Progetti LPO 23

Consiste nel realizzare un progetto che è costituito da una applicazione standalone realizzata in Java.Il progetto può essere realizzato da un gruppo composto da un minimo di 1 ad un massimo di 3. In generale, uno deve essere l'eccezione non la regola.
Potete scegliere il progetto tra le tre possibili idee progettuali previste nel documento in allegato(progetti.pdf).
Vi ricordiamo che il progetto ha un peso del 50% (quindi max. 15 punti) sulla valutazione finale.
Il restante 50% della valutazione consisterà in un colloquio orale singolo volto ad accertare le conoscenze acquisite durante il corso. Inoltre, nel colloquio ci sarà anche una discussione del progetto.
NOTA: Il progetto deve essere consegnato 5 giorni prima dell'appello inviando una e-mail al docente manifestando la volontà di sostenere l'esame. Pertanto, il docente e considereranno il progetto sviluppato entro tale scadenza. Ciò vuol dire che qualunque modifica apportata successivamente non verrà presa in considerazione. Tale regola non vale per il primo appello della sessione estiva (ovvero primo appello di giugno 2022). Chiaramente bisogna effettuare la registrazione anche nel sistema di segreteria virtuale.

FORM

Per la composizione dei gruppi è necessario compilare la seguente Microsoft form:

https://forms.office.com/Pages/ResponsePage.aspx?id=fIrwndcxJECbpI7V76waAczDmqkIFIJFtN5nFVryuRxUQTQzSERUWUY2RE85NTg4WExLQVFDTFVFTiQlQCN0PWcu .
La compilazione della form deve avvenire da parte di uno dei componenti del gruppo. Vi chiediamo cortesemente di compilare la form con molta attenzione.
Prima di compilare la formtutti i componenti del gruppo devono creare un account su GitHub. Vi consigliiamo di creare l'account con un nome che vi identifichi. GitHub sarà un vostro "biglietto" da visita virtuale per il vostro futuro accademico e lavorativo. La compilazione di tale form è obbligatoria.

ASSIGNMENT

Una volta compilata la form si deve accedere al servizio classroom di github (https://classroom.github.com). Il docente all'interno di una classroom(https://classroom.github.com/classrooms/126865705-lpodisim2023-classroom-d393f5) ha creato dei cosiddetti assignment (uno per ogni progetto). Di seguito sono elencati i link degli assignment per ogni progetto.
Scacchi: https://classroom.github.com/a/lEF68i77
Clash of Univaq: https://classroom.github.com/a/gC1gT0kg
Gonnect: https://classroom.github.com/a/Ijrzf2S_
I singoli componenti del team si devono iscrivere al relativo assignment/progetto scelto. Vi ricordiamoche uno dei componenti deve creare il team e, successivamente, gli altri componenti devono fare Join nel team.

REPOSITORY GITHUB

Una volta che il team è stato creato, in GitHub troverete un repository per team con una struttura del progetto predefinita. Lo scheletro del progetto utilizza Apache Maven che chiaramente deve essere utilizzato per il progetto. Pertanto, tale repository dovrà essere utilizzato per gestire il codice sorgente del progetto. Tale repository conterrà il sistema da consegnare e di conseguenza permetterà di valutare il progetto durante l'esame. Inoltre, il repository conterrà una cartella doc dove potete trovare il template del progetto documentazione-progetto.docx da utilizzare per documentare il vostro progetto. Vi ricordo che tale documento è obbligatorio e costituisce parte integrante del progetto. Il repository deve essere utilizzato da tutti i componenti del gruppo in modo uniformedurante lo sviluppo del progetto. Pertanto, ogni singolo componente deve effettuare un numero congruo di commit rispetto agli altri componenti del gruppo. Inoltre, i files contenuti in ogni singolo commit devono essere congrui rispetto agli altri.

NOTA importante: Qualora decidiate di abbandonare il gruppo dovete comunicarlo al docente inviando una e-mail con in copia carbone (cc) gli altri componenti del gruppo. Chiaramente per il/la rinunciatario/a dovrete ripetere il processo da capo, ovvero compilare la form e iscriversi al relativo assignment.

