**Project Main Report**

(Inheritance & Polymorphism)

**Problem statement:**

Implement the following class hierarchy in C++. Write a driver to test the hierarchy displaying data of your favorite players.

**Requirements:**

* Program should be well designed using OOP concepts learned so far.
* Computeave() is used to calculate average goals/score of players
* Data should be displayed by overloading stream insertion operators
* You are allowed to add appropriate data members and functions as required
* Use text file to read data of your favorite 10-15 players.

**Objectives:**

Main objectives of this project are undermentioned:

1. To Develop a program that can perform above mentioned features efficiently
2. To develop an efficient management system for sports board
3. To deeply understand clear concepts of Inheritance, Multiple Inheritance, Multi-level Inheritance
4. To understand the concept of Polymorphism
5. To review overloading and filing concepts .
6. To understand the practical real life relation of user’s demand for a simple software like this.

**Conclusion:**

After completing this project, we are practically aware of the concepts of Inheritance, polymorphism and operator overloading. Practical programming approach is a bit different than the educational one. We were successful to develop a program through the concept of file handling that can act as a text editor for different database systems. In my case I have developed it for a Sports Board management system.

**Brief Tour De Program**

Lets take a brief of tour of this program…Ready..Start..GO!!!!

* It is a program that will act as a record management system for Pakistan Sports Board
* This Board has two different divisions for Football and Cricket.
* System is able to store new data for players also it can display all the previous record of registered players with Pakistan sports board.
* At the start program will also provide instructions on how to use this program
* This program is developed using concepts Inheritance, Multi-level Inheritance, Multiple Inheritance, Polymorphism, operator overloading and file handling
* This program is compiled through separate compilation as we have 6 header file, 1 implementation file and 1 Main file.

*Thank You!*

**Main.cpp**

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

#include"Sportsman.h"

#include"Footballer.h"

#include"Cricketer.h"

#include"Bowler.h"

#include"Batsman.h"

#include"Allrounder.h"

using namespace std;

int main()

{

system("color 0A");

int choice;

int n;

f: cout << " \*\*\*Welcome to Pakistan Sports Board\*\*\*\n";

cout << "Press 1 to display previous record of all Football Players: \n";

cout << "Press 2 to display previous record of all cricketers: \n";

cout << "Press 3 to enter Record of Football players\n";

cout << "Press 4 to enter record of Cricketers which includes Bowler,Batsman and Allrounders\n";

cin >> choice;

switch (choice)

{

case 1:

{

cout << " \*\* Welcome To Football Division of PSB\*\*\*\n";

cout << "Following players are regidtered in PSB at this time:\n";

Footballer F1;

F1.show();

cout << endl;

goto f;

}

case 2:

{

cout << " \*\*Welcome to Cricket Division of PSB\*\*\n";

cout << "- Following players are registered with PSB at this time:\n";

Cricketer CC;

CC.show();

cout << endl;

goto f;

}

case 3:

{

cout << " \*\*Footallers Record System\*\*\n";

cout << "kindly enter the requested Data for Football players:\n";

Footballer F;

F.getdata();

cout << "\n \*Displaying Footballer's Data: \n";

cout << F;

cout << "Goals average of this player: " << F.computeave();

cout << endl;

goto f;

}

case 4:

{

cout << " \*\*Cricketers Record System\*\*\n";

int wish;

cout << "Press 1 for Bowler - 2 for Batsman - 3 for Allrounder\n";

cin >> wish;

if (wish == 1)

{

cout << "- Enter requested data for Bolwer:\n";

Bowler B;

B.getdata();

cout << "-Displaying Player's Record\n";

cout << B;

cout << "Wickets average of this player: " << B.computeave();

cout << endl;

goto f;

}

if (wish == 2)

{

cout << "- Enter requested data for Batsman:\n";

Batsman BT;

BT.getdata();

cout << "-Displaying Player's Record\n";

cout << BT;

cout << "Score/Runs average of this player: " << BT.computeave() << endl;

goto f;

}

if (wish == 3)

{

cout << "- Enter requested data for Allrounder:\n";

Allrounder AL;

cout << "Enter Data for Allrounder: " << endl;

AL.getdata();

cout << "-Displaying Player's Record\n";

cout << AL;

cout << "Allrounder's batting average: " << AL.computeave() << endl;

cout << "Allrounder's bowling average: " << AL.Bowler::computeave() << endl;

goto f;

}

}

default:

cout << "You have entered invalid choice! Try again:\n";

goto f;

}

system("pause");

return 0;

}

**implementation.cpp**

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

#include"Sportsman.h"

#include"Footballer.h"

#include"Cricketer.h"

#include"Bowler.h"

#include"Batsman.h"

#include"Allrounder.h"

using namespace std;

