INGEGNERIA DEL SOFTWARE:

MONOPOLY

Documentazione del progetto “Elaborato 2014/2015”

Università degli Studi di Brescia

Dipartimento di Ingengeria dell'Informazione

Via Branze 38

25231- Brescia

Relazione dell'elaborato, parte I

Autori

*Nome: Falleti Davide*

*Matricola: 85973*

*Nome: Mele Fabrizio*

*Matricola: 85971*

*Nome: Cordioli Francesco*

*Matricola: 86538*

Sommario

1. Introduzione...................................................................4
2. Strumenti applicativi utilizzati........................................4
3. Modello di processo.......................................................4
4. Manuale d’uso................................................................5
5. Release 1.........................................................................9

5.1 Diagramma UML delle classi.................................9

5.2 Casi d'uso.............................................................10

1. Release 2........................................................................12

6.1 Diagramma UML delle classi................................12

6.2 Casi d'uso.............................................................13

1. Release 3........................................................................15

7.1 Diagramma UML delle classi................................15

7.2 Casi d'uso.............................................................16

1. Release 4........................................................................19

8.1 Diagramma UML delle classi................................19

8.2 Casi d'uso.............................................................20

1. Release 5........................................................................23

9.1 Diagramma UML delle classi................................23

9.2 Casi d'uso.............................................................24

1. Testing.........................................................................27

Ingegneria del Software

Monopoly

Introduzione

Il progetto ha l'obbiettivo di realizzare tramite un processo di sviluppo incrementale un'applicazione software che consenta di giocare a Monopoly.

Strumenti applicativi utilizzati

Linguaggio di programmazione: Java

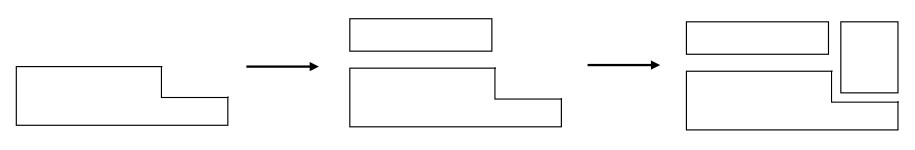
Ambiente di sviluppo: Eclipse

Strumenti di condivisione: GitHub,Sourcetree

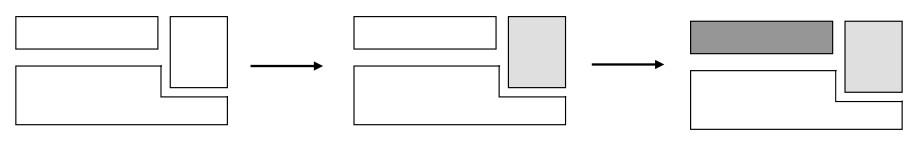
Modello di processo

Il modello di processo adottato per lo sviluppo del software è stato il modello Incrementale/Iterativo. Questo modello consente di creare più versioni del software,ognuna delle quali integra un maggior numero di requisiti rispetto alla versione precedente

Le varie fasi possono essere rappresentate da questo modello:

Approccio incrementale:

Approccio Iterativo:



Manuale d’uso

Installazione

L’applicazione è stata scritta nel linguaggio di programmazione OO Java, quindi è eseguibile suqualsiasi calcolatore che abbia installato JRE (Java Runtime Enviroment).

Esecuzione

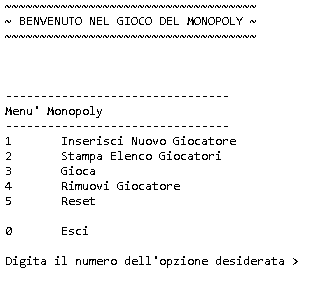
Per l’avvio dell’applicazione (.jar) eseguire i seguenti passaggi:

Aprire il Terminale di sistema ( Shell di Unix o Prompt di Windows, a seconda del Sistema Operativo)

* Portarsi nella cartella in cui è presente il file **Progetto\_Exe.jar**
* Eseguire il file .jar mediante il comando “java –jar Progetto\_Exe.jar”

A questo punto, se non vengono visualizzati errori a causa di una errata versione di JRE, dovrebbe presentarsi questa situazione

Comandi



|  |  |
| --- | --- |
| **Nome** | Inserimento nuovo giocatore |
| **Numero** | 1 |
| **Funzione** | L’applicazione permette di inserire il nome di un nuovo giocatore. |
| **Immagini** |  |
| **Note** | Se è già stato inserito il numero massimo di giocatori, il programma impedisce l'inserimento di un nuovo giocatore e lo comunica all'utente. |

|  |  |
| --- | --- |
| **Nome** | Stampa elenco giocatori |
| **Numero** | 2 |
| **Funzione** | L’applicazione permette di visualizzare l'elenco di tutti i giocatori che sono stati inseriti.  caricato!il!modello!nel!sistema! |
| **Immagini** | ! |
| **Note** | Se non ci sono giocatori inseriti, il programma lo comunica all'utente. |

