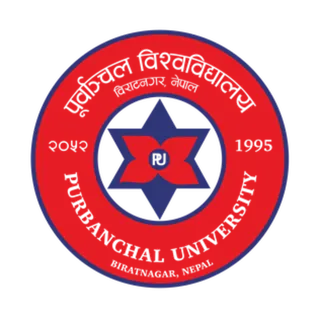
**GOMENDRA MULTIPLE COLLEGE**

**Birtamod-4, Jhapa**

**Affiliated to Purbanchal University**



**Program: BCA-IT**

**Semester: First**

**Faculty: Science and Technology**

Project report on

**CRICKET SCORER APP**

**Submitted By:**

* **Surya Bahadur Basnet**
* **Bikul Khadka**
* **Aakriti Niroula**

**Under the supervision of**

**Assistant lecturer:**

**Mr. Nabin Prasain**

**Acknowledgment**

Developing a project is a challenging endeavor that relies on the cooperation and assistance of various individuals. Often, words fall short when expressing sincere gratitude to those who have contributed to the project's successful completion.

First and foremost, I would like to extend my heartfelt thanks to Mr. Rupak Khanal, the college chief of Gomendra Multiple College, for granting me this invaluable opportunity. Today, as I conclude my project work, my satisfaction knows no bounds. While we encountered numerous obstacles throughout the project's creation, our team leader, Mr. Bharat Oli, provided invaluable support during these challenging times. I must also express my gratitude to my fellow team members for their invaluable assistance and guidance.

I would also like to convey my special appreciation to Mr. Nabin Parsai for his unwavering guidance and invaluable advice throughout the project's duration. He not only supervised our practical work but also imparted essential theoretical knowledge about computerized transportation management systems and their significance in the real practical world.

Table of Contents

|  |  |  |
| --- | --- | --- |
| S.N. | Title | Page No |
| 1. | Introduction to the Project | 1 |
| 2. | Objective of the Project | 2 |
| 3. | Importance of Cricket Scorer App | 3 |
| 4. | Algorithm of Cricket Scorer App | 4 |
| 5. | System Flowchart | 6 |
| 6. | Source Code | 7 |
| 7. | Output of the Project ( Demo ) | 15 |
| 8. | Abstract | 18 |
| 9. | Future Use and Implementation | 19 |
| 10. | Conclusion | 20 |
| 11. | Bibliography | 21 |

**Introduction to the project**

Scores and wickets are extremely important to manage and keep track of in the world of sports, especially in cricket. As a crucial tool for this, the Cricket Scorer App rises up to the plate. For efficient match management in a dynamic and ever-changing sporting environment, accuracy and real-time access to cricket data are crucial.

In order to address these issues, the Cricket Scorer App offers a complete and user-friendly framework for storing, organizing, and retrieving cricket match data. For fans and enthusiasts, this app not only centralizes cricket match results but also provides real-time information.

The Cricket Scorer App's key features include the capacity to log each ball bowled, keep tabs on runs scored and wickets taken, maintain a full scoreboard, and display the team of the batsman and bowler. In conclusion, cricket fans, coaches, and players all rely on the Cricket Scorer App as a crucial tool. It makes sure that the core of the game is precisely and effectively retained, making it an important tool for managing and enjoying cricket.

**Objectives**

The objective of Cricket Scorer App is to streamline the process of scoring cricket matches and provide valuable information to players, coaches, and fans. Here are the key objectives of such an app:

* **Efficient Scoring**: The primary objective is to make scoring cricket matches more efficient. It should allow scorers to easily record every ball bowled, runs scored, and wickets taken with minimal effort.
* **Accuracy:** Ensure the accuracy of the recorded data. Eliminate errors that can occur when scoring manually with pen and paper.
* **Real-time Updates:** Provide real-time updates of the match, including live scores and statistics. This keeps players, coaches, and fans informed about the progress of the game.
* **Historical Data:** Create a repository of historical match data. This allows teams to track their performance over multiple seasons and identify trends.
* **User-Friendly Interface:** Design an intuitive and user-friendly interface so that scorers can quickly adapt to the app. This reduces the learning curve and ensures efficient scoring.
* **Data Accessibility:** Make match data easily accessible to authorized individuals, including team captains, coaches, and league administrators.
* **Customization:** Allow users to customize the app to suit various formats of the game, such as T20, One Day, or Test matches.
* **Fan Engagement**: Engage cricket fans by providing live scores, commentary, and highlights through the app.