Sportsman::Sportsman()

{

firstname = " ";

lastname = " ";

matchesplayed = 0;

}

Sportsman::Sportsman(string FN, string LN, int MP)

{

firstname = FN;

lastname = LN;

matchesplayed = MP;

}

void Sportsman::getdata()

{

cout << "Enter Firstname of player: \n";

cin >> firstname;

cout << "Enter Lastname of player: \n";

cin >> lastname;

cout << "Enter Total number of matches played: \n";

cin >> matchesplayed;

}

void Sportsman::display()

{

cout << "First name: " << firstname << endl;

cout << "Last name: " << lastname << endl;

cout << "Matches played: " << matchesplayed << endl;

}

Footballer::Footballer()

{

team = " ";

goals = 0;

position = 0;

}

Footballer::Footballer(string FN, string LN, int MP, string TM, int G, int P) :Sportsman(FN, LN, MP)

{

team = TM;

goals = G;

position = P;

}

void Footballer::getdata()

{

Sportsman::getdata();

cout << "Enter Footballer's Team: \n";

cin >> team;

cout << "Enter Total Goals of player: \n";

cin >> goals;

cout << "Enter position of player in Team: \n";

cin >> position;

}

void Footballer::show()

{

fstream file;

file.open("Football.txt", ios::in);

if (file)

{

string a;

while (!file.eof())

{

string s;

getline(file, s);

cout << s << endl;

}

}

file.close();

}

int Footballer::computeave()

{

int ave;

ave = goals / matchesplayed;

return ave;

}

Cricketer::Cricketer()

{

cteam = " ";

runs = 0;

}

Cricketer::Cricketer(string FN, string LN, int MP, string CTM, int RNS) :Sportsman(FN, LN, MP)

{

cteam = CTM;

runs = RNS;

}

void Cricketer::getdata()

{

Sportsman::getdata();

cout << "Enter player's team: " << endl;

cin >> cteam;

cout << "Enter player's Total runs: " << endl;

cin >> runs;

}

void Cricketer::show()

{

fstream file;

file.open("Cricket.txt", ios::in);

if (file)

{

string a;

while (!file.eof())

{

string s;

getline(file, s);

cout << s << endl;

}

}

file.close();

}

int Cricketer::computeave()

{

int ave;

ave = runs / matchesplayed;

return ave;

}

Bowler::Bowler()

{

type = " ";

wickets = 0;

}

Bowler::Bowler(string FN, string LN, int MP, string CTM, int RNS, string TYP, int WI) :Cricketer(FN, LN, MP, CTM, RNS)

{

type = TYP;

wickets = WI;

}

void Bowler::getdata()

{

Cricketer::getdata();

cout << "Enter Bowlers's type: " << endl;

cin >> type;

cout << "Enter Total wickets taken: " << endl;

cin >> wickets;

}

int Bowler::computeave()

{

int ave;

ave = wickets / matchesplayed;

return ave;

}

Batsman::Batsman()

{

centuries = 0;

}

Batsman::Batsman(string FN, string LN, int MP, string CTM, int RNS, int CE) :Cricketer(FN, LN, MP, CTM, RNS)

{

centuries = CE;

}

void Batsman::getdata()

{

Cricketer::getdata();

cout << "Enter Total centuries by player: " << endl;

cin >> centuries;

}

void Batsman::display()

{

Cricketer::display();

cout << "No of Centuries: " << centuries << endl;

}

int Batsman::computeave()

{

int avee;

avee = runs / matchesplayed;

return avee;

}

Allrounder::Allrounder()

{

Bowler::Bowler();

Batsman::Batsman();

}

void Allrounder::getdata()

{

Bowler::getdata();

cout << "Enter Total centuries by player: " << endl;

cin >> centuries;

}

int Allrounder::computeave()

{

int avee;

avee = Bowler::runs / Bowler::matchesplayed;

return avee;

}

Sportsman.h

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

using namespace std;

class Sportsman {

protected:

string firstname;

string lastname;

int matchesplayed;

public:

Sportsman();

void getdata();

Sportsman(string FN, string LN, int MP);

void display();

virtual int computeave() = 0;

};

**Footballer.h**

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

#include"Sportsman.h"

using namespace std;

class Footballer :public Sportsman {

protected:

string team;

int goals;

int position;

public:

Footballer();

void getdata();

Footballer(string FN, string LN, int MP, string TM, int G, int P);

friend ostream& operator<< (ostream& output, const Footballer& F)

{

output << "Full name of player: " << F.firstname << " " << F.lastname << endl;

output << "Total matches played: " << F.matchesplayed << endl;

output << "Player's Team: " << F.team << "\nTotal Goals by player: " << F.goals << endl;

output << "Player's position in Team: " << F.position << endl;

return output;