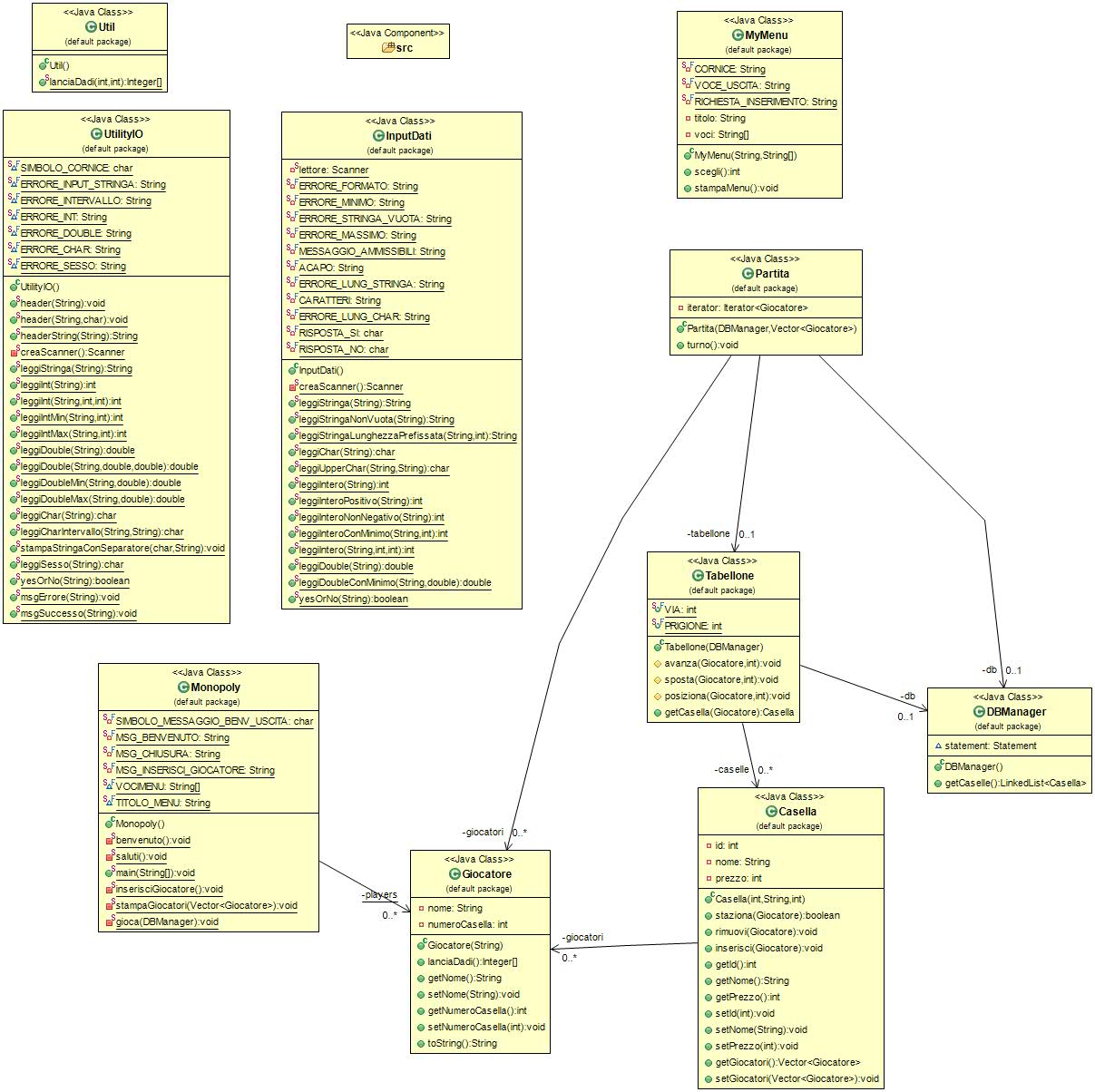
|  |  |
| --- | --- |
| **Nome** | Gioca |
| **Numero** | 3 |
| **Funzione** | L’applicazione permette di giocare una partita a Monopoly, permettendo l'avanzamento dei turni tramite la pressione del tasto invio |
| **Immagini** | !    !  ! |
| **Immagini** |  |

|  |  |
| --- | --- |
| **Nome** | Rimuovi giocatore |
| **Numero** | 4 |
| **Funzione** | L’applicazione permette di rimuovere un giocatore, presentando la lista dei giocatori precedentemente inseriti |
| **Immagini** | !    !  ! |
| **Note** | I valori da inserire possono variare rispetto a quelli presentati nell’immagine, a seconda delle opzioni disponibili. |

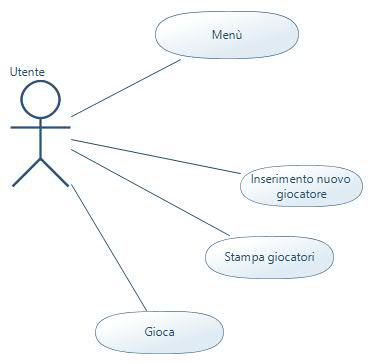
|  |  |
| --- | --- |
| **Nome** | Esci |
| **Numero** | 0 |
| **Funzione** | L’applicazione si chiude.  caricato!il!modello!nel!sistema! |
| **Immagini** | !    !  ! |
| **Note** | Per la riesecuzione del programma seguire nuovamente la procedura di esecuzione. |

Release 1

**Diagramma UML delle classi**



**Diagramma UML Casi d'uso**



Casi d'uso

|  |  |
| --- | --- |
| **Nome** | Menu Principale |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’applicazione presenta il menu di scelta  2. L’utente sceglie l’azione da svolgere  3. L’utente ha scelto di uscire dall’applicazione  4. L'applicazione viene terminata. |
| **Scenario Alternativo** | 3.a L’utente ha scelto un’operazione da svolgere  L’applicazione svolge l’operazione selezionata, torna al punto 1! |
| **Scenario Alternativo** | 3.a L’utente digita un input errato  Viene visualizzato un messaggio d’errore, Torna al punto 1! |

**Gestione del menu':**

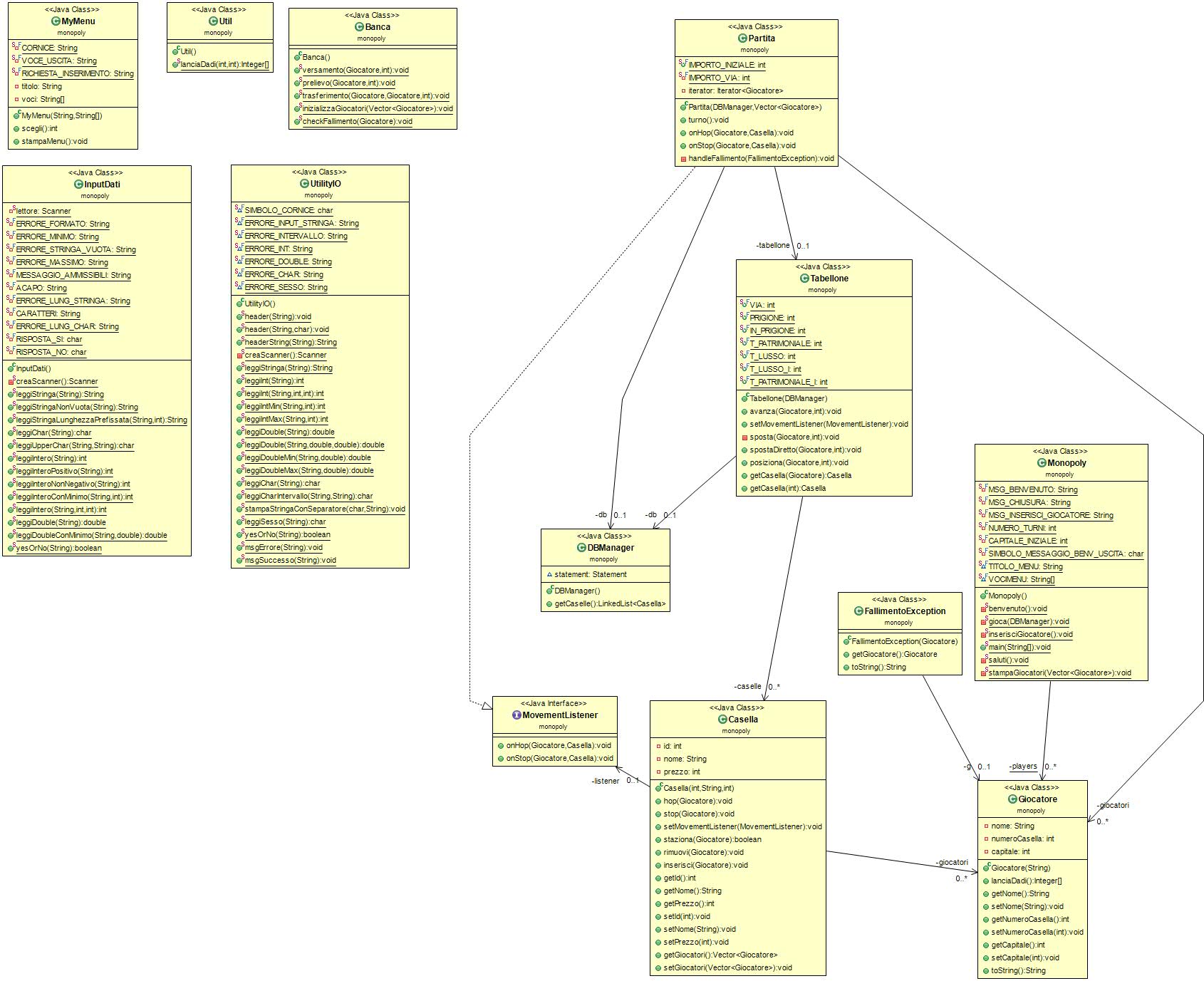
|  |  |
| --- | --- |
| **Nome** | Inserimento nuovo giocatore |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di inserire un nuovo giocatore 2. Il sistema fa inserire all’utente il nome del giocatore 3. Il sistema controlla che non ci siano più di 6 giocatori   Postcondizione: i nomi dei giocatori vengono memorizzati nel sistema. |
| **Scenario Alternativo** | 1.a L’utente decide di inserire un nuovo giocatore  Torna al punto 2 |
| **Scenario Alternativo** | 3.a L’utente inserisce più di 6 giocatori  Il sistema rileva che ci sono più di 6 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