Overall, the Cricket Scorer App aims to enhance the cricket experience for players, coaches, and fans by simplifying the scoring process, providing accurate data, and fostering engagement with the game.

**Importance of Cricket Scorer App**

In summary, the Cricket Scorer App is of paramount importance in the world of cricket for the following reasons:

* Efficiency: It streamlines scoring processes, making them faster and more efficient.
* Accuracy: Reduces errors in scoring, ensuring precise match data.
* Real-Time Information: Provides live updates, keeping players, coaches, and fans informed.
* Customization: Adaptable to various cricket formats, Such as IPL, T20, ODI.
* Cost Savings: Reduces production and distribution costs of printed scorecards.
* Offline Capabilities: Supports scoring in areas with no internet connectivity.

Overall, the Cricket Scorer App enhances the cricket experience, promotes accuracy, and contributes to the growth and development of the sport.

**Algorithm: CricketScoringApp**

1. Constants and Global Variables:

- Define PASSWORD as "bottleOpener"

- Define other global variables for scoring and game status

2. Main Function:

- Display an admin access prompt.

- Get the admin's name and validate access using adminValidation().

- Open a file for admin access logging.

- Display a welcome message for the Cricket App.

- Get team names and total overs from the user.

- Call dataCollect() to start the scoring process.

3. adminValidation Function:

- Get the password from the admin securely without echoing.

- Compare the entered password with the predefined PASSWORD.

- If the passwords match, authorize the user; otherwise, prompt for the password again.

4. dataCollect Function:

- Recursively call itself until the match is over:

- Get input for each ball: 0 for wrong ball, 1 for the right ball.

- Process the ball based on user input:

- If it's a wrong ball (0), handle wide or no-ball with relevant comments.

- If it's the right ball (1), get input for wicket status (w for wicket, n for no wicket).

- Update scores, wickets, and check for the end of the over.

- If the inning is over, call dataCreate() to log data.

- Display match result if both innings are completed.

5. dataCreate Function:

- Open a file for store scored data.

- Log current inning details, total overs, team names, overs, balls, wickets and current score into ‘score.txt’ file.

- If the match is over, display the match result.

- Close the file.

6. displayData Function:

- Open the ‘score.txt’ file for reading.

- Move to the specified position in the file.

- Display the data from that position until the end of the file.

- Close the file.

7. isOver Function:

- Check if the over is completed (ball reaches 6).

- If the total over is reached, start the second inning.

- Increment the inning counter.

8. wicketAll Function:

- Check if all wickets are down (wicket reaches 10).

- If all wickets are down, start the next inning.

- Reset over, score, and wicket counters.

9. secondInning Function:

- Call dataCreate() to log data for the first inning.

- Swap team names for the second inning.

- Display a message indicating the start of the second inning.

10. getScore Function:

- Get input for the run scored.

- Validate the input to be within a valid range (0 to 6).

- Update the global score variable.

11. randomNum Function:

- Generate a random number between 0 and 2 for comments.

- Return the random number.

12. transformLower Function:

- Convert a string to lowercase.

- Return the transformed string.

13. End of Algorithm.

**SYSTEM FLOWCHART**

Start

dataCollect();

Switch Ball Type

End

**If (Inning > 2)**

dataCreate();

return 0;

**True**

getScore();

Case 0

Case 1

**False**

**True**

**True**

Case 0 Message Display

Case 1 Message Display

dataCreate();

dataCollect();

**False**

**Source Code**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <time.h>

#include <termios.h>

#include <unistd.h> // for STDIN\_FILENO

#define PASSWORD "bottleOpener"

// imp functions

void adminValidation();

void dataCreate(char matchOver);

int dataCollect();

void displayData();

// utilities functions

void isOver();

void wicketAll();

void secondInning();

void getScore(int \*tempRunPtr);

int randomNum();

char \*transformLower(char \*string);

// Global Variables

int score = 0;

int wicket = 0;

int over = 0;

int ball = 0;

int inning = 1;

int totalOver;

int team1run;

char team1[40], team2[40];

// pointers

FILE \*appendAndReadFp = NULL;

int main()