Criteri di valutazione

La valutazione dell'applicazione sarà condotta considerando diversi aspetti quali:

- Completezza dell'implementazione;
- Utilizzo delle convenzioni nella scrittura del codice Java;
- Copertura dei concetti fondazionali spiegati a lezione:
    - Classi,
    - Ereditarietà,
    - Polimorfismo,
    - Interfacce,
    - Eccezioni,
    - Generics,
    - Collections and Lambda calculus.
- Organizzazione dell'applicazione in package, classi, ecc.;
- Gestione degli errori meditante eccezioni;
- Usabilità del software prodotto;
- Documentazione mediante commenti delle porzioni di codice più significative;
- Progettazione e class diagram
- Documentazione in cui si dettagliano le classi, gli attributi e i metodi

Ogni progetto deve essere accompagnato da una breve relazione. Il template lo troverete all'interno del progetto associato con il repository di GitHub.

Regole comuni a tutti i progetti:

- Il codice deve essere consegnato 5 giorni prima della data di esame;
- Il codice deve essere compilabile ed eseguibile tramite comandi maven;
- Il codice deve essere sviluppato in maniera collaborativa usando git come VCS (Nota bene: si analizzeranno il numero di commit e l'autore dello stesso)
- Il codice deve essere sviluppato in modo bilanciato dagli elementi del gruppo;

- Il progetto deve prevedere il riutilizzo di una libreria nota (ad esempio quelle fornite da apache foundation):
    - Ovviamente, non è consentito usare una libreria che implementino il gioco degli scacchi o degli altri progetti proposti

Gioco degli scacchi

L'obiettivo del progetto è implementare una semplificazione del gioco degli scacchi in Java.

Requisiti

- Il progetto deve permettere tre modalità di gioco:
    1. Giocatore contro Giocatore
    2. Giocatore contro Computer (non impatterà in alcun modo la poca intelligenza del giocatore Computer)
    3. Recupero di una partita salvata.
    4. Data una lista di partite il sistema deve essere in grado di ordinarle in base ai seguenti criteri:
        a. numero di mosse effettuate nella partita;
        b. Numero complessivo di pezzi sulla scacchiera;
        c. Valore complessivo dei pezzi sulla scacchiera (Potete assegnare un peso arbitrario per ogni tipo di pezzo);

IDEA: Se realizzate con interazioni da linea di comando prendete in considerazione l'utilizzo di parametri per la selezione della modalità.

- Non è richiesta un'interfaccia grafica, ma sarà presa in considerazione la modalità di interazione con il gioco e la rappresentazione dello stato dello scacchiere e della partita in generale. Un esempio di rappresentazione della scacchiera tramite linea di comando.

```
|b_t1|b_c1|b_a1|b_q1|b_k1|b_a2|b_c2|b_t2|
|b_p1|b_p2|b_p3|b_p4|b_p5|b_p6|b_p7|b_p8|
|    |    |    |    |    |    |    |    |
|    |    |    |    |    |    |    |    |
|    |    |    |    |    |    |    |    |
|    |    |    |    |    |    |    |    |
|n_p1|n_p2|n_p3|n_p4|n_p5|n_p6|n_p7|n_p8|
|n_t1|n_c1|n_a1|n_q1|n_k1|n_a2|n_c2|n_t2|
```

- Il gioco deve loggare su un file tutte le mosse svolte durante un partita.
- Il gioco deve gestire eventuali eccezioni dovute a interazioni sbagliate (esempio mossa non valida)
- Il gioco deve permettere di interrompere una partita, salvarla su file e, in successive esecuzioni, caricarla dal file di salvataggio.
- Il sistema deve prevedere di poter annullare al massimo le ultime 5 giocate.

SEMPLIFICAZIONE DEL GIOCO:

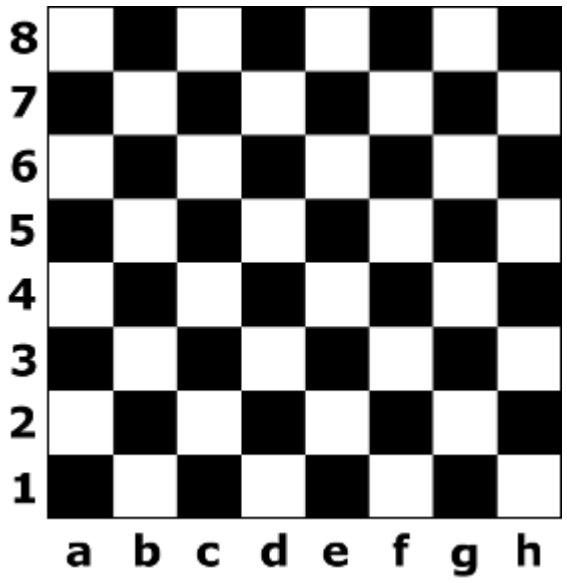
- Nessuna mossa speciale (ad esempio arrocco)
- No richiedo l'implementazione della cattura en passant.
- Fine partita come descritto di seguito.