|  |  |
| --- | --- |
| **Nome** | Stampa giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di stampare a video la lista dei giocatori 2. Il sistema stampa la lista dei giocatori inseriti |

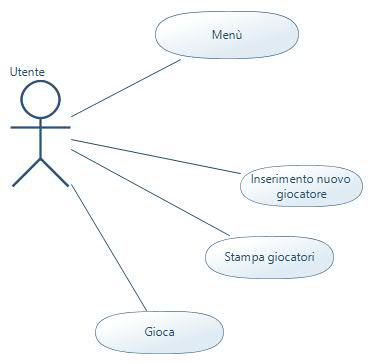
|  |  |
| --- | --- |
| **Nome** | Gioca |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di iniziare la partita 2. Il sistema controlla che i giocatori siano più di 2 3. Il sistema svolge la partita 4. Il sistema termina la partita e stampa a video tutti i turni con i relativi lanci di dadi dei giocatori e i loro movimenti |
| **Scenario Alternativo** | 2.a L’utente inserisce meno di 2 giocatori  Il sistema rileva che ci sono meno di 2 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

Release 2

**Diagramma UML delle classi**



**Diagramma UML Casi d'uso**



Casi d'uso

|  |  |
| --- | --- |
| **Nome** | Menu Principale |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’applicazione presenta il menu di scelta  2. L’utente sceglie l’azione da svolgere  3. L’utente ha scelto di uscire dall’applicazione  L’applicazione viene terminata. |
| **Scenario Alternativo** | 3.a L’utente ha scelto un’operazione da svolgere  L’applicazione svolge l’operazione selezionata Torna al punto 1 |
| **Scenario Alternativo** | 3.a L’utente digita un input errato  Viene visualizzato un messaggio d’errore, torna al punto 1 |

**Gestione del menù:**

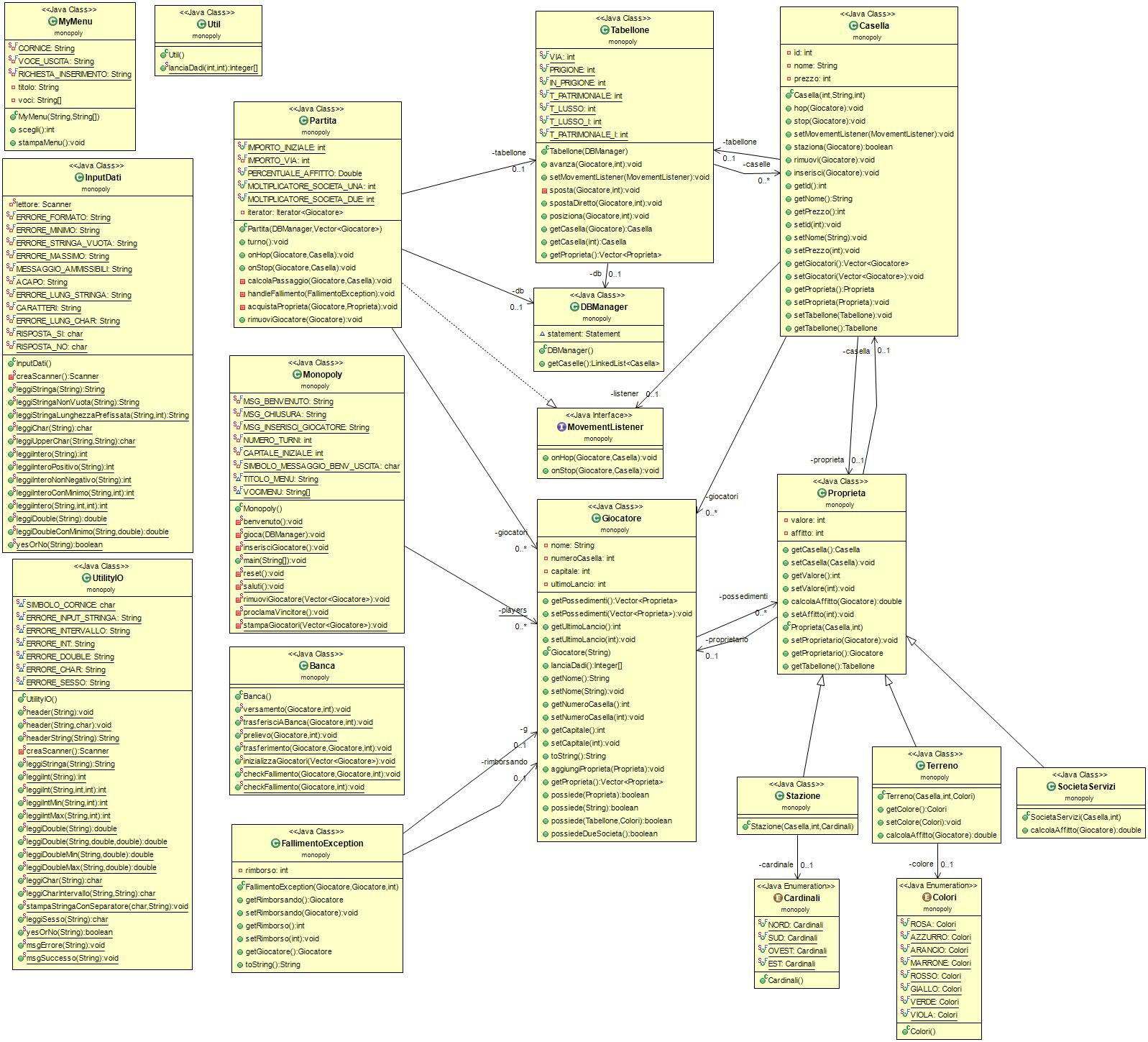
|  |  |
| --- | --- |
| **Nome** | Inserimento nuovo giocatore |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di inserire un nuovo giocatore 2. Il sistema fa inserire all’utente il nome del giocatore 3. Il sistema controlla che non ci siano più di 6 giocatori   Postcondizione: i nomi dei giocatori vengono memorizzati nel sistema. |
| **Scenario Alternativo** | 1.a L’utente decide di inserire un nuovo giocatore  Torna al punto 2 |
| **Scenario Alternativo** | 3.a L’utente inserisce più di 6 giocatori  Il sistema rileva che ci sono più di 6 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

