Progetto esame di Programmazione e Modellazione a Oggetti

Sessione estiva 2019/2020

Backgammon con interfaccia grafica

Studente: Crisan Sergiu

Matricola: 290247

Specifica del Problema

Si deve realizzare un software con interfaccia grafica che permetta a due persone di giocare a Backgammon.

Il programma dovrà permettere di compiere le azioni principali del gioco, determinare il vincitore e comunicarlo.

Le azioni principali comprendono:

a) Muovere le proprie pedine;

b) Mangiare le pedine dell'avversario;

c) Rimettere in tavola le pedine che sono state mangiate;

d) Portare fuori le pedine.

e) Lanciare i dadi;

- Gestione delle doppie

Tutte le azioni necessitano dei dovuti controlli che vanno implementati per impedire ai giocatori di fare mosse non consentite dalle regole.

L'interfaccia dovrà mostrare: la tavola, le pedine in gioco, le pedine mangiate, i dadi ancora utilizzabili, il turno del giocatore ed eventuali messaggi informativi (es. "Devi lanciare i dadi", "Ha vinto il giocatore Nero").

Studio del Problema

Analizzando il problema le criticità maggiori risultano essere le principali azioni in quanto bisogna impedire ai giocatori di effettuare mosse contro il regolamento del gioco stesso.

Analizzandole una ad una:

Reinserimento

Reinserire le pedine mangiate è possibile solo se:

- Si hanno pedine mangiate;
- Il triangolo dove bisogna inserire è: libero, occupato dal giocatore che sta reinserendo oppure occupata da solo una pedina avversaria (il che implica mangiargli la pedina);

Togliere le pedine

Togliere le pedine dalla tavola è possibile solo se:

- Non si hanno pedine mangiate;
- Tutte le pedine del giocatore sono nella propria base;
- La posizione indicata dal dado dalla quale togliere è occupata da pedine del giocatore oppure alla sinistra di questa posizione non ci sono altre pedine del giocatore;

Movimento

Muovere le pedine su backgammon è possibile solo se:

- Non ci sono pedine mangiate;
- La posizione di arrivo non è occupata con più di una pedina dall'avversario;
- Non si esce dalla tavola;

Mangiare le pedine avversarie

Mangiare le pedine dell'avversario è possibile durante le azioni di movimento o reinserimento solo se il triangolo di destinazione è occupato da una sola pedina avversaria.

Lanciare i dadi

Lanciare i dadi è possibile solo se:

- L'avversario ha esaurito le mosse possibili, dunque si cambia il turno;
- Non si ha già lanciato i dadi durante il proprio turno

Inoltre bisogna considerare che si gioca con due dadi i quali una volta tirati possono essere utilizzati solo una volta per dado per muovere una pedina. Questo è vero a meno che non sia uscito lo stesso numero su entrambi i dadi il che aumenta le volte che si possono muovere le pedine da 2 a 4.

Quando si vince?

Il vincitore è il giocatore che per primo riesce a togliere tutte le pedine dalla tavola.

Come si è scelto di affrontare le criticità?

Per risolvere le problematiche prima descritte si è optato per una prima analisi quando vengono lanciati i dadi responsabile delle seguenti:

- Controllare che siamo ancora in gioco, quindi che nessuno dei due giocatori abbia vinto;
- Controllare che ancora i dadi siano utilizzabili, quindi che il giocatore abbia effettivamente finito il turno.

Successivamente se questa prima analisi è stata passata si controlla in quale dei tre casi si trova il giocatore ovvero:

- Se ha pedine fuori;
    - Se il reinserimento è possibile;
- Se tutte le sue pedine sono nella propria base;
    - Se è possibile togliere pedine;
- Se è possibile muovere le pedine;

Come si può intuire dal primo punto di questo paragrafo bisogna inserire anche un controllo che verifica se uno dei due giocatori ha vinto.

Procedimento di analisi dei tre casi

1) *Se ha pedine fuori*: il controllo avviene verificando se ci sono o meno pedine nella posizione adibita.

