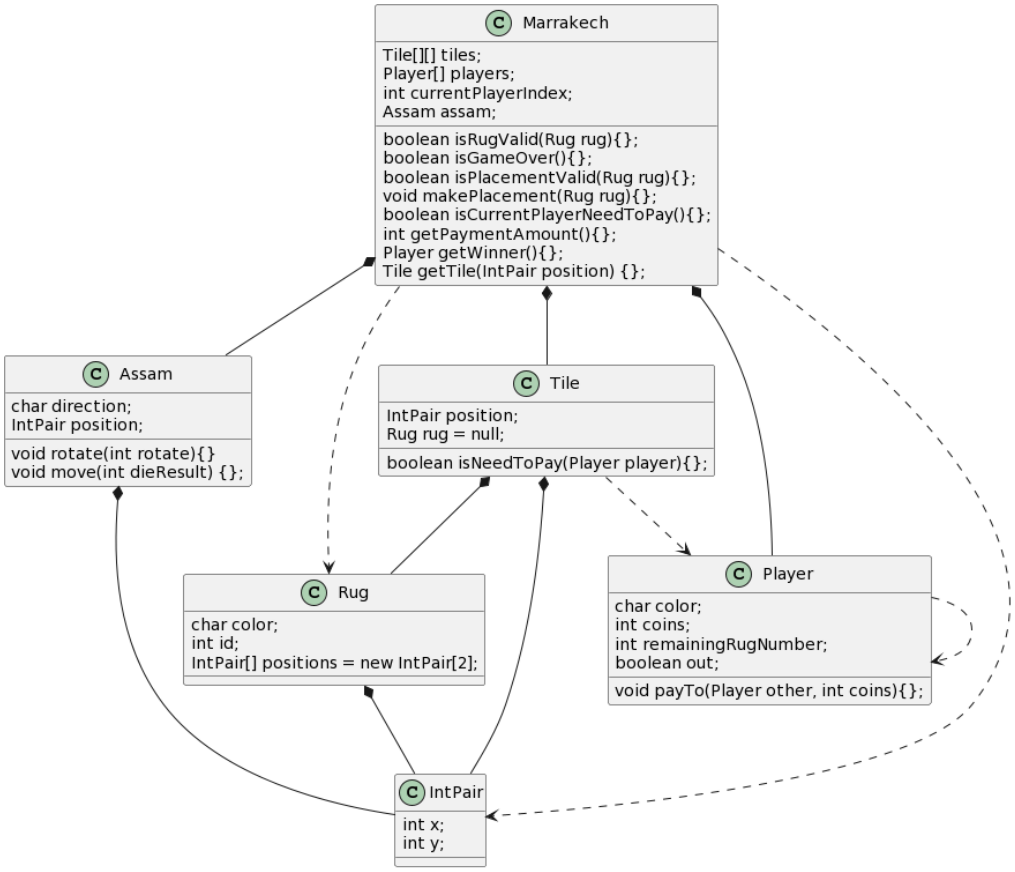
DESIGN SKELETON

PROJECT	COMP6710 ASSIGNMENT2
GROUP	tue10b4
MEMBERS	Xinyang Li, Tashia Tamara, Jiangbei Zhang