|  |  |
| --- | --- |
| **Nome** | Stampa giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di stampare a video la lista dei giocatori 2. Il sistema stampa la lista dei giocatori inseriti e il relativo capitale |

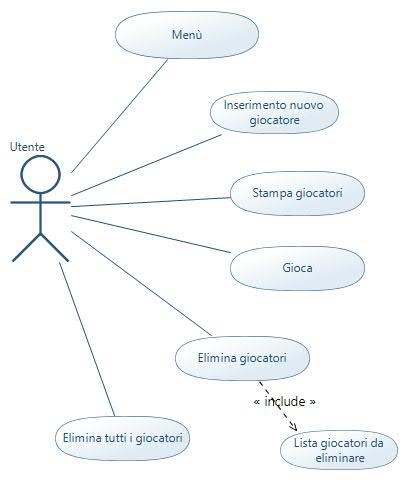
|  |  |
| --- | --- |
| **Nome** | Gioca |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di iniziare la partita 2. Il sistema controlla che i giocatori siano più di 2 3. Il sistema svolge la partita 4. Il sistema termina la partita e stampa a video tutti i turni con i relativi lanci di dadi dei giocatori, i loro movimenti e i loro capitali attuali 5. Il sistema dichiara il vincitore |
| **Scenario Alternativo** | 2.a L’utente inserisce meno di 2 giocatori  Il sistema rileva che ci sono meno di 2 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

Release 3

**Diagramma UML delle classi**



**Diagramma UML casi d'uso**



**Casi d'uso**

|  |  |
| --- | --- |
| **Nome** | Menu Principale |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’applicazione presenta il menu di scelta  2. L’utente sceglie l’azione da svolgere  3. L’utente ha scelto di uscire dall’applicazione  4. L’applicazione!viene!terminata! FINE! |
| **Scenario Alternativo** | 3.a L’utente ha scelto un’operazione da svolgere  L’applicazione svolge l’operazione selezionata, torna al punto 1! |
| **Scenario Alternativo** | 3.a L’utente digita un input errato  Viene visualizzato un messaggio d’errore Torna al punto 1! |

**Gestione del menù:**

|  |  |
| --- | --- |
| **Nome** | Inserimento nuovo giocatore |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di inserire un nuovo giocatore 2. Il sistema fa inserire all’utente il nome del giocatore 3. Il sistema controlla che non ci siano più di 6 giocatori   Postcondizione: i nomi dei giocatori vengono memorizzati nel sistema. |
| **Scenario Alternativo** | 1.a L’utente decide di inserire un nuovo giocatore  Torna al punto 2 |
| **Scenario Alternativo** | 3.a L’utente inserisce più di 6 giocatori  Il sistema rileva che ci sono più di 6 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

|  |  |
| --- | --- |
| **Nome** | Stampa giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di stampare a video la lista dei giocatori  2. Il sistema stampa la lista dei giocatori inseriti |

|  |  |
| --- | --- |
| **Nome** | Elimina giocatori |
| **Attore** | Utente |
| **Scenario Principale** | * + - 1. L’utente sceglie di eliminare un giocatore       2. <<include>>” Lista dei giocatori da eliminare”       3. Il sistema elimina il giocatore scelto dall’utente |

|  |  |
| --- | --- |
| **Nome** | Elimina tutti i giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di eliminare tutti i giocatori 2. Il sistema elimina tutti i giocatori inseriti |

|  |  |
| --- | --- |
| **Nome** | Gioca |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di iniziare la partita  2. Il sistema controlla che i giocatori siano più di 2  3. Il sistema svolge la partita  4. Il sistema termina la partita e stampa a video tutti i turni con i relativi lanci di dadi dei giocatori, le proprietà acquistate, i loro movimenti, i capitali attuali.  5. Il sistema dichiara il fallimento di un giocatore  6. Il sistema dichiara il vincitore  Il sistema rileva che un giocatore non ha capitale sufficiente a pagare l’affitto all’avversario e lo dichiara fallito facendolo uscire dalla partita. |
| **Scenario Alternativo** | 2.a L’utente inserisce meno di 2 giocatori  Il sistema rileva che ci sono meno di 2 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

