National University of Computer and Emerging Sciences, Lahore Campus

Course:	OOP Lab	Course Code:	CL 1004
Program:	BS	Semester:	Fall 2021
Duration:	2 Hours	Total Marks:	100
Paper Date:	17-Jan-2022	Weight	40%
Section:	BCS 3G1, 3G2, 3H1, 3H2, 3J1, 3J2	Page(s):	
	BSE 3A1. 3A2		
Exam:	Lab Final Term	Reg. No.	

The word impossible itself says
I'm Possible.

Read below Instructions Carefully:

- Understanding the question statement is also part of the exam, so do not ask for any clarification. In case of any ambiguity, make suitable assumptions.
- You have to complete exam in 2 hrs. Remaining time will be used for submission.
- For Q1, submit a single file (containing classes definitions and main) named as 21L-1122-Q1.cpp.
- For Q2, submit a single file (containing classes definitions and main) named as 21L-1122-Q2.cpp.
- Submission path: \\Cactus\Xeon\Fall2021\OOPLabFinal
- Submit both questions .cpp file on Google Classroom under assignment titled as **OOP- Lab Final Exam Submission. (Don't create zip file)**
- Your code should be intended and commented properly. Use meaningful variablenames.
- It is your responsibility to save your code from being copied. All matching codes will be considered cheating cases. PLAGIARISM will result in forwarding of case to Disciplinary Committee and negative marks in Midterm.

Question 1:

Implement the following classes, provide the classes definition and implementation code.

A class `Player`. A class `Batsman`. A class `Bowler`.

Create class **Player** having data members

```
class Player {
        char* playername;
        int score;
        int ballsPlayed;
```

CS CamScanner

```
        int matchesPlayed;
        //Your class player should have all necessary functions required to complete the below task
};
```

Each player, either he is bowler or batsman, has a name, score, ballsplayed, and matchesPlayed so these private instance variables appear in which particular class ?

Create class **Batsman**(Batsman is a player) having additional data members,

```
class Batsman {
        char* battingStyle;
        int totalScore;
        int noOfTimesBowled;
        //Your class should have all necessary functions required to complete the below task
};
```

Create class **Bowler**(Bowler is a player) having additional data members

```
class Bowler {
        char* bowlingStyle;
        int totalWickets;
        int totalRunsConceded;
        //Your class should have all necessary functions required to complete the below task
};
```

calCulateAverage() method certainly applies generically to all players. But each average calculation depends on the players's type. Each derived class overrides calCulateAverage() with an appropriate implementation. We cannot calculate the average for a *general* Player.We first must know the *specific* type of Player to determine the appropriate earnings calculation.

Note: Batting average is calculate using totalScore / noOfTimesBowled and Bowling average is calculated using TotalRunsConceded / totalWickets.

display() method in class Player displays the name, score, ballsPlayed, etc. Each derived class of Player overrides function display() to output the player's type (e.g., "Batsman / Bowler:") followed by the rest of the player's information.

```
int main()
{
        Batsman b1{ "Ali ", 23, 160, 5,  "Right - hand", 180, 2 };
        b1.display();
        cout << b1.calCulateAverage();
        Bowler b2{ "Sam ",10, 170, 7,  "Fast - bolwer", 10, 210 };
        b2.display();
        cout << b2.calCulateAverage();
        Player **player_arr = new* Player[2];
        player_arr[0] =// pointer to Batsman object
        player_arr[1] =//pointer to Bolwer object
        for (int i = 0; i < 2; i++)
        {
                player_arr[i]->display();
                player_arr[i]->calCulateAverage();
        }
}
```

Create class **CTeam** having following data members Add:

```
class CTeam {

        char* teamName;

        Player**players;

        //Your class should have all necessary functions required to complete the below task

};

int main() // part of previous main function

{

        CTeam team{" Lahore Qalandars", { "Sam","ALI","David","Hafeez","Wahab" }};// taking team
        name and array of names (names of players)

        team.setTeam();//Run a loop in setTeam and declare players by calling constructor of class
        Player

        cout<< team.getHighest();// returns index of player having highest score

        cout<< team.getHighestBatAvg(); // calculate and return index of player having highest bating
        average rate

        cout<< team.getName(3);//  returns name of 3rd player
        cout<< team.getScore(4);// returns score of 4th player

        cout<< team.getBalls(1);// returns balls of 1st player

        return 0;

}
```

Question 2:

Implement a class **MyDirectory**

At the beginning, declare the following const types:

const int NAMELEN = 10

const int NUMBERLEN = 8

const int ADDRESSLEN = 30

Class named **MyDirectory** having following data members

```
Class MyDirectory {

        Id                      //an integer value between 1-1000

        Name                    //Length of name is 10; char Name[NAMELEN])

        TelephoneNumber         // This is also a char array of size NUMBERLEN but not NULL terminated.

        Address               //Length of Address array is ADDRESSLEN; char Address[ADDRESSLEN]

        //Your class should have all necessary functions required to complete the below task

};
```

Add the constructor and copy constructor to the **MyDirectory** class. Make sure the destination object is a deep copy of the source object. In the constructor without parameters initialize everything to zero/NULL. Also, make a constructor that takes all the above 4 data members as parameter.

```
int main()

{

        MyDirectory directory(20,"FAST NU", "11112812","Block B, Faisal Town");

        cout << directory;

        //The cout statement should output

        //20
        //FAST NU
        //11112812
        //Block B, Faisal Town
        directory[3] = 9 //should make the telephone number 11192812
        int x = directory[5]//  should store 8 in x.
        cout<<x; // should output 8 on screen.
        MyDirectory directory1;
        directory1= directory; //make sure = operator correctly implemented so that no error is generated
        on this statement.
        cout << directory1;
        //The cout statement should also output
        //20
        //FAST NU
        //11192812
        //Block B, Faisal Town
        MyDirectory directory2;
        cin >> directory2;
        cout<< directory2;
        return 0;
}
```