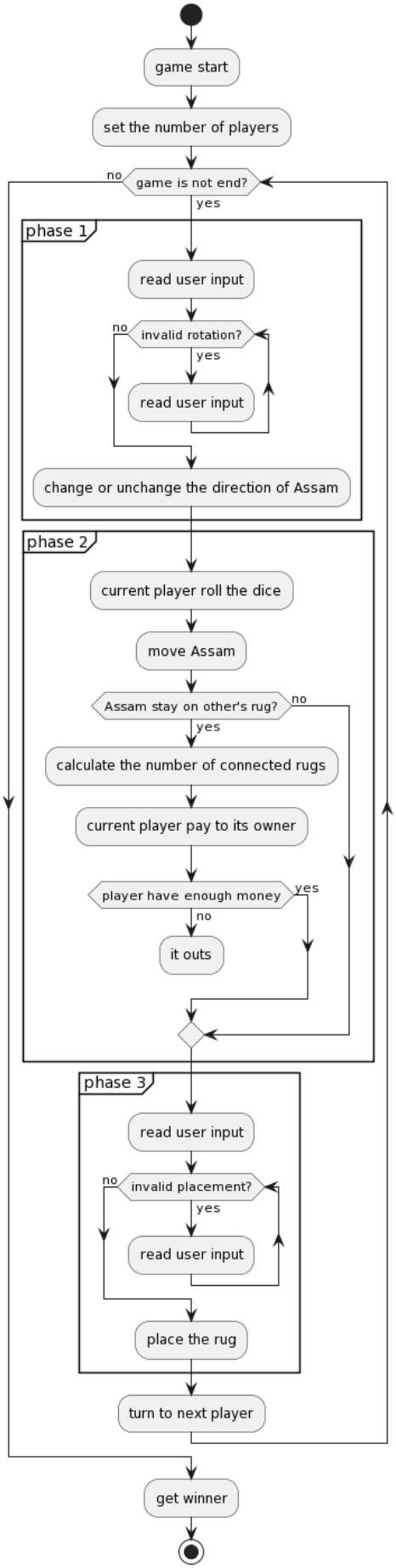
Class Diagram




```
@startuml
class Player {
	char color;
	int coins;
	int remainingRugNumber;
	boolean out;
	void payTo(Player other, int coins){};
}

class IntPair {
	int x;
	int y;
}

class Marrakech {
	Tile[][] tiles;
```

```
        Player[] players;
        int currentPlayerIndex;
        Assam assam;
        boolean isRugValid(Rug rug){};
        boolean isGameOver(){};
        boolean isPlacementValid(Rug rug){};
        void makePlacement(Rug rug){};
        boolean isCurrentPlayerNeedToPay(){};
        int getPaymentAmount(){};
        Player getWinner(){};
        Tile getTile(IntPair position) {};
}

class Assam {
        char direction;
        IntPair position;
        void rotate(int rotate){}
        void move(int dieResult) {};
}

class Rug {
        char color;
        int id;
        IntPair[] positions = new IntPair[2];
}

class Tile {
        IntPair position;
        Rug rug = null;
        boolean isNeedToPay(Player player){};
}

Marrakech *-- Tile
Marrakech *-- Player
Marrakech *-- Assam
Marrakech ..> Rug
Marrakech ..> IntPair
Assam *-- IntPair
Player ..> Player
Tile *-- IntPair
Tile *-- Rug
Tile ..> Player
Rug *-- IntPair
@enduml
```

Game Flow


game start
set the number of players
no
game is not end?
yes
phase 1
read user input
no
invalid rotation?
yes
read user input
change or unchange the direction of Assam
phase 2
current player roll the dice
move Assam
Assam stay on other's rug?
no
yes
calculate the number of connected rugs
current player pay to its owner
player have enough money
yes
no
it outs
phase 3
read user input
no
invalid placement?
yes
read user input
place the rug
turn to next player
get winner

```
@startuml
start

:game start;
:set the number of players;

while (game is not end?) is (yes)
	partition phase 1{
		:read user input;
		while (invalid rotation?) is (yes)
			:read user input;
		endwhile (no)
		:change or unchange the direction of Assam;
	}

	partition phase 2{
		:current player roll the dice;
		:move Assam;
		if (Assam stay on other's rug?) then (yes)
			:calculate the number of connected rugs;
			:current player pay to its owner;
			If(player have enough money) then (yes)
			else (no)
				:it outs;
				kill
			endif
		else (no)
		endif
	}

	partition phase 3{
		:read user input;
		while (invalid placement?) is (yes)
			:read user input;
		endwhile (no)
		:place the rug;
}

:turn to next player;
endwhile (no)

:get winner;

stop
@enduml
```