Release 4

Diagramma UML delle classi

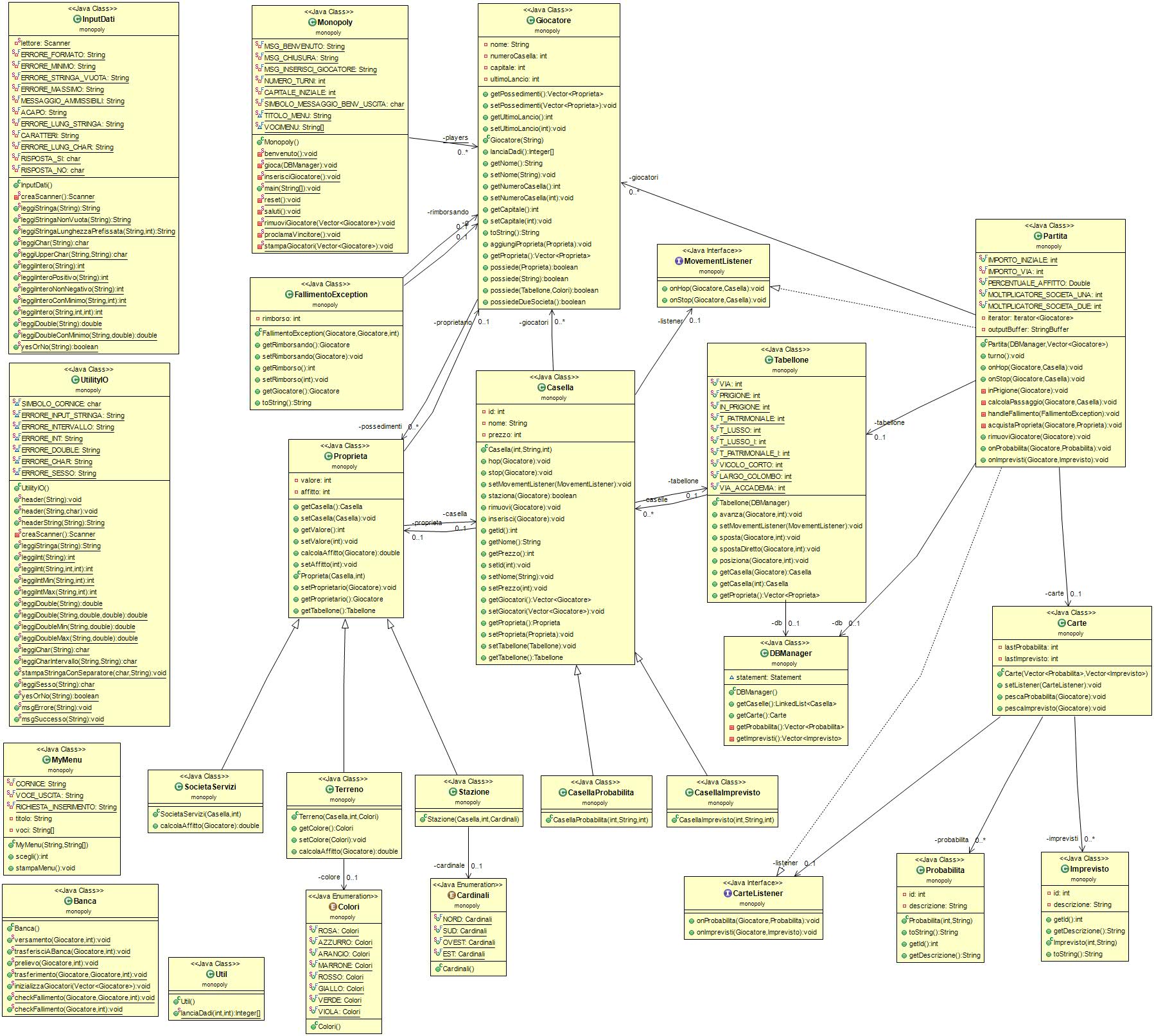
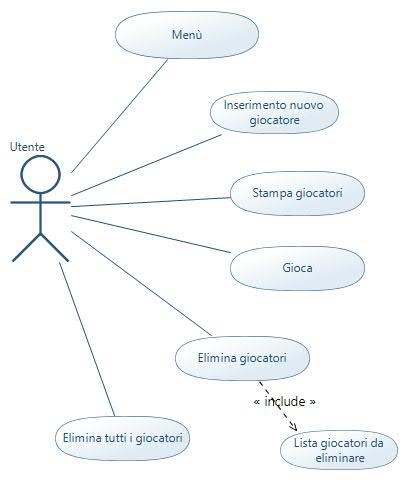


Diagramma UML casi d'uso



Casi d'uso

|  |  |
| --- | --- |
| **Nome** | Menu Principale |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’applicazione presenta il menu di scelta  2. L’utente sceglie l’azione da svolgere  3. L’utente ha scelto di uscire dall’applicazione  4. L’applicazione!viene!terminata! FINE! |
| **Scenario Alternativo** | 3.a L’utente ha scelto un’operazione da svolgere  L’applicazione svolge l’operazione selezionata, torna al punto 1! |
| **Scenario Alternativo** | 3.a L’utente digita un input errato  Viene visualizzato un messaggio d’errore Torna al punto 1! |

***Gestione del menu':***

|  |  |
| --- | --- |
| **Nome** | Inserimento nuovo giocatore |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di inserire un nuovo giocatore   2. Il sistema fa inserire all’utente il nome del giocatore  3. Il sistema controlla che non ci siano più di 6 giocatori  Postcondizione: i nomi dei giocatori vengono memorizzati nel sistema. |
| **Scenario Alternativo** | 1.a L’utente decide di inserire un nuovo giocatore  Torna al punto 2 |
| **Scenario Alternativo** | 3.a L’utente inserisce più di 6 giocatori  Il sistema rileva che ci sono più di 6 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

|  |  |
| --- | --- |
| **Nome** | Stampa giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di stampare a video la lista dei giocatori  2. Il sistema stampa la lista dei giocatori inseriti |

|  |  |
| --- | --- |
| **Nome** | Elimina giocatori |
| **Attore** | Utente |
| **Scenario Principale** | * + - 1. L’utente sceglie di eliminare un giocatore       2. <<include>>” Lista dei giocatori da eliminare”       3. Il sistema elimina il giocatore scelto dall’utente |

|  |  |
| --- | --- |
| **Nome** | Elimina tutti i giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di eliminare tutti i giocatori 2. Il sistema elimina tutti i giocatori inseriti |

|  |  |
| --- | --- |
| **Nome** | Gioca |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di iniziare la partita  2. Il sistema controlla che i giocatori siano più di 2  3. Il sistema svolge la partita  4. Il sistema termina la partita e stampa a video tutti i turni con i relativi lanci di dadi dei giocatori, le proprietà acquistate, i loro movimenti, i capitali attuali.  5. Il sistema dichiara il fallimento di un giocatore  6. Il sistema dichiara il vincitore  Il sistema rileva che un giocatore non ha capitale sufficiente a pagare l’affitto all’avversario e lo dichiara fallito facendolo uscire dalla partita. |
| **Scenario Alternativo** | 2.a L’utente inserisce meno di 2 giocatori  Il sistema rileva che ci sono meno di 2 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