Regole degli Scacchi

Scacchiere: La scacchiera si compone di 64 caselle uguali distribuite su otto righe e otto colonne. Le parti della scacchiera hanno nomi particolari:

- traversa - le otto righe orizzontali della scacchiera si chiamano traverse;
- fila - le otto colonne verticali della scacchiera si chiamano file;
- diagonale - la linea retta di case dello stesso colore che va da un angolo all'altro della scacchiera viene chiamata diagonale;
- centro - le quattro case situate in mezzo alla scacchiera vengono chiamate centro.

Ogni singola casa possiede un nome che permette di trascrivere le mosse della partita. Esistono vari sistemi di annotazione delle mosse, ma la «notazione algebrica» è il più diffuso ed è considerato il sistema ufficiale. In questo sistema ogni casa prende il nome dalla riga e dalla colonna in cui si trova. Le traverse (righe) sono numerate da 1 a 8 iniziando dal lato bianco della scacchiera e spostandosi verso il lato nero. Le file (colonne) sono contrassegnate da lettere minuscole da **1 a 8**, andando da sinistra verso destra secondo la prospettiva del giocatore bianco. La casa è denominata con la lettera seguita dal numero; di conseguenza la casa nella estremità inferiore sinistra è conosciuta come a1. La struttura di denominazione è visibile nel seguente schema:



Disposizione iniziale

La disposizione del pezzi sulla scacchiera è identica sia sul lato bianco che sul lato nero. Nella prima riga, partendo da entrambe le case esterne e andando verso l'interno, si posizionano la torre, il cavallo e l'alfiere. Nelle due case rimanenti si posizionano la regina (donna) nella casa del suo stesso colore e il re nella casa rimanente. Una volta terminato, i pezzi uguali si trovano gli uni di fronte agli altri. Nella seconda riga viene posizionata una fila di pedoni. L'aspetto della scacchiera completata sarà simile alla figura seguente.

La *partita* consiste di 2 *giocatori*, uno *scacchiere* ed una lista di *pezzi* schierati nello scacchiere. l gioco degli scacchi ha sei tipi di *pezzi*:

- il *pedone*,
- la *torre*,
- il *cavallo*,
- l'*alfiere*,
- la *regina* e
- il *re*

Ogni pezzo si *muove* in modo unico. Non possono muovere oltre il limite della scacchiera né tornare dall'altro lato; il bordo dello *scacchiere* è un confine che non può essere oltrepassato. Nessun pezzo, ad eccezione del cavallo, può saltare gli altri pezzi - tutte le case, tra quella in cui il pezzo inizia la sua mossa quella dove la termina, devono essere vuote. La mossa non può terminare in una casa già occupata da un pezzo dello stesso colore.

Se un pezzo termina la sua mossa in una casa occupata da un pezzo avversario, quest'ultimo viene *catturato* ed eliminato dal gioco. Tutti i pezzi possono essere catturati, ad eccezione del re.

La partita termina alla mossa precedente alla cattura del re - «scacco matto». Per catturare è necessario che, nel muovere, il pezzo attaccante finisca nella casa del pezzo avversario.

Pedone: il pedone muove solo in avanti, una casa alla volta. È l'unico pezzo che non cattura allo stesso modo in cui si muove: il pedone cattura un pezzo avversario muovendo in diagonale di una casa - non può catturare muovendo dritto davanti a sé.

Alfiere: L'alfiere si muove sulla scacchiera diagonalmente in linea retta. Può muovere di quante case si desidera, finché non raggiunge la fine della scacchiera o incontra un altro pezzo.

