# Code Review

This is the third entry to my portfolio and I have been tasked this week to look into reviewing code. This involves looking at code and finding faults, whether they are general principle violations, best practice rule violations, or if the code has any coding smells.

The tasks I must complete this week are as follows:

- Choose the code review challenge which best demonstrates my skills
- Copy the code into my portfolio
- Provide descriptive commentary that identifies the problems
- Show my improved version of the code in a new code block
- Explain why my solution is a good a good one

I will make note that I have recently started coding in C# and that my previous academic experience taught me how to do things which I have now discovered aren't correct. This is important due to multiple reasons, the first is the way that code is typed in C# is different to what I am used to in other languages. I am, of course, working on this, however with the time sensitive window for work hand in's, I may lag behind for a few weeks. The second reason is that I learned in class that what I was previously taught was wrong, for example, with comments, In my previous academic years, I was taught to leave constant detailed comments explaining every part of my code. I discovered now that this isn't a professional way to do this, and it's better to make my code descriptive and only include comments if necessary.

These reasons have lead to blunders on even the most simple of coding reviews, as I may have thought comments were missing, or that the code was correct when it may be correct. However, I have made my best attempt at the tasks required.

## The problematic code

```
1 - public string GetPlayerStatus(Player player)
2 - {
3 -     static int name = player.name;
4 -
5 -     if (player.IsOnline) {
6 -         if (player.CurrentGame != null) {
7 -             return "Player is currently in a game";
8 -         } else {    // Player is waiting
9 -             if (player.PendingInvitations.Count > 0) {
10-                 return "Player has pending invitations";
11-           } else {
12-                 return "Player is online";
13-            }
14-        }
15-    } else {
16-        return "Player is offline";
17-    }
18-}
```

The code above is created simply to show the status of a player, most likely in a video game. I have annotated it with line numbers to help with clarity.

After a brief glance I did find some obvious issues. As can be seen from line 3, the player name is created however it is never used. This is in breach of the YAGNI principle, which states that code should be added when necessary and not pre-emptively. The second issue I saw with this line is that it is a static variable in a non static function. This cannot exist in C# and would cause a compile error, so this is a breach of C# coding conventions. The third issue I considered was that the variable is called "name" but expects an "int", I considered that this would violate best practice rule which declares "Use appropriate names for variables, functions, classes, etc". However, I would note that it wasn't considered as this in the marking scheme.

Continuing on, from line 5 to line 17 is an if statement with nested if statements. Although the code works it could have been done simpler, thus this breaches the KISS princples.

Something that I missed was the comment on line 8 being in breach of "Use recognised coding conventions". The reason for this is that the comment is inline, and it should be above the code that it is describing as opposed to beside the else statement. It also doesn't provide much information, considering there are numerous nested if statements.

## The improved code

```
public string GetPlayerStatus(Player player)
{

    if (!player.IsOnline)
    {
        return "Player is offline";
    }

    if (player.CurrentGame != null)
    {
        return "Player is currently in a game";
    }

    if (player.PendingInvitations.Count > 0)
    {
        return "Player has pending invitations";
    }

    return "Player is online";

}
```

The code provided contains none of the issues of the first code block. The unused, static variable has been removed entirely as it contributed nothing to the code, this eliminates the YAGNI breach. The nested if statements have been separated into a much clearer format which is easier to read and understand, which elimites the KISS breach. Finally, the comment was removed entirely, as it provided nothing of value and the code is now easily readable, thus not requiring any comments.