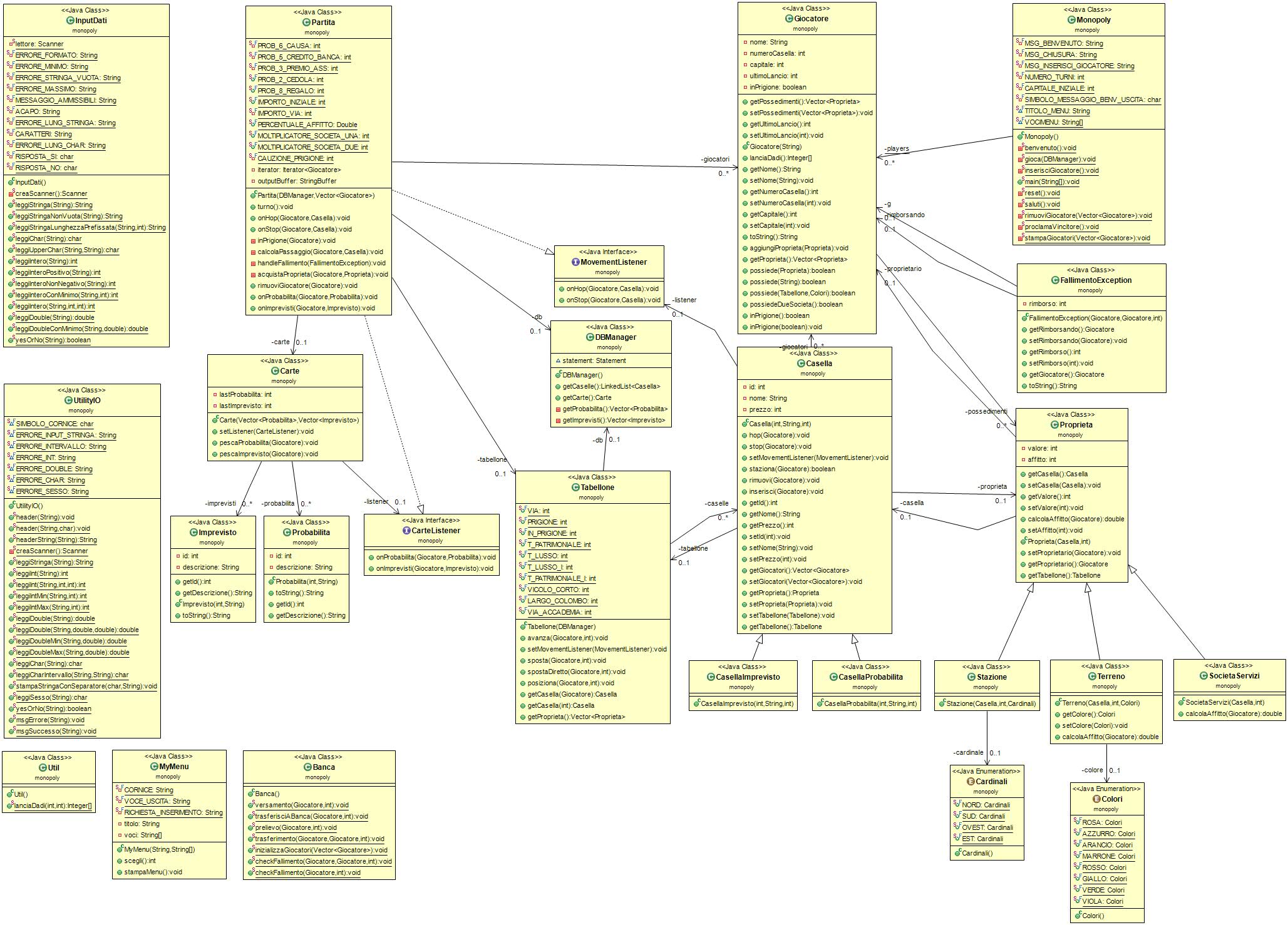
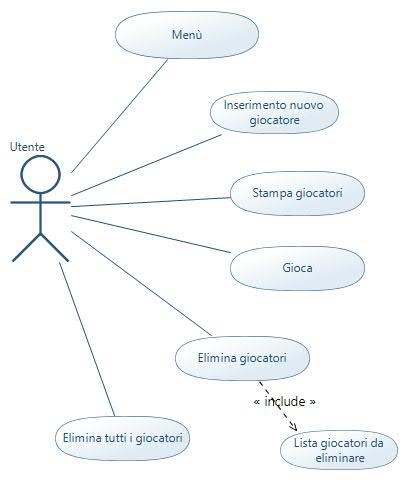
Release 5

Diagramma UML delle classi

Diagramma UML casi d'uso



Casi d'uso

|  |  |
| --- | --- |
| **Nome** | Menu Principale |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’applicazione presenta il menu di scelta  2. L’utente sceglie l’azione da svolgere  3. L’utente ha scelto di uscire dall’applicazione  4. L’applicazione!viene!terminata! FINE! |
| **Scenario Alternativo** | 3.a L’utente ha scelto un’operazione da svolgere  L’applicazione svolge l’operazione selezionata, torna al punto 1! |
| **Scenario Alternativo** | 3.a L’utente digita un input errato  Viene visualizzato un messaggio d’errore Torna al punto 1! |

***Gestione del menu':***

|  |  |
| --- | --- |
| **Nome** | Inserimento nuovo giocatore |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di inserire un nuovo giocatore  2. Il sistema fa inserire all’utente il nome del giocatore  3. Il sistema controlla che non ci siano più di 6 giocatori  Postcondizione: i nomi dei giocatori vengono memorizzati nel sistema. |
| **Scenario Alternativo** | 1.a L’utente decide di inserire un nuovo giocatore  Torna al punto 2 |
| **Scenario Alternativo** | 3.a L’utente inserisce più di 6 giocatori  Il sistema rileva che ci sono più di 6 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

|  |  |
| --- | --- |
| **Nome** | Stampa giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di stampare a video la lista dei giocatori  2. Il sistema stampa la lista dei giocatori inseriti |

|  |  |
| --- | --- |
| **Nome** | Elimina giocatori |
| **Attore** | Utente |
| **Scenario Principale** | * + - 1. L’utente sceglie di eliminare un giocatore       2. <<include>>” Lista dei giocatori da eliminare”       3. Il sistema elimina il giocatore scelto dall’utente |

|  |  |
| --- | --- |
| **Nome** | Elimina tutti i giocatori |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di eliminare tutti i giocatori 2. Il sistema elimina tutti i giocatori inseriti |

|  |  |
| --- | --- |
| **Nome** | Gioca |
| **Attore** | Utente |
| **Scenario Principale** | 1. L’utente sceglie di iniziare la partita  2. Il sistema controlla che i giocatori siano più di 2  3. Il sistema svolge la partita  4. Il sistema termina la partita e stampa a video tutti i turni con i relativi lanci di dadi dei giocatori, le proprietà acquistate, i loro movimenti, i capitali attuali.  5. Il sistema dichiara il fallimento di un giocatore  6. Il sistema dichiara il vincitore  Il sistema rileva che un giocatore non ha capitale sufficiente a pagare l’affitto all’avversario e lo dichiara fallito facendolo uscire dalla partita. |
| **Scenario Alternativo** | 2.a L’utente inserisce meno di 2 giocatori  Il sistema rileva che ci sono meno di 2 giocatori e stampa a video un messaggio d’errore  Torna al menù principale |