Torre: La torre muove in linea retta in orizzontale o verticale per un qualsiasi numero di case inoccupate, finché non arriva alla fine della scacchiera o non resta bloccata da un altro pezzo

Cavallo: Il cavallo è il pezzo più particolare della scacchiera, dal momento che possiede una flessibilità che lo rende molto forte: è l'unico pezzo che può saltare altri pezzi. Il cavallo muove di due case, in orizzontale o verticale, e poi ancora di una casa ad angolo retto: la sua mossa è a forma di «L».

Regina: La regina è considerata il pezzo più forte della scacchiera. Può muovere di un qualsiasi numero di case in linea retta - in verticale, orizzontale o diagonale. A meno che non catturi un pezzo, la regina deve spostarsi su una casa libera e non può saltare altri pezzi.

Re: Negli scacchi il re è il pezzo più importante. Se il re è intrappolato e la sua cattura è inevitabile, la partita è finita e il giocatore ha perso. Il re può muovere su qualsiasi casa adiacente, spostandosi cioè di una casella in qualsiasi direzione: in orizzontale, verticale o diagonale. Non può posizionarsi su una casa occupata da un pezzo dello stesso colore. Il re cattura lungo il suo percorso, andando ad occupare la casa del pezzo avversario. Il movimento del re possiede un limite ulteriore: non può muovere su una casa che lo renderebbe vittima di un attacco da parte di un pezzo avversario (detto «scacco»). Il re può anche essere costretto a muovere se si trova sotto attacco e l'unico modo per sventarlo è muovere lo stesso re.

Scacco: Quando il re subisce l'attacco da parte di un pezzo avversario che ne minaccia la cattura, esso viene dichiarato «sotto scacco». Il re deve uscire dallo scacco immediatamente. Esistono tre possibilità che permettono di uscire dallo scacco:

- catturare il pezzo attaccante;
- allontanare il re dall'attacco mettendolo al sicuro in una casa dove non subisca l'attacco da parte di un pezzo avversario;
- bloccare l'attacco posizionando un altro pezzo tra quello che attacca e il re (azione non possibile se l'attaccante è un cavallo o pedone).

Finali di partita:

- Scacco matto: Se il re non può sfuggire allo scacco, la posizione viene chiamata «scacco matto» e la partita termina. Il giocatore che è sotto scacco matto perde la partita; il re non viene realmente catturato ed eliminato dalla scacchiera.
- Resa: In qualsiasi momento della partita un giocatore può arrendersi (abbandonare). La partita termina e ha vinto il giocatore avversario.
- Patta: regola delle cinquanta mosse, se nelle ultime cinquanta mosse non è stato catturato alcun pezzo o non è stato mosso alcun pedone;

Clash of Univaq

Requisiti

Implementare un gioco che di carte fantasy che prende inspirazione da https://clashroyale.com/it/

- Il progetto deve permettere tre modalità di gioco:
    1. Giocatore contro Giocatore
    2. Giocatore contro Computer (non impatterà in alcun modo la poca intelligenza del giocatore Computer)
    3. Recupero di una partita salvata.
    4. Data una lista di partite il sistema deve essere in grado di ordinarle in base ai seguenti criteri:
        a. numero di mosse effettuate nella partita;
        b. Numero complessivo di pezzi sulla scacchiera;
        c. Valore complessivo dei pezzi sulla scacchiera (Potete assegnare un peso arbitrario per ogni tipo di pezzo);

IDEA: Se realizzate con interazioni da lina di comando prendete in considerazione l'utilizzo di parametri per la selezione della modalità.

- Non è richiesta un'interfaccia grafica, ma sarà presa in considerazione la modalità di interazione con il gioco e la rappresentazione dello stato dello campo di battaglia e della partita in generale.
- Il gioco deve loggare su un file tutte le mosse svolte durante un partita.
- Il gioco deve gestire eventuali eccezioni dovute a interazioni sbagliate (esempio mossa non valida)
- Il gioco deve permettere di interrompere una partita, salvarla su file e, in successive esecuzioni, caricarla dal file di salvataggio.
- Il sistema deve prevedere di poter annullare solo l'ultima mossa