}

void show();

int computeave();

};

**Cricketer.h**

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

#include"Sportsman.h"

using namespace std;

class Cricketer :public Sportsman {

protected:

string cteam;

int runs;

public:

Cricketer();

void getdata();

Cricketer(string FN, string LN, int MP, string CTM, int RNS);

friend ostream& operator<< (ostream& output, const Cricketer& C)

{

output << "Full name of player: " << C.firstname << " " << C.lastname << endl;

output << "Total matches played: " << C.matchesplayed << endl;

output << "Player's team: " << C.cteam << endl;

output << "Total runs by the player: " << C.runs << endl;

return output;

}

void show();

int computeave();

};

**Bowler.h**

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

#include"Sportsman.h"

#include"Cricketer.h"

using namespace std;

class Bowler :public Cricketer {

protected:

string type;

int wickets;

public:

Bowler();

void getdata();

Bowler(string FN, string LN, int MP, string CTM, int RNS, string TYP, int WI);

friend ostream& operator<< (ostream& output, const Bowler& B)

{

output << "Full name of player: " << B.firstname << " " << B.lastname << endl;

output << "Total matches played: " << B.matchesplayed << endl;

output << "Player's Team: " << B.cteam << "\nTotal runs by player: " << B.runs << endl;

output << "Bowler's Type: " << B.type << endl;

output << "Wickets taken in career: " << B.wickets << endl;

return output;

}

int computeave();

};

**Batsman.h**

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

#include"Sportsman.h"

#include"Cricketer.h"

using namespace std;

class Batsman :public Cricketer {

protected:

int centuries;

public:

Batsman();

void getdata();

friend ostream& operator<< (ostream& output, const Batsman& BT)

{

output << "Full name of player: " << BT.firstname << " " << BT.lastname << endl;

output << "Total matches played: " << BT.matchesplayed << endl;

output << "Player's Team: " << BT.cteam << "\nTotal runs by player: " << BT.runs << endl;

output << "Total centuries by player: " << BT.centuries << endl;

return output;

}

Batsman(string FN, string LN, int MP, string CTM, int RNS, int CE);

void display();

int computeave();

};

**Allrounder.h**

#pragma once

#include<iostream>

#include<conio.h>

#include<string>

#include<fstream>

#include"Sportsman.h"

#include"Cricketer.h"

#include"Bowler.h"

#include"Batsman.h"

using namespace std;

class Allrounder :public Bowler, public Batsman {

public:

Allrounder();

void getdata();

friend ostream& operator<< (ostream& output, const Allrounder& AL)

{

output << "Full name of player: " << AL.Bowler::firstname << " " << AL.Bowler::lastname << endl;

output << "Total matches played: " << AL.Bowler::matchesplayed << endl;

output << "Bowler Type: " << AL.type << "\nTotal wickets taken in career: " << AL.wickets << endl;

output << "Player's Team: " << AL.Bowler::cteam << "\nTotal runs by player: " << AL.Bowler::runs << endl;

output << "Total centuries by player: " << AL.centuries << endl;

return output;

}

Allrounder(string FN, string LN, int MP, string CTM, int RNS, int CE, string TYP, int WI)

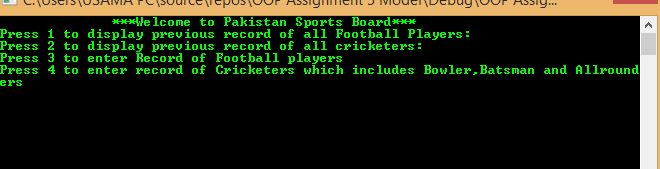
{}

int computeave();

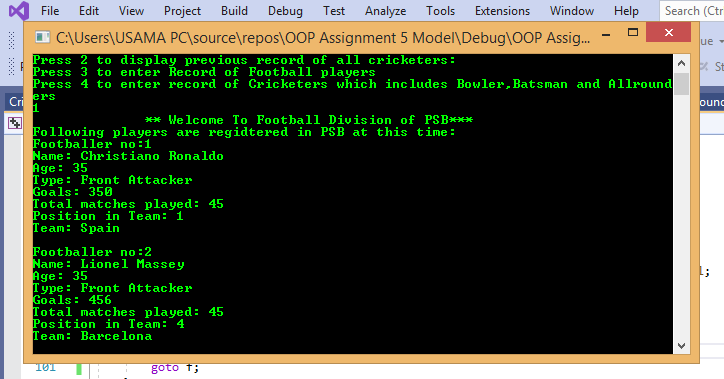
};