Testing

|  |  |
| --- | --- |
| **Test Suite** | 1 |
| **Software Testato** | Menu Principale. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Risultato** | Il Menu Principale risponde correttamente ai comandi dell’utente. |
| **Autore** | Cordioli Francesco. |
| **Data** | 09/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Ambiente** | *Driver* Main *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | 0 |
| **Uscite attese** | Uscita dal programma con relativo messaggio di uscita. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il menu principale risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Ambiente** | *Driver* Main *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | 1 |
| **Uscite attese** | Inserimento e salvataggio di un nuovo giocatore. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il menu principale risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 3 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Ambiente** | *Driver* Main  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | 2 |
| **Uscite attese** | Visualizzazione dei giocatori inseriti dall’utente. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il menu principale risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 4 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Ambiente** | *Driver* Main  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | 3 |
| **Uscite attese** | Visualizzazione della partita giocata con il numero della faccia dei dadi tirati e spostamento del giocatore. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il menu principale risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 5 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Ambiente** | *Driver:*  Main  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | 4 |
| **Uscite attese** | Messaggio di errore e ritorno al menu principale. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | E’ stato dato in ingresso un’opzione del menù che non esiste. |

|  |  |
| --- | --- |
| **Test Suite** | 2 |
| **Software Testato** | Inserimento dei giocatori. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto inserimento dei giocatori e il numero massimo di essi. |
| **Risultato** | L’inserimento risponde correttamente ai comandi dell’utente. |
| **Autore** | Cordioli Francesco. |
| **Data** | 09/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare il corretto inserimento dei dati. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | “Dab”. |
| **Uscite attese** | Salvataggio del nome. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Se vengono inseriti più di 6 giocatori viene stampato a video un messaggio che informa l’utente che è stato raggiunto il numero massimo di giocatori. |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare il corretto inserimento dei dati |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Messaggio d’errore. |
| **Uscite effettive** | Salvataggio del nome. |
| **Risultato'** | FAIL!  Il programma non risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Manca il controllo sul mancato inserimento del nome del giocatore. |

|  |  |
| --- | --- |
| **Test Case** | 3 |
| **Obiettivo** | Verificare il corretto inserimento dei dati |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Messaggio d’errore. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Sistemato il controllo sull’inserimento del nome. |

|  |  |
| --- | --- |
| **Test Suite** | 3 |
| **Software Testato** | Visualizzazione dei giocatori. |
| **Versione** | 1 |
| **Obiettivo** | Verificare la corretta visualizzazione della lista dei giocatori inseriti dall’utente. |
| **Risultato** | La visualizzazione risponde correttamente ai comandi dell’utente. |
| **Autore** | Cordioli Francesco. |
| **Data** | 09/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare la corretta visualizzazione della lista dei giocatori. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Lista dei nomi. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Suite** | 4 |
| **Software Testato** | Gioco-Partita. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del gioco. |
| **Risultato** | Il gioco funziona come da consegna. |
| **Autore** | Cordioli Francesco. |
| **Data** | 09/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del gioco. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Lista di 20 turni con varie descrizioni di dadi e posizioni aggiornate. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Il programma da messaggio di errore se sono stati inseriti meno di 2 giocatori. |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare che tutti i giocatori inizino la partita dalla casella VIA! |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | I giocatori si trovano tutti sulla casella VIA! |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 3 |
| **Obiettivo** | Verificare che dopo 3 volte che il giocatore ottiene lo stesso punteggio per entrambi i dadi finisce in prigione. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore dopo 3 tiri uguali finisce in prigione. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 4 |
| **Obiettivo** | Verificare che una volta che il giocatore arriva alla casella 40 riparta dalla casella VIA! |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Alla conclusione di ogni giro il conteggio della posizione del giocatore riparte da 0. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |

|  |  |
| --- | --- |
| **Test Case** | 5 |
| **Obiettivo** | Verificare che una partita termina dopo 20 turni. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Termine della partita. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Suite** | 5 |
| **Software Testato** | Gioco-partita con l’interazione della banca. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Risultato** | Il risultato finale corrisponde alla consegna data. |
| **Autore** | Cordioli Francesco. |
| **Data** | 10/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare che ad ogni giocatore sia assegnata la cifra di 5000€. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Ogni giocatore ha un capitale di 5000€. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare che ad ogni passaggio dal via il giocatore riceva 500€. |
| **Ambiente** | *Driver:*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Ogni utente riceve 500€ dalla banca. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 3 |
| **Obiettivo** | Verificare che dalla casella 30 (“IN PRIGIONE”) venga spostato alla casella 10 (“PRIGIONE/TRANSITO”) senza ricevere i 500€ dalla banca per il passaggio dal via. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Passaggio dalla casella 30 alla casella 10 e i soldi devono rimanere gli stessi. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 4 |
| **Obiettivo** | Verificare il pagamento di 250€ alla banca per aver sostato sulla casella TASSA PATRIMONIALE. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore ha 250€ in meno. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 5 |
| **Obiettivo** | Verificare il pagamento di 10€ alla banca per aver sostato sulla casella TASSA DEL LUSSO. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore ha 10€ in meno. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 6 |
| **Obiettivo** | Verificare il fallimento di un giocatore e l'uscita di esso dalla partita. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore esce dalla partita. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 7 |
| **Obiettivo** | Verificare il vincitore della partita giocata. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Messaggio a video con il nome del vincitore. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Il giocatore ha vinto perché o tutti gli altri giocatori hanno fallito oppure è quello più ricco. |