Regole del gioco

Il gioco consiste nel combattimento tra 2 giocatori per distruggere il castello avversario.
La mappa di gioco consiste in 2 torri (una per player) e tre *strade* (sx, centro, e dx) che congiungono le due torre e dove schierare i personaggi.
La battaglia avviene mediante un sistema di turnazione dove il giocatore di mano può schierare alcune delle sue carte (ogni giocatore deve avere un roaster di almeno 5 tipologie di carte differenti). E' possibile schierare più carte dello stesso tipo sul terreno di gioco. La scelta della carta da giocare viene effettuata su un sottoinsieme del roster per evitare che un player selezioni sempre la stessa carta.
Ogni carta ha un costo di schieramento (Energia). Il giocatore accumula Energia extra ogni turno e può scegliere di usarla nel turno attuale o conservarla per i turni successivi. é possibile schierare carte se si ha sufficiente energia.
Le carte possono essere incantesimi o personaggi.
Gli incantesimi hanno sempre un personaggio schierato come target sia del proprio team che dell'avversario (esempio cura un personaggio schierato, blocca attacco per un turno di personaggio schierato, rendi invulnerabile per un turno un personaggio schierato, etc.)
Ogni tipologia di personaggio ha punti vita PV, punti armatura PA, danno provocato DP, Mana M ed una mossa speciale MS. Possibili categorie di personaggi:

- Tank
    - Gigante
    - paladino con scudo
- Assassino
    - assassino invisibile
    - assassino con colpo critico
    - bombarolo
- Mago
    - curatore
    - stregone di ghiaccio

La MS è eseguita all'inizio del turno se un personaggio ha sufficiente Mana (indipendentemente dal giocatore di turno). Ogni turno ogni personaggio accumula Mana extra.
Modificate a vostro piacimento le classi di carattere o estendere ulteriormente le tre tipologie con altri classi di personaggi. Inoltre, sentitevi liberi di definire le vostre mosse speciali (ricarica energia, doppio armor, attacco critico, stun, etc) (vi ricordo che ogni giocatore deve contare su almeno un roaster di 5 tipologie di personaggi)

Un turno è organizzato da tre fasi

- Schieramento: in accordo con l'energia del giocatore
- Scelta personaggi in difesa. I personaggi in difesa ottengono un doppio armor
- Scelta personaggi in attacco. Il giocatore deve definire i personaggi disposti in attacco per il turno e selezionare quale rispettivo personaggio avversario attaccare.

Così come i personaggi le torri hanno punti vita.

Una volta schierata una carta incantesimo lo eseguirà. Se viene schierato in una via strada un personaggio sarà posizionato nella rispettiva via per la torre. Una volta schierate le carte, il giocatore di turno deve scegliere per ogni personaggio schierato in posizione di attacco quale personaggio avversario attaccare (si possono attaccare solo i personaggi avversari schierati nella stessa via del personaggio scelto).
Nel caso non siano presenti personaggi in una via il personaggio potrà attaccare direttamente la torre. Se il personaggio attaccato ha meno vita del danno provocato, esso morirà. I PA del personaggio attaccante in avanzo dai PV del personaggio attaccato verranno detratti alla torre. Utilizzare l'armor del giocatore attaccato nella funzione di attacco.

Il primo giocatore che distrugge la torre avversaria avrà vinto.

Eventuali aggiunte e/o nuove regole al gioco sono possibili previa discussione con il docente.