**Sample Outputs**

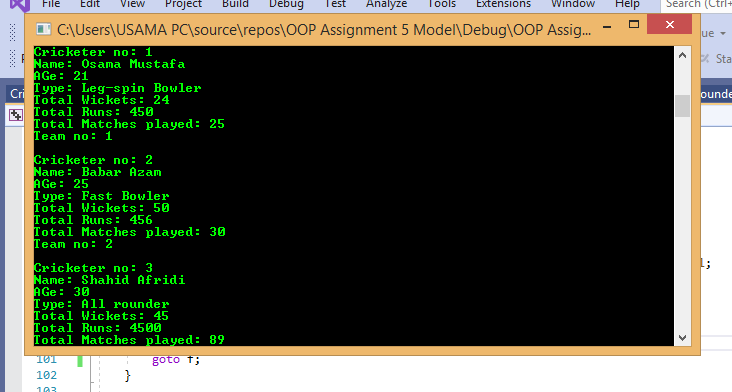
Front Interface – Menu



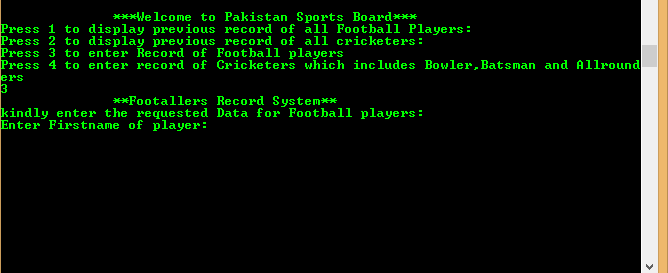
By pressing 1 – Displays all record of Football players from .txt file



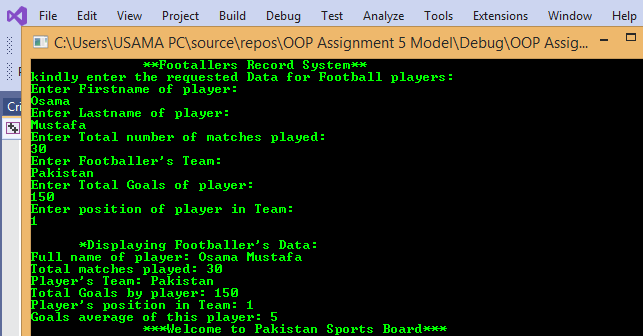
By pressing 2 – Displays all record of Cricketers from .txt file



By pressing 3 – Enters football div – Asks you to enter player’s Data

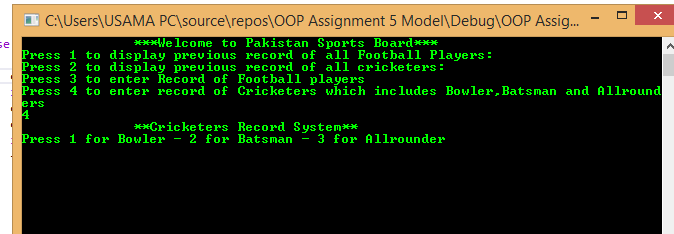


After you enter data it displays the Footballer’s data

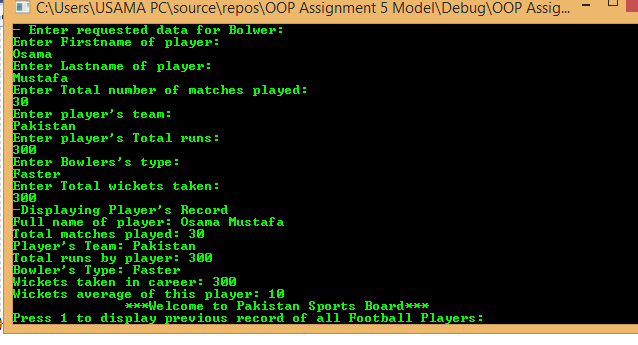


By pressing 4 – You enter Cricket Division

It asks you to select 1 for bowler – 2 for batsman – 3 for Allrounder



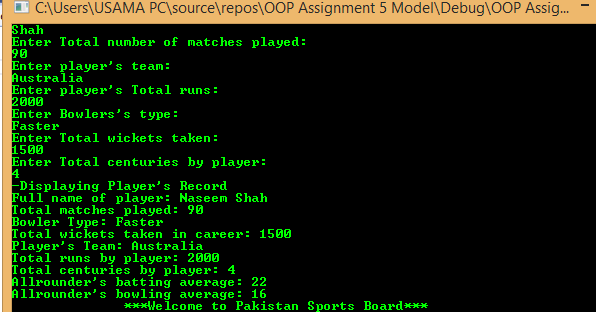
After you Press 1 – You enter Bowler’s sub-division



After pressing 2 – You enter Batsman sub-division



After pressing 3 – You enter Allrounder’s sub-division



*Thank You!*