|  |  |
| --- | --- |
| **Test Suite** | 6 |
| **Software Testato** | Menu Principale. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento delle due voci aggiunte al menù principale. |
| **Risultato** | Il Menu Principale risponde correttamente ai comandi dell’utente. |
| **Autore** | Falleti Davide. |
| **Data** | 20/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Ambiente** | *Driver* Main *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | 4 |
| **Uscite attese** | Visualizzazione di una lista di giocatori per l’eliminazione. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il menu principale risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Visualizzo una lista di giocatori da eliminare; per poter scegliere quale eliminare premo il numero della lista corrispondente al giocatore. |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Ambiente** | *Driver* Main *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | 5 |
| **Uscite attese** | Eliminazione di tutti i giocatori. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il menu principale risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Suite** | 7 |
| **Software Testato** | Gioco-partita con l’interazione della banca e con il valore su ogni casella. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Risultato** | Il risultato finale corrisponde alla consegna data. |
| **Autore** | Falleti Davide. |
| **Data** | 20/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare che ogni casella abbia un valore in euro. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Ogni casella ha il suo costo. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare che quando un giocare arriva su una casella che non è ancora stata acquistata, la acquista.  automaticamente. |
| **Ambiente** | *Driver:*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore acquista automaticamente la proprietà. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 3 |
| **Obiettivo** | Verificare che il giocare acquisti la proprietà solo se ha denaro sufficiente. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Messaggio di fondi insufficienti e non avviene l’acquisto della proprietà. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 4 |
| **Obiettivo** | Verificare che quando un giocatore arriva su un terreno o una stazione di proprietà di un altro giocatore, deve pagare a quest'ultimo il 10% del valore della casella. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Pagamento del 10% della casella all’avversario. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 5 |
| **Obiettivo** | Verificare che quando un giocatore arriva su una società di proprietà di un altro giocatore, l'affitto deve essere 4 volte il numero ottenuto con i dadi. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | L’affitto pagato (in €) all’avversario è dato dal valore dei dadi moltiplicato per 4. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Questo avviene se l’avversario possiede una sola società di servizi. |

|  |  |
| --- | --- |
| **Test Case** | 6 |
| **Obiettivo** | Verificare che quando un giocatore arriva su una società di proprietà di un altro giocatore, l'affitto deve essere 10 volte il numero ottenuto con i dadi. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | L’affitto pagato (in €) all’avversario è dato dal valore dei dadi moltiplicato per 10. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Questo avviene se l’avversario possiede entrambe le società di servizi. |

|  |  |
| --- | --- |
| **Test Case** | 7 |
| **Obiettivo** | Verificare che se il giocatore possiede tutti i terreni di un certo colore, l'affitto raddoppia. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | L’affitto pagato all’avversario è doppio. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 8 |
| **Obiettivo** | Verificare il fallimento di un giocatore. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore esce dal gioco. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Questo avviene se un giocatore non possiede un capitale sufficiente per pagare per intero l’affitto dovuto a un altro giocatore. |

| **Test Case** | 9 |
| --- | --- |
| **Obiettivo** | Verificare che se un giocatore si sposta su una casella di un altro giocatore ma non ha i soldi per pagare l'affitto, fallisce dando il suo capitale al giocatore che possiede la proprietà e quest'ultimo riceve la differenza del valore dovuto dalla banca. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | L’avversario riceve i soldi dal giocatore che ha fallito e riceve il resto dei soldi dalla banca. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 10 |
| **Obiettivo** | Verificare che le proprietà del giocare fallito ritornino acquistabili. |
| **Ambiente** | *Driver*  Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Le proprietà tornano acquistabili. |
| **Uscite effettive** | Le stesse. |
| **Risultato** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Suite** | 8 |
| **Software Testato** | Gioco-partita con l’aggiunta delle PROBABILITA’ e degli IMPREVISTI. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Risultato** | Il risultato finale corrisponde alla consegna data. |
| **Autore** | Falleti Davide. |
| **Data** | 27/03/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare che all'inizio di ogni partita ciascuno dei due mazzi (IMPREVISTI e PROBABILITA') venga mescolato casualmente. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Mazzo mescolato casualmente. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare che il giocatore quando arriva su una casella di probabilità o imprevisti, peschi la carta in cima al mazzo. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore pesca la prima carta del mazzo. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 3 |
| **Obiettivo** | Verificare che il giocatore esegua l'operazione scritta sulla carta di probabilità o imprevisti. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore si comporterà in modo diverso in base all’operazione scritta sulla carta. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 4 |
| **Obiettivo** | Verificare che il giocatore depositi la carta in fondo al mazzo. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore mette la carta in fondo al mazzo. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 5 |
| **Obiettivo** | Verificare il fallimento di un giocatore. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore fallisce ed esce dalla partita. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | Un giocatore fallisce quando non possiede un capitale sufficiente da eseguire l'azione scritta sulla carta di probabilità o di imprevisti. |

|  |  |
| --- | --- |
| **Test Case** | 6 |
| **Obiettivo** | Verificare che se un giocatore fallisce, l'eventuale resto che dovrebbe dare all'avversario venga fornito dalla banca. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | L’avversario riceve una certa quantità di denaro dalla banca. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** | L'eventuale resto è dovuto dal fatto che il giocatore non ha soldi per eseguire l'azione scritta sulla carta di probabilità o imprevisti e fallendo cede tutto il suo capitale all'avversario a cui è dovuto. |

|  |  |
| --- | --- |
| **Test Suite** | 9 |
| **Software Testato** | Gioco-partita con l’aggiunta di alcune regole per la gestione della PRIGIONE. |
| **Versione** | 1 |
| **Obiettivo** | Verificare il corretto funzionamento del Menu Principale. |
| **Risultato** | Il risultato finale corrisponde alla consegna data. |
| **Autore** | Cordioli Francesco. |
| **Data** | 03/04/2015 |

|  |  |
| --- | --- |
| **Test Case** | 1 |
| **Obiettivo** | Verificare che il giocatore finisca in prigione quando pesca una carta di PROBABILITA' O IMPREVISITI che gli indica di andare in prigione. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore entra in prigione. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 2 |
| **Obiettivo** | Verificare che il giocatore finisca in prigione quando, tirando i dadi, esce per tre volte dadi doppi. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore finisce in prigione. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 3 |
| **Obiettivo** | Verificare che, al turno successivo, prima di tirare i dadi, il giocatore paghi 50€ alla banca per uscire di prigione. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore esce di prigione e la banca ha 50€ in più. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 4 |
| **Obiettivo** | Verificare che se un giocatore è in prigione e non possiede un capitale sufficiente per uscirne, fallisce. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore fallisce. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 5 |
| **Obiettivo** | Verificare che tutte le proprietà del giocatore fallito ritornino acquistabili. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Le proprietà sono riacquistabili. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 6 |
| **Obiettivo** | Verificare che tutto il capitale del giocatore fallito venga ceduto alla banca. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | La banca ha in più il denaro del giocatore fallito. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |

|  |  |
| --- | --- |
| **Test Case** | 7 |
| **Obiettivo** | Verificare che il giocatore fallito esca dalla partita. |
| **Ambiente** | *Driver:* Terminal\_in\_out  *Oracle:* Main  *Stub:* |
| **Dati in ingresso** | Nessuno. |
| **Uscite attese** | Il giocatore non c’è più nella partita. |
| **Uscite effettive** | Le stesse. |
| **Risultato'** | PASS!  Il programma risponde correttamente ai comandi dell’utente. |
| **Osservazioni** |  |