   Se il reinserimento è possibile: vengono controllati i valori dei dadi, se i triangoli corrispondenti ai valori dei dadi sono liberi e/o occupati dal giocatore che deve muovere.

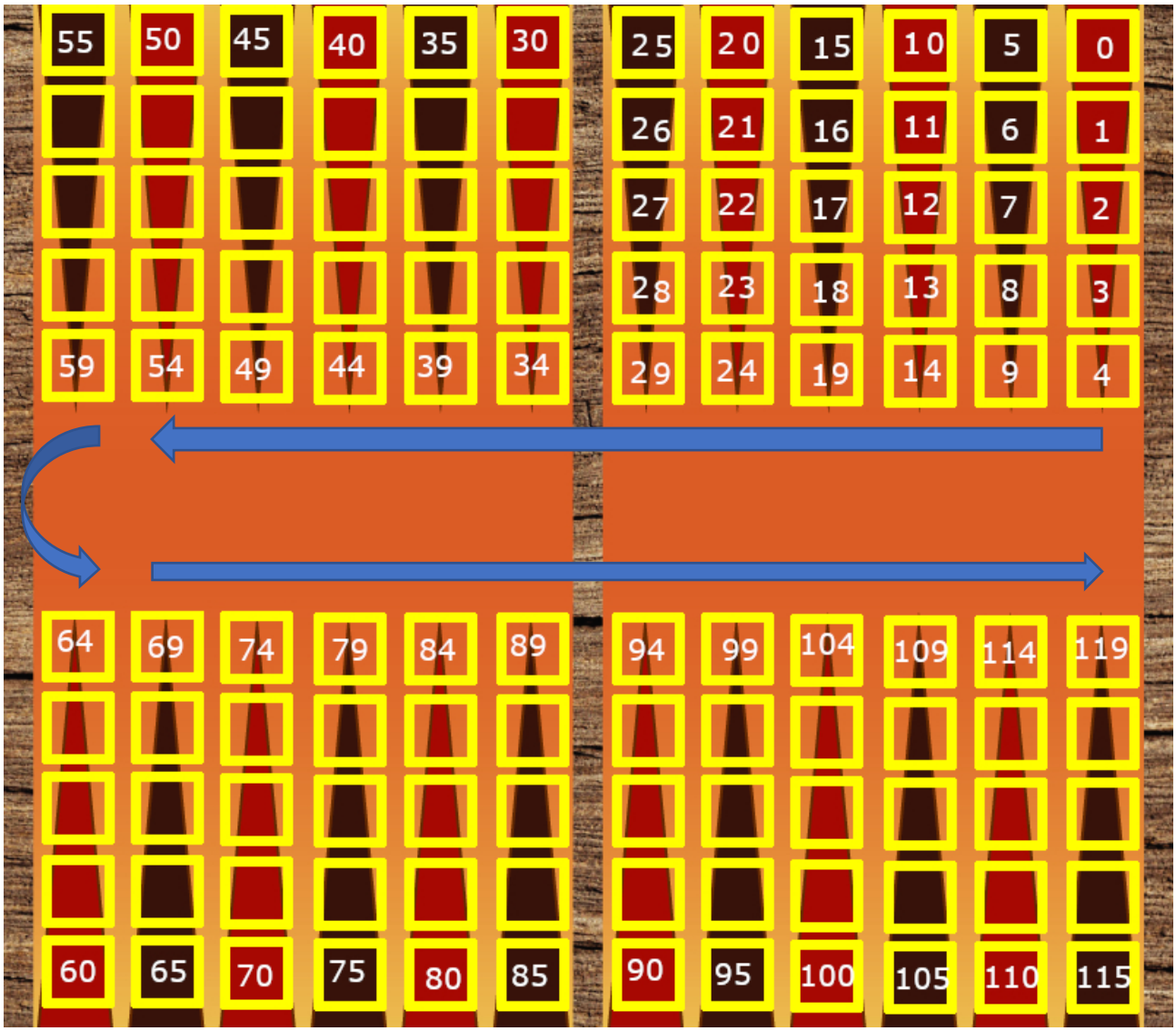
2) *Se tutte le pedine sono nella propria base*: viene tenuto il conto di tutte le pedine di ogni giocatore e si controlla se il numero di pedine totali del giocatore e il numero di pedine presenti nella sua base corrispondono. Se ciò accade al giocatore viene comunque data una scelta ovvero se vuole muovere o se voglie togliere la pedina.

