Documentation

Project: Cricket Scoreboard

Date: 13-Sep-25

Author: Abhishek Garg

1. Project Overview

This project is a console-based cricket match management system that displays the output on the scoreboard based on input from the user.

It provides functionality for:

* Number of teams to be played
* Number of matches that has to be played
* Player creation for both team in a match
* Live scoreboard display
* Match result declaration

The project is modularized into separate header (.h) and src (.cpp) files for maintainability.

1. Assumptions

Certain assumptions were made during development:

* As per cricket standard a match is played between two teams.
* Every player comes to bat in the order they are added to the list.
* As per cricket standard there can be a case where every player has to bat, so player class has member variable runs.
* Only one player is batting at time.

1. Build & Run Instruction

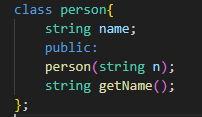




1. Core Classes & Responsibilities

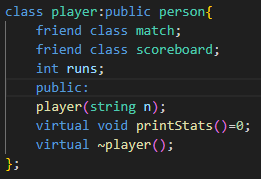
**Person Class**

Purpose: It contains member variable of type string that stores **name**. And a public function that displays name.



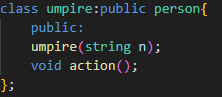
**Player Class**

Purpose: It is an abstract class with member variable as **runs**. It is inherited from Person class. It is friend of class match and scoreboard, which will be explained in next segments.



**Umpire Class**

Purpose: This class is inherited from person class. And displays the action of umpiring.



**Specializations**

There are 4 specialization classes Batsman, Bowler, AllRounder, Wbat. All are inherited from player class.

Batsman: If a player can bat and field.

Bowler: If a player is bowl and field.

AllRounder: If a player is bat and bowl.

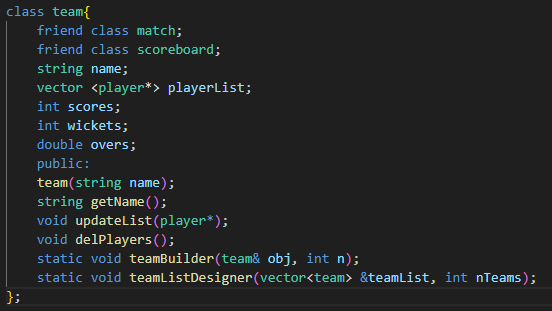
Wbat: If a player can bat and can do wicketkeeping.

**Team Class**

This class contains member variables scores, wickets, overs, name of the team as member variables and a vector for list of players that stores pointer of type player. All this things are useful in displaying in scoreboard.

It has two static functions, first is teamListDesigner that is used to take input of team name and assign them to the list. And second teamBuilder which takes input from the user about the player of the team and create them dynamically and assign the specialization based on the input from the user.

This class is friend of match class and scoreboard class



**Scoreboard class**

This class is used to display the output,

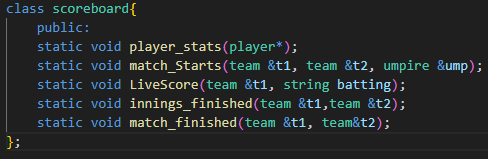
Player\_stats displays the player specializations

Match\_starts displays the initial details of the match, like umpire name, team names, etc.

LiveScore displays the score after every ball, it shows team name, score/wickets, overs and name of player whose batting

Innings\_finished displays the score of the first team after the innings is finished along with runs scored by each batsman.

Match\_finished displays the outcome and run scored by batsman of each teams.



**Match Class**

It contains data member overs which stored the number of over for an match has to be played. Pointers for team 1 and team 2 and a pointer for umpire.

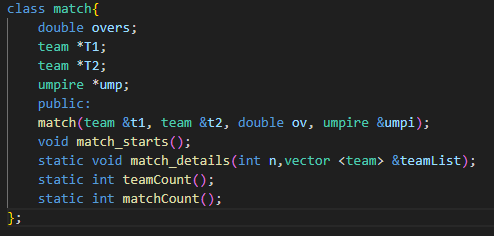
It has three static functions:

teamCount: It is used to take input of total playing teams.

matchCount: It is used to take input of total matches that has to be played.

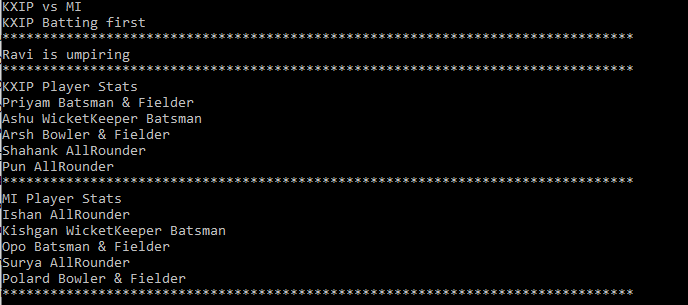
Match\_details: It is used to choose the from the list of teams and then number of players playing and then call teambuilder function of team class.

After that it creates an object of match type with all the details and call match start.



1. Code Flow

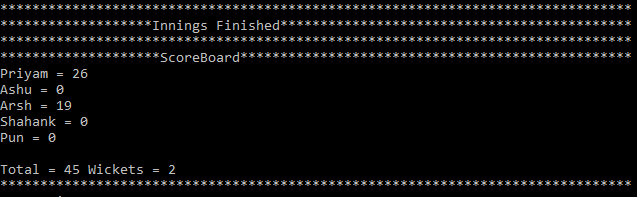
* A list of total teams is present in main. Then api match::teamCount() is used to take input of total number of teams, based on this result team::teamListDesigner is called to add the input to the team list.
* After that match::matchCount() is called to take input of total matches that has to be played. With this result match::match\_details is called.
* Match details take input of total players playing among each, and then ask to choose a team based on team::teamBuilder.
* team::teamBuilder takes input for each player and ask for specialization and based on this it dynamically creates the player.
* After that umpire is created.
* After that match\_starts is called.
* Then scoreboard:match\_Starts is called which displays the name of team playing, specialization of each player, which team is batting first.



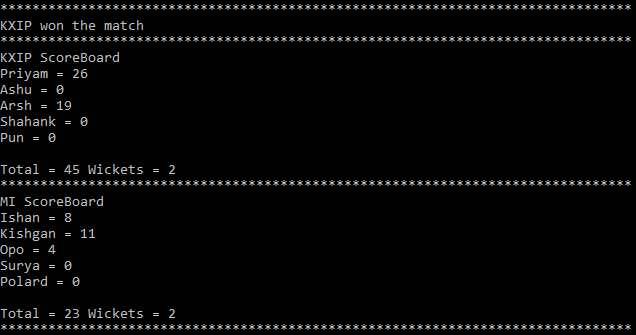
* Then team play bats and with every ball scoreboard::LiveScore is called which displays the live scoreboard, playing team name || score/wicket || overs || Whose Batting



* After first inning is completed, it calls scoreboard::innings\_finished that dispels score of first team and runs scored by each player of the team.



* Now second teams bats and live score is also shown for every ball.
* After this scoreboard::match\_finished is called which shows the outcome and scores of each team and scores of each players.



* After that players are deleted.

1. Future Enhancements

* Scoreboard can show who is bowling
* Bowling stats can also be visible
* Two batters can be on field at a time.