Eval Function

```
private static int[] evaluationfunction(int redballs, int blueballs, int depth,boolean turn)
{
   if(redballs == 0 || blueballs == 0 ) {


      if(turn==true)
      {
         int g[]={((2*redballs)+(3*blueballs)),1};
         return g;
      }
      else
      {
         int g[]={-((2*redballs)+(3*blueballs)),1};
         return g;
      }


   }
   if(depth<=0)
   {
       if(turn==true)
      {
         int g[]={((2*redballs)+(3*blueballs)),1};
         return g;
      }
      else
      {
         int g[]={-((2*redballs)+(3*blueballs)),1};
         return g;
      }
   }
      int g[]={0,-1};
      return g;
}
```

This code defines an evaluation function for the game.The function takes in four parameters:

- redballs: the number of red balls left on the board
- blueballs: the number of blue balls left on the board
- depth: the current depth in the game tree (how many moves have been made so far)
- Turn: true if it is a computer. False if it is a human

The function returns an array of two integers:

- The first integer represents the score of present state of the game
- The second integer is a flag indicating whether the game is over (1 if the game is over, -1 if the game is not over).

The minimizingFunction and maximizingFunction methods invoke the eval function to assess the game's current state.

The function first checks if either the red balls or the blue balls are empty. If this is the case, the function returns a score based on the remaining balls (2 points for each red ball and 3 points for each blue ball) and sets the game over flag to 1.

If the game is not over, the function checks if the maximum search depth has been reached. If it has, the function returns the same score as if the game were over, and sets the game over flag to 1.

If neither of these conditions apply, the function returns a score of 0 and sets the game over flag to -1 (indicating the game is not over yet). This means that the function is not making any evaluation at this point and will rely on other parts of the algorithm to explore further moves.