 1/4 risposte corrette)
    - Contiene un parametro "spostamento", scelto in modo randomico da 1 a 6
  - **Go Back Card** - *"Vai indietro"*
    - Possibilità di estrazione: 0% (45% considerando 3/4 risposte sbagliate)
    - Contiene un parametro "spostamento", scelto in modo randomico da 1 a 6
  - **Switch Position Card** - *"Scambia posizione"*
    - Possibilità di estrazione: 20%
    - Scambia la posizione del giocatore con quella del giocatore successivo
    - Se si gioca con un solo partecipante, non ha effetto
  - **Throw Again Card** - *"Tira ancora"*
    - Possibilità di estrazione: 20%
    - Permette di effettuare un'altro tiro di dadi

Square - *Casella*

- Classe base, estesa dagli oggetti più specifici
- Contiene un parametro "messaggio" (array di char), un parametro "colore" (enumeratore)
- Espone un costruttore vuoto (per permettere l'ereditarietà), un costruttore con il messaggio, 2 metodi per leggere e modificare il messaggio, un metodo "effetto"
- Ogni classe figlia è contraddistinta da un colore
- Estesa da:
  - **Void Square** - *"Vuota"*
    - Possibilità di estrazione: 100% □ 70% □ 40% □ 10% □ 0%
    - Durante la generazione del tabellone, se viene scelta la sua percentuale si abbassa, al contrario, se non viene scelta, la sua percentuale torna a 100%

- **Start Square** - *"Partenza"*
  - Possibilità di estrazione: 1
  - Casella iniziale fissa
- **Finish Square** - *"Arrivo"*
  - Possibilità di estrazione: 1
  - Casella finale fissa
- **Draw Card Square** - *"Pesca una carta"*
  - Possibilità di estrazione: se non viene scelta la casella vuota, 45%
  - Estrae una carta dal mazzo
- **Move Square** - *"Vai avanti/indietro"*
  - Possibilità di estrazione: se non viene scelta la casella vuota, 42%
  - Contiene un parametro "spostamento", scelto in modo randomico da -6 a 6 (non 0)
  - Spostamento eventualmente modificato in modo da muovere il giocatore esclusivamente su caselle vuote
- **Miss Turn Square** - *"Salta N turni"*
  - Possibilità di estrazione: se non viene scelta la casella vuota, 10%
  - Nega il tiro dei dadi per N turni successivi
  - Contiene un parametro "turni" da saltare
  - Usata esclusivamente con "turni" = 1 per bilanciare il gioco
- **Back Start Square** - *"Torna all'inizio"*
  - Possibilità di estrazione: se non viene scelta la casella vuota, 3%
  - Sposta il giocatore sulla casella iniziale

Player - *Giocatore*

- Contiene i parametri "nome" (array di char), "posizione" (int) e "colore" (enumeratore)
- Tiene traccia della posizione del giocatore
- Ogni partecipante è contraddistinto da un colore

Game - *Gioco*

- Classe principale, chiamata dal Main del programma
- Usa tutti gli altri oggetti
- Contiene un array di *Player*, un array di *Square*, un *Deck*
- Metodi principali:
  - **Init Players** - *Generazione giocatori*
    - Input del numero di giocatori (da 1 a 4)
    - Input dei nomi dei giocatori
  - **Init Squares** - *Generazione caselle*
    - Crea il tabellone secondo le percentuali prestabilite
  - **Game Loop** - *Ciclo di gioco*
    - Stampa il tabellone e comunica il giocatore corrente e la sua posizione
    - Effettua il tiro di dadi (2 da 6 facce)
    - Sposta il giocatore ed esegue eventualmente l'effetto della casella di arrivo

    - **Show End** - *Messaggi finali*
        - Output tabellone finale, posizioni dei giocatori, vincitore

Strumenti utilizzati

Durante lo sviluppo il team si è appoggiato ad alcuni fondamentali strumenti:

Git

- Tecnologia che agevola lo sviluppo in contemporanea di un progetto riducendo drasticamente i conflitti tra parti di codice diverse
- Permette di tenere la cronologia di tutte le modifiche apportate (repository)
- Rende il progetto facilmente condivisibile ad altri sviluppatori per la visualizzazione, l'utilizzo o la modifica
- GitHub
    - Principale servizio di hosting per repository Git
    - Usato grazie ai benefici dell'account universitario UniBo

Trello

- Eccellente strumento per la gestione di attività, incarichi e andamento di un progetto
- Permette di organizzare lo sviluppo in micro-attività assegnabili a membri differenti del team, con la possibilità di controllarne lo stato di avanzamento e l'effettivo completamento

CLion

- IDE sviluppato da JetBrains per lo sviluppo in C e C++
- Fornisce molteplici strumenti, come l'integrazione con Git, l'auto-completamento intelligente e il supporto per linguaggio MarkDown
- E' stata utilizzata una piattaforma di sviluppo comune per ridurre al minimo i problemi di compatibilità interni al team
- Licenza inclusa nei benefici dell'account universitario UniBo

MarkDown

- Linguaggio di markup con una semplice sintassi che permette la creazione di documenti di testo convertibili facilmente in molteplici formati come HTML, DOC, PDF, TXT, Rich Text Format (RTF)
- Usato per la creazione del ReadMe e della Relazione
- Supportato nativamente da GitHub per fornire una descrizione del progetto

Note Finali

Il progetto è stato pensato, realizzato e documentato per essere reso pubblico e open-source, in modo da fornire un esempio concreto del paradigma ad oggetti in C++, applicato allo sviluppo di un semplice gioco, ai neofiti del linguaggio o della programmazione in generale.

Il team

- **Simone Cavazzoni** - Website
- **Stefano Notari**
- **Alessandro Tedeschi**