GAME MANAGER

```
try{
 while(table.getRound()<=table.getmaxRounds())
        {
           synchronized(table)
        {
           while(table.getroundCards().size()!=4)
            table.wait();
        round_Winner = roundWinner();
        round_Winner.setRoundsWon(round_Winner.getroundsWon()+1);
           Thread.sleep(100);
           System.out.println("Player "+ round_Winner.getPlayersName() + " won round "+ table.getRound());
           table.getroundCards().clear();
           table.setRound(table.getRound()+1);
           if(table.getRound()<=table.getmaxRounds())
           {
              System.out.println(" round "+ table.getRound() + " is going to being");
              setPlayersTurn();
              table.setcurrentTurn(1);
          }
           table.notifyAll();
       } // end of synch
        } // end of while
} // end of try
catch(Exception e){
        e.printStackTrace();}

Player winnerOfGame=gameWinner();
System.out.println("Winner of the game: "+winnerOfGame.getPlayersName());
System.out.println(" game is over");
```

PLAYER

```
try
 {
    while(table.getRound()<=table.getmaxRounds())
    {
      synchronized(table)
      {
        while((this.getTurn()!=0) && (this.getTurn()!=table.getcurrentTurn()) && (table.getRound()<=table.getmaxRounds()))
          table.wait();
        if(table.getRound()<=table.getmaxRounds())
        {
          Thread.sleep(100);
          cardToThrow=selectCardToThrow();
          System.out.println("player "+this.getPlayersName()+" has thrown "+cardToThrow.getcategory()+" "+cardToThrow.getnumber());
          table.getroundCards().add(cardToThrow);
          playerCards.remove(cardToThrow);
          table.setcurrentTurn(table.getcurrentTurn()+1);
          table.notifyAll();
        }
      } // end of synch
    } // end of while
} // end of try
catch(Exception e){
    e.printStackTrace();}


    } // end of run method
```