   Se è possibile togliere: si controlla che ci siano pedine nei triangoli indicati dal valore dei dadi e quindi se possibile poi viene tolta.

3) *Se è possibile muovere*: dapprima si fa un controllo per verificare che ci siano pedine che possano fare i movimenti indicati dai dadi, se è possibile si prosegue altrimenti si passa il turno. Se è possibile, viene controllato che la pedina scelta inizialmente appartenga al giocatore a cui spetta il turno poi si verifica che questa pedina possa effettivamente muovere nel triangolo di arrivo.

Scelte Architetturali

Logica Posizione Pedine



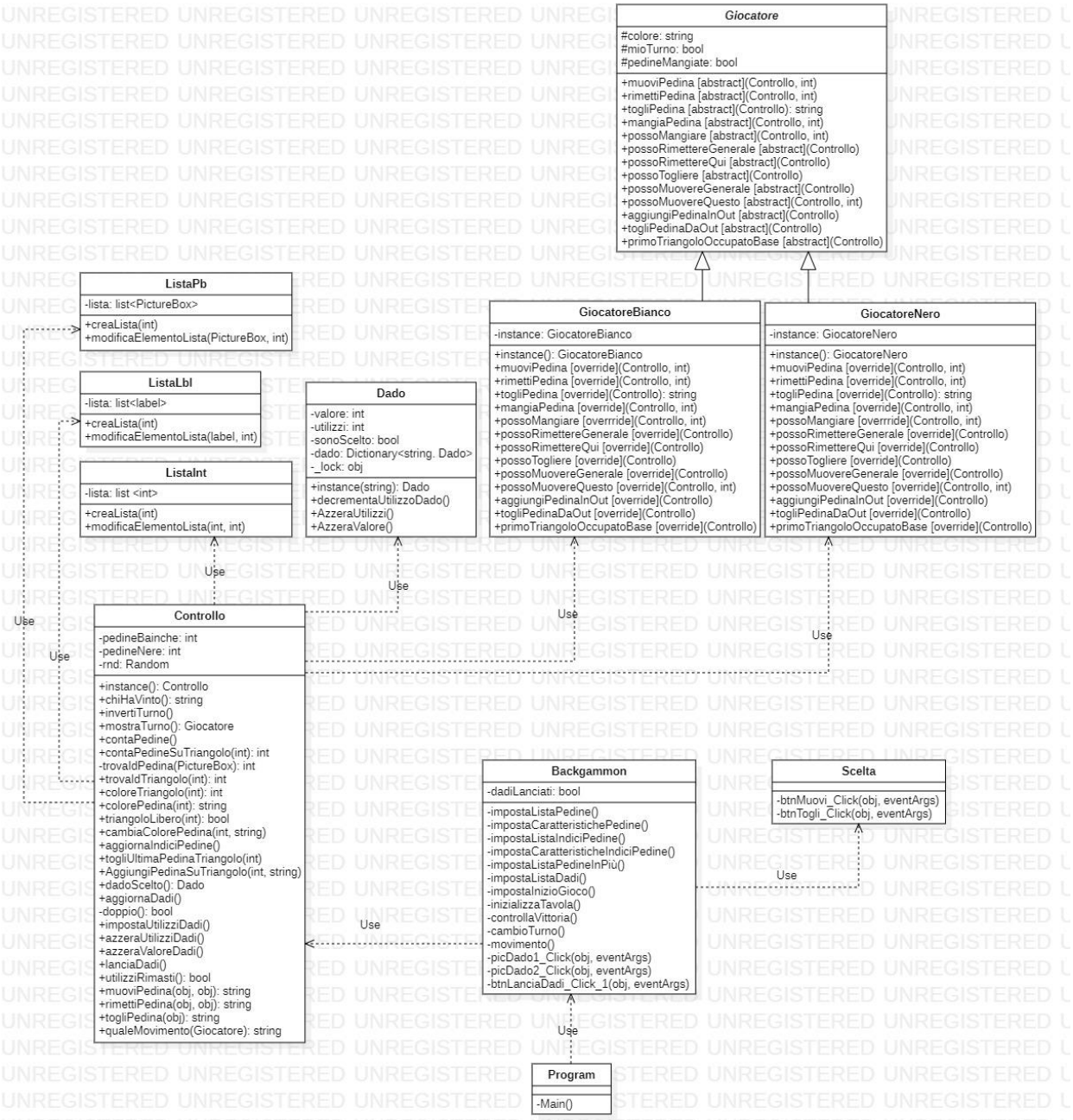
Ogni possibile posizione delle pedine ha il suo ID come indicato nel grafico, per un totale di 122 posizioni possibili.