Gonnect

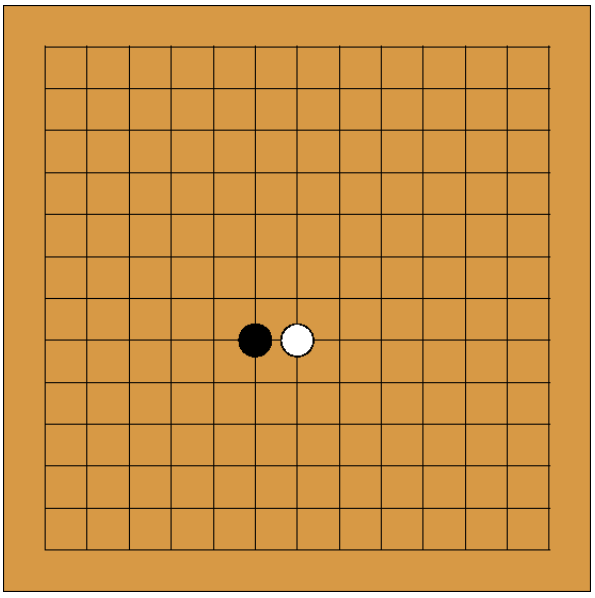
Requisiti

- Il progetto deve permettere tre modalità di gioco:
    1. Giocatore contro Giocatore
    2. Giocatore contro Computer (non impatterà in alcun modo la poca intelligenza del giocatore Computer)
    3. Recupero di una partita salvata.
    4. Comparazione partite precedenti ed ordinamento: Data una lista di partite il sistema deve essere in grado di ordinarle in base ai seguenti criteri:
        a. numero di mosse effettuate nella partita;
        b. Numero complessivo di pezzi sulla scacchiera;
        c. differenza numeri pezzi tra i due giocatori;

    IDEA: Se realizzate con interazioni da linea di comando prendete in considerazione l'utilizzo di parametri per la selezione della modalità.

- Non è richiesta un'interfaccia grafica, ma sarà presa in considerazione la modalità di interazione con il gioco e la rappresentazione dello stato dello scacchiere e della partita in generale. Un esempio di rappresentazione della scacchiera tramite linea di comando.

```
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   |   |  |X  |   |   |   | ->  |   |   |  |X  | O |   |   |  |->   |   |  |X  |O  |   |   |   |   ->
…
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
```

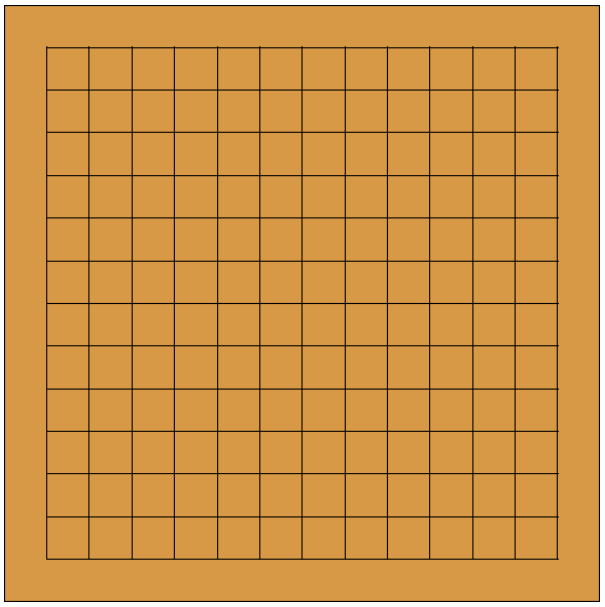


- Il gioco deve loggare su un file tutte le mosse svolte durante un partita;
- Il gioco deve gestire eventuali eccezioni dovute a interazioni sbagliate (esempio mossa non valida);

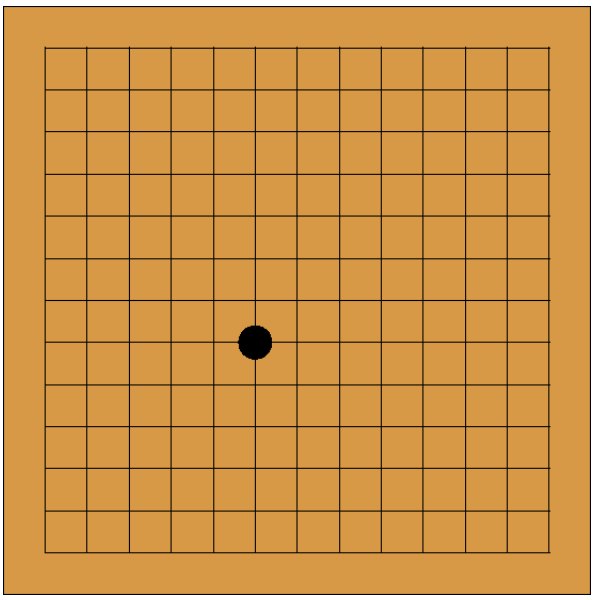
- Il gioco deve permettere di interrompere una partita, salvarla su file e, in successive esecuzioni, caricarla dal file di salvataggio;
- Il sistema deve prevedere di poter annullare le ultime tre mosse.

