GAME RULES::

1. Checkers is played by two players. Each player begins the game with 12 colored discs. (Typically, one set of pieces is black and the other red.)
2. The board consists of 64 squares, alternating between 32 dark and 32 light squares. It is positioned so that each player has a light square on the right side corner closest to him or her.
3. Each player places his or her pieces on the 12 dark squares closest to him or her.
4. Black moves first. Players then alternate moves.
5. Moves are allowed only on the dark squares, so pieces always move diagonally. Single pieces are always limited to forward moves (toward the opponent).
6. A piece making a non-capturing move (not involving a jump) may move only one square.
7. A piece making a capturing move (a jump) leaps over one of the opponent's pieces, landing in a straight diagonal line on the other side. Only one piece may be captured in a single jump; however, multiple jumps are allowed on a single turn.
8. When a piece is captured, it is removed from the board.
9. If a player is able to make a capture, there is no option -- the jump must be made. If more than one capture is available, the player is free to choose whichever he or she prefers.
10. A player wins the game when the opponent cannot make a move. In most cases, this is because all of the opponent's pieces have been captured, but it could also be because all of his pieces are blocked in.

GAME CODE(TO BE MADE VISIBLE AS PROBLEM STATEMENT)::

```
int main()
{
    int board[8][8],i,j;
    //Set the board
    initialize(board);
    //Start the game
    start(board);
    for(i=0;i<8;i++)
```

```
    {

        for(j=0;j<8;j++)
            cout<<board[i][j]<<" ";
        cout<<"\n";
    }
    return 0;
}

void start(int board[][8])
{
    int draw=0,aux[8][8],row1,col1,row2,col2,end=1;
    int flag=0;
    while(!game_end(board))
    {
        make_copy(board,aux);
        if(flag==0)
        {
            //move_black(board,row1,col1,row2,col2,black);
            if(valid(board,black,row1,col1,row2,col2))
            {
                flag=1;
                if (check_draw(aux,board))
                {
                    draw++;
                    if(draw==10)
                    {
                        cout<<"Draw";
                        end=0;
                        break;
                    }
                }
                else
                {
                    draw=0;
                }
                update(board,row1,col1,row2,col2,black);
            }
            else
            {
                cout<<"Invalid Move";
                end=0;
                break;
            }
        }
        else
        {
            //move_red(board,row1,col1,row2,col2,red);
            if(valid(board,red,row1,col1,row2,col2))
            {
                flag=0;
                if (check_draw(aux,board))
                {
                    draw++;
                    if(draw==10)
                    {
                        cout<<"Draw";
                        end=0;
                        break;
                    }
                }
```

```
                else
                {
                    draw=0;
                }
                update(board,row1,col1,row2,col2,red);
            }
            else
            {
                cout<<"Invalid Move";
                end=0;
                break;
            }
        }
    }
    if(end==1)
    {
        winner(board);
    }
}
```

THE CONTESTANT HAS TO MAKE THE MOVE FUNCTION.

WHETHER THEY WILL BE GIVEN BLACK OR RED WILL BE DECIDED RANDOMLY AT THE TIME OF CONTEST.