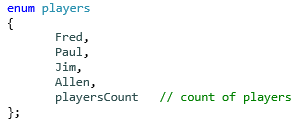
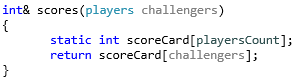
The code starts off creating an enum (Enumeration) called players. An enum allows the code to create its own data type within the application and unlike predefined data types, you can determine the scope and significance of the enum type. It acts like a custom type.

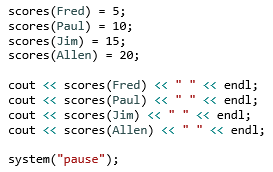
The enum is of 4 players with a count at the bottom.



The code then makes a **scores** function which uses a reference in its return type (int&). This function also takes in the player as an argument which tells me that the enum players is used as an array of scores also. This function is an LValue as when values are passed in and scores are given, it must be on the Left-hand side as seen further below.



When the code is ran we can see that this **scores** function returns the score of the player that is passed in as an argument. So when Fred is given a score of 5 the function simply returns 5. Also, when Fred is given the score of 5 the 5 is on the Right-hand side of the equals making this the RValue and the scores(Fred) is on the Left-hand side of the equals making this an LValue. The same goes for COUT being on the Left-hand side making it an LValue.





In this code, the playersCount isn’t displayed to the user but all you need to do is put the following line of code into the main and it returns the 4 players that were put into the enum at the start:

cout << playersCount << " " << endl;