Regole Gonnect

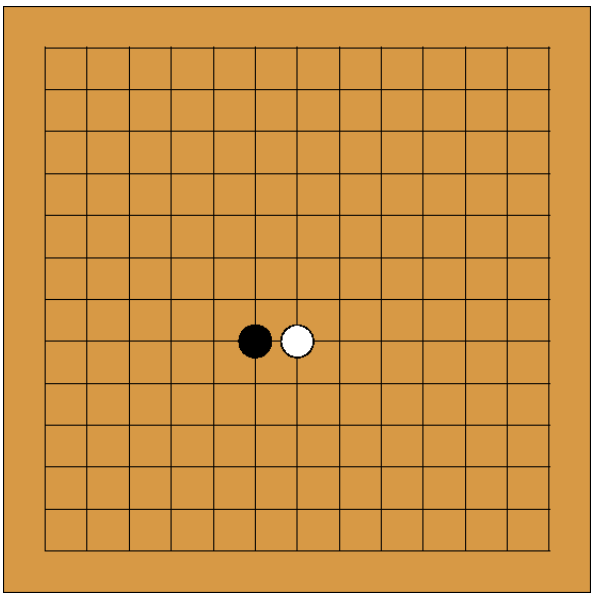
L'obiettivo del progetto è implementare il gioco "Gonnect" in Java. Questo gioco è stato inventato da João Neto. Il gioco è una combinazione di due giochi famosi: Go and Hex.
Si gioca su una tavola costituita da una griglia di righe orizzontali e verticali di dimensione 13x13:



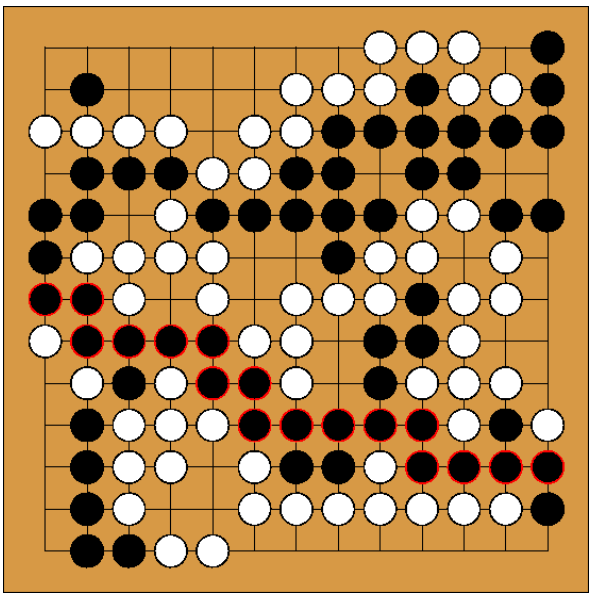
I due giocatori, alternativamente, riempiono la tavola mettendo delle pedine (una alla volta) sulle intersezioni libere. Per esempio, un inizio di partita può essere



e successivamente



Le regole del gioco sono le seguenti:
Swap Invece della sua prima mossa, il secondo giocatore può scegliere di cambiare i colori.
Catena Pedine dello stesso colore che sono adiacenti (verticalmente o orizzontalmente) formano una catena: due pedine appartengono alla stessa catena se e solamente se esiste un modo per passare dall'una all'altra camminando (verticalmente o orizzontalmente) solamente su pedine dello stesso colore.
Libertà Una pedina ha una libertà se almeno una delle sue intersezioni adiacenti è libera. Una catena ha una libertà se contiene almeno una pedina con una libertà.
Cattura Se in seguito all'inserimento di una pedina una catena si ritrova senza libertà, tutte le pedine che la compongono vengono catturate e sono rimosse della tavola.
No suicide Una mossa non può essere fatta se ha per conseguenza la cattura di una catena dello stesso colore della pedina aggiunta.
Mossa Si può fare una mossa solamente se la catena alla quale appartiene la nuova pedina ha una libertà. La libertà è calcolata dopo le eventuali catture delle catene del colore opposto alla pedina aggiunta.
Vittoria Un giocatore vince se connette con una catena due lati opposti della tavola o se l'altro giocatore si trova nell'impossibilità di fare una mossa, ovvero tutte le sue mosse formano catene senza libertà.
Per esempio, una tavola vincente è la seguente:



Il nero ha vinto: ha connesso il lato sinistro con il destro.