{

char adminAccessUser[40];

printf("-------- ADMIN ACCESS REQUIRED --------\n");

printf("Enter your Name: ");

fgets(adminAccessUser, 40, stdin);

adminValidation();

appendAndReadFp = fopen("adminLogin.csv", "a");

fprintf(appendAndReadFp,"Date of Access,Name \n");

time\_t current\_time;

time(&current\_time);

char \*dateAndTime = ctime(&current\_time);

dateAndTime[strlen(dateAndTime) -1] = '\0';

fprintf(appendAndReadFp, "%s,%s", dateAndTime, adminAccessUser);

fclose(appendAndReadFp);

printf("\n------------------ Cricket App ------------------\n\n");

printf("Enter the Batsmans team Name: ");

scanf("%s", &team1);

printf("Enter the Bowlers team Name: ");

scanf("%s", &team2);

printf("Enter the total overs: ");

scanf("%d", &totalOver);

dataCollect();

return 0;

}

void adminValidation()

{

char password[14];

char ch;

struct termios old\_settings, new\_settings;

tcgetattr(STDIN\_FILENO, &old\_settings);

new\_settings = old\_settings;

new\_settings.c\_lflag &= ~(ICANON | ECHO);

tcsetattr(STDIN\_FILENO, TCSANOW, &new\_settings);

printf("Enter Password:\t");

for (int i = 0; i < 12; i++)

{

ch = getc(stdin);

password[i] = ch;

printf("\*");

}

password[12] = '\0';

tcsetattr(STDIN\_FILENO, TCSANOW, &old\_settings);

if (strcmp(password, PASSWORD) != 0)

{

printf("\nWrong Password Please Try Again \n\n");

adminValidation();

}

else printf("\n --- USER AUTHORIZED --- \n\n");

}

int dataCollect()

{

char tempBallC[2];

int tempBall;

char ballStatus[2];

char tempWicketStatus[2];

int tempRun;

while (!(strcmp(tempBallC, "0") == 0 || strcmp(tempBallC, "1") == 0))

{

printf("\n\n Enter ball (0 for wrong ball) (1 for right ball): ");

scanf("%s", &tempBallC);

}

tempBall = atoi(tempBallC);

switch (tempBall)

{

case 0:

{

char wideBallComment[][100] = {

"!!! That one slipped way down the leg side. Wide ball called by the umpire !!!",

"!!! The bowler loses control there, and it's a wide ball. Extra run for the batting side !!!",

"!!! A wide delivery outside off-stump. The batsman doesn't have to play at that !!!"};

char noBallComment[][105] = {

"!!! Oh, that's a costly mistake by the bowler. No-ball! The batsman gets a free hit now !!!",

"!!! The bowler overstepped, and it's a no-ball. The batsman will be looking to capitalize on this !!!",

"!!! A no-ball! That's not what the bowling side needed. Extra delivery and a free hit coming up !!!"};

int randomIndex = randomNum();

while (!(strcmp(transformLower(ballStatus), "w") == 0 || strcmp(transformLower(ballStatus), "n") == 0))

{

printf("\n Enter wide (W) or No Ball(N): ");

scanf("%s", &ballStatus);

}

getScore(&tempRun);

if (strcmp(ballStatus, "w") == 0)

{

printf("\n\n COMMENT:- %s", wideBallComment[randomIndex]);

}

else

{

printf("\n\n COMMENT:- %s", noBallComment[randomIndex]);

}

printf("\n\n");

}

break;

case 1:

{

while (!(strcmp(transformLower(tempWicketStatus), "w") == 0 || strcmp(transformLower(tempWicketStatus), "n") == 0))

{

printf("\n (W) for wicket \t (N) for no wicket : ");

scanf("%s", &tempWicketStatus);

}

ball++;

if (strcmp(transformLower(tempWicketStatus), "w") == 0)

{

printf("\n\n Comment:- What a loss for team %s", team1);

int a = randomNum();

wicket++;

wicketAll();

}

getScore(&tempRun);

isOver();

}

break;

default:

printf("\n\n Invalid input");

break;

}

if (inning > 2)

{

inning--;

dataCreate('y');

inning = 0;

return 0;

}

else

{

dataCreate('n');

return dataCollect();

}

}

void dataCreate(char matchOver)

{

long currentPosition;

appendAndReadFp = fopen("score.txt", "a+");

if (appendAndReadFp == NULL)

{

printf("Unable to open the file.\n");

exit(1);