Le prime 120 (0 – 119) sono quelle indicate nel grafico, le posizioni con id 120 e 121 rappresentano le pedine mangiate rispettivamente di colore nero e bianco. La logica della numerazione degli id avviene in due direzioni, dalla base del triangolo (valore minore) alla punta (valore maggiore) e in senso antiorario da in alto a destra nel tabellone (minimo) a in basso a destra (massimo).

Inoltre per praticità si farà riferimento ad ogni triangolo con l'id corrispondente alla pedina alla base del triangolo.

Se su un triangolo dovessero esserci più di 5 pedine quelle in eccesso verranno notificate con un numero sopra la quinta pedina, lo stesso vale per le pedine mangiate se dovessero essere più di 1.

Schema UML



Design Patterns:

Le classi: Controllo, GiocatoreBianco e GiocatoreNero utilizzano tutte il singleton in quanto il programma necessità di una sola istanza di queste classi.

La classe Dado utilizza il multiton. Il gioco ha bisogno di 2 dadi ognuno con le proprie proprietà da non sovrascrivere.

Per la natura del linguaggio tutte le parti clickabili utilizzano l'observer pattern in quanto si utilizza il meccanismo dei delegati e degli eventi.

Use Case con relativo schema UML

Use case

Lancio dadi

Id

LD

Actor

Giocatore (G)

Preconditions

No utilizzi rimasti o inizio gioco

Basic course of events

- G preme il pulsante LANCIA DADI
- Il valore dei dadi viene scelto randomicamente
- Le picturebox dei dadi cambiano

Postconditions

- Le picturebox dei dadi rappresentano il valore
- Gli utilizzi dei dadi vengono aggiornati

Alternative paths

Non ci sono alternative

Use case

Movimento Bianco

Id

MovB

Actor

Giocatore Bianco (GB)

Preconditions

- Deve essere il turno di GB
- Deve muovere una pedina bianca
- Il triangolo di arrivo deve essere libero oppure occupato da GB

- Deve aver lanciato i dadi

- Non deve avere pedine mangiate

Basic course of events

- GB clicca sul dado corrispondente al numero di posizioni che vuole muovere in avanti

- GB clicca sulla pedina che vuole spostare

Postconditions

La pedina viene spostata di tanti triangoli quanti ne indicava il dado

Alternative paths

In caso di doppio non GB non deve specificare il dado se vuole utilizzarlo nuovamente

---

Use case

Bianco Mangia Pedina

Id

BMP

Actor

Giocatore Bianco (GB), Giocatore Nero (GN)

Preconditions

- Deve essere il turno di GB

- Deve muovere una pedina bianca

- Il triangolo di arrivo deve essere occupato da una sola pedina di GN

- Deve aver lanciato i dadi

Basic course of events

- GB clicca sul dado corrispondente al numero di posizioni che vuole muovere in avanti

- GB clicca sulla pedina che vuole spostare

- La pedina di GN viene mangiata

Postconditions

- La pedina di GB viene spostata di tanti triangoli quanti ne indicava il dado

- La pedina di GN viene messa fuori dal tabellone

Alternative paths

GB reintroduce una pedina mangiata su un triangolo occupato da una sola pedina di GN