}

fprintf(appendAndReadFp, "\n");

currentPosition = ftell(appendAndReadFp);

fprintf(appendAndReadFp, "\t Current Inning -----> %d\n", inning);

fprintf(appendAndReadFp, "\t Total Over To play -> %d \n", totalOver);

fprintf(appendAndReadFp, "\t %s (batsman) Vs %s (bowler) \n", team1, team2);

fprintf(appendAndReadFp, "\t %d Runs In %d.%d over with %d wickets \n", score, over, ball, wicket);

if (matchOver == 'y')

{

fprintf(appendAndReadFp, "\n\t --------------- Match Result --------------- \n");

if (team1run > score)

{

fprintf(appendAndReadFp, "\n\t %s Won the match with %d runs \n", team2, team1run - score);

}

else if (team1run == score)

{

fprintf(appendAndReadFp, "\n\t Match Draw \n");

}

else

{

fprintf(appendAndReadFp, "\n\t %s Won the match with %d wickets \n", team1, 10 - wicket);

}

fprintf(appendAndReadFp, "\n\t -------------------------------------------- \n");

}

fclose(appendAndReadFp);

displayData(currentPosition);

}

void displayData(long currentPosition)

{

printf("\n");

appendAndReadFp = fopen("score.txt", "r");

if (appendAndReadFp == NULL)

{

printf("Unable to open the file.\n");

exit(1);

}

fseek(appendAndReadFp, currentPosition, SEEK\_SET);

printf("\n\t--- Displaying Data ---\n\n");

char ch;

while ((ch = fgetc(appendAndReadFp)) != EOF)

{

putc(ch, stdout);

}

fclose(appendAndReadFp);

}

void isOver()

{

if (ball == 6)

{

over++;

ball = 0;

if (over == totalOver)

{

inning++;

inning == 2 ? secondInning() : 0;

}

}

}

void wicketAll() {

if (wicket == 10) {

inning++;

inning == 2 ? secondInning() : 0;

over = 0;

score = 0;

wicket = 0; } }

void secondInning()

{

dataCreate('n');

char temp[20];

strcpy(temp, team1);

strcpy(team1, team2);

strcpy(team2, temp);

printf("\n\n----------Second Inning Started----------\n\n");

team1run = score;

over = 0;

score = 0;

wicket = 0;

}

void getScore(int \*tempRunPtr)

{

printf("\n Run scored: ");

scanf("%d", tempRunPtr);

if (\*tempRunPtr > 7)

{

printf("\n Invalid run Entered, Enter Run scored:) -> ");

scanf("%d", tempRunPtr);

}

score += \*tempRunPtr;

}

int randomNum()

{

srand(time(NULL));

int randomNum = rand() % 3;

return randomNum;

}

char \*transformLower(char \*string)

{

int i = 0;

while (string[i] != '\0')

{

if (string[i] >= 'A' && string[i] <= 'Z') string[i] |= 32;

i++;

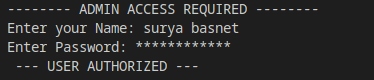
}

return string;

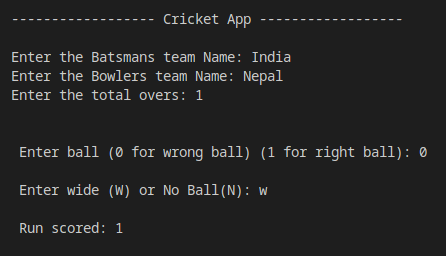
}

**Output of Program**

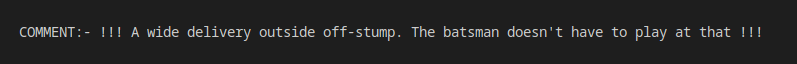
1. Authenticating user



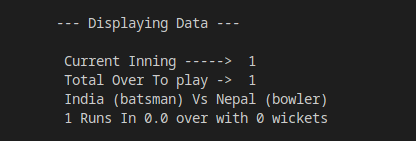
1. Inserting ball and score Records



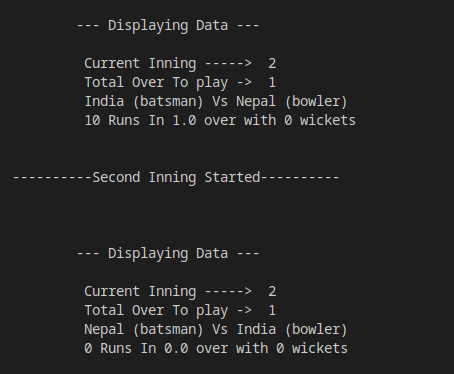
1. Comments by commenter



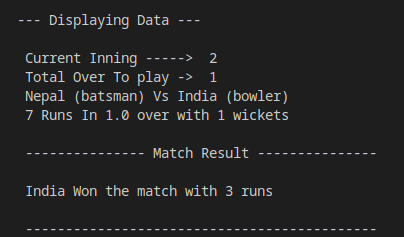
1. Displaying Data



1. On the end of first inning (Dummy Data)



1. In the end of second inning (Dummy Data)



**Abstract**

We, a committed team of developers, are pleased to introduce our project, the "Cricket Scorer App," which was created completely in the C programming language in response to the current needs of cricket scoring and data management within the cricketing community. Utilizing the strength and adaptability of C, this project solves the urgent demand for an updated and effective cricket scoring solution.

Our Cricket Scorer App is built on the C programming language due to its low-level memory access, minimum runtime support requirements, and powerful file handling features. These characteristics make C the perfect platform for creating applications that require accurate record-keeping and real-time data updates, which are crucial in cricket scoring.

Our Cricket Scorer App is a complete solution made to meet the many requirements of cricket fans, including players, coaches, officials, and supporters. It has a simple user interface and strong live scoring, player statistics, and match administration capabilities that were all created in C to assure accurate and effective performance.

The software is painstakingly designed to simplify the scoring procedure, lowering the possibility of mistakes and offering immediate access to live scores and match information. It has tools for tracking historical data, analyzing player performance, and engaging fans with highlights and live commentary. Users can easily record information about cricket matches with this C-based program, assuring accurate and current records.

Our objective is to promote accuracy and fair play in cricket while streamlining scoring and data administration. We have developed a reliable and effective cricket scoring tool that strengthens community relationships by utilizing the power of the C programming language. With the Cricket Scorer App, we embrace technology's ability to improve the cricketing experience while celebrating the game's timeless character.

**Future Use and Implementation of Cricket Scorer App**

The Cricket Scorer App, built using the versatile C programming language, holds significant potential for future use and implementation in various cricketing contexts:

* Local Cricket Clubs and Leagues: Local cricket clubs and leagues can adopt the app to streamline their scoring processes. It offers an efficient way to score matches and generate match reports. This can enhance the overall experience for players and fans.
* Amateur and School Cricket: The app can be introduced in school and amateur cricket settings. Coaches and teachers can use it to teach scoring techniques to aspiring cricketers while maintaining records of school or club matches.
* Tournament Management: Cricket tournaments, especially at the grassroots level, can benefit from the app. It can be integrated into tournament management systems to provide real-time scores and statistics to participants and spectators.
* Digital Scoreboards: The app can be integrated with digital scoreboards in stadiums, providing live scores and statistics to spectators attending matches.
* Data Store and Export: The data collected by the app can be exported or stored in data storage unit.
* Broadcasting and Commentary: The app can provide real-time data feeds for broadcasters and commentators, enhancing their ability to provide insightful commentary during live broadcasts.

In conclusion, the Cricket Scorer App has the potential to revolutionize cricket scoring and data management at all levels of the game. Its versatility, accuracy, and user-friendly interface make it a valuable tool for cricket enthusiasts and stakeholders, paving the way for the future of cricket scoring and analysis.

**Conclusion**

Upon the successful implementation of the "Cricket Scorer App," a revolutionary solution for cricket scoring and data management, many challenges faced by cricket enthusiasts and organizations will be addressed comprehensively. This app introduces a modern approach to cricket scoring, ushering in a new era for the sport. For example, for Efficient Scoring, Player Performance Analysis, User-Friendly Interface, Real-Time Updates, etc.

In conclusion, the "Cricket Scorer App" is not just a scoring tool; it's a game-changer for cricket. It modernizes scoring, empowers users with data, and enhances the cricketing experience for all stakeholders. As it becomes an integral part of the cricketing world, many challenges associated with traditional scoring methods will be overcome, ushering in a new era of cricket scoring and enjoyment.

**Bibliography**

1. Books:

* Author: **Perry Grey & Dean Miller**
* Title of the book: C Programming Absolute Beginner’s Guide
* Publication year: 22 August, 2013
* Publisher: ‘Que’

1. Web Sources:**o**

File Pointer Location https://www.tutorialspoint.com/c\_standard\_library/c\_function\_ftell.htm