|  |
| --- |
| **Cricket score card**  **21CSS101J – PROGRAMMING FOR PROBLEM-SOLVING**  **Mini Project Report**  *Submitted by*  **Harsh Kumar [Reg. No.: RA2311027010065]**  **B.Tech. CSE - DSBS**  **SRMIST-01.jpg**  **SCHOOL OF COMPUTING**  **COLLEGE OF ENGINEERING AND TECHNOLOGY**  **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**  **(Under Section 3 of UGC Act, 1956)**  S.R.M. NAGAR, KATTANKULATHUR – 603 203  CHENGALPATTU DISTRICT  **November 2023**  **SRMIST-01.jpg**  **BONAFIDE CERTIFICATE**  Certified that Mini project report titled **Cricket score card** is the bonafide work of RA23110270100065 Harsh Kumar who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.  **SIGNATURE SIGNATURE**  **(GUIDE) (HEAD OF THE DEPARTMENT)** |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S No.** | **Title** | **Page No.** |
| 1 | Problem Statement | 1 |
| 2 | Methodology | 2 |
| 3 | Flowchart | 3 |
| 4 | Coding (C) | 4 |
| 5 | Result | 11 |
| 6 | Conclusion | 15 |
| 7 | References | 16 |

**Problem Statement**

The Cricket Score Sheet project is a simple application written in the C programming language. It employs file management to store data like as runs, wickets, overs, and extras, among other things. The application may show runs, wickets, batsman and bowler names, overs, extras, bowler economy, batsman strike rate, and so on. It also shows the game's start and end times. The source code is comprehensive, devoid of errors, and simple to comprehend.

The program will ask the user to press "e" to edit the information and "c" to continue after entering these facts. The application will ask for the file name once the user selects "2" from the main menu. If the file is found, it will be displayed. If not, an error message will appear on the screen. The third option in the main menu is Exit. If the number "3" is entered in the main menu, the Cricket Score Sheet project is terminated.

1

**Methodology**

The following actions are carried out by the cricket score sheet project file when it is run:

The welcome screen appears at the start of the project and fades up to reveal the main menu. On the main menu, there are three options:

* Make a fresh score sheet.
* Check out the previous score sheet.
* Exit

Project Cricket Score Sheet will prompt for the name of the new score if the number '1' is entered. A notification will appear on the screen after the file is created. The user must then complete a scoring sheet which requires the following details:

* Competition\Venue
* Between and against the winning Toss team
* Draw winner chosen
* Date and shift
* The name of each batsman and the runs scored by each
* Each blower listing the pitcher's name and run

This project's C source code is incredibly basic, with just seven user supplied functions. They're mentioned here, along with the jobs they're responsible for.

* void date() – to store current date or date of game.
* void print() – to print the output in specific format.
* void filewrite() – to write in a file on hard drive of computer to store the input data such as runs, wickets, balls, over etc.
* void fileread() – to extract or read the data from the file created to store the data.
* void fileopen(char) – opens a cricket score sheet project file from the computer.
* int limitedinput(int) – to input some limited data.
* void newscoresheet() – to create new score sheet in a new file.

2

**Flow chart**

START

ENTER THE DETAILS OF THE BATSMAN AND BOWLERS

ENTER THE RUNS AND WICKETS TAKEN BY THE BATSMAN AND BOWLER

THE TOTAL RUNS AND WICKETS TAKEN BY THE BATSMAN AND BOWLERS IN THE COMPLETE MATCH WILL BE DISPLAYED AS A TABLE

STOP

3

**Coding (C)**

* C programming language

C programming is a fundamental skill for aspiring software developers, particularly those interested in system programming, embedded systems, and game development. While it demands dedication and perseverance to master, C programming provides a comprehensive understanding of programming concepts and low-level system interactions.

1. Data Types and Variables: C offers a variety of data types, including integers, floating- point numbers, characters, and pointers. Variables serve as storage locations for these data types.
2. Operators: C provides a rich set of operators, including arithmetic, relational, logical, bitwise, and assignment operators, enabling manipulation of data.
3. Control Flow Statements: C employs control flow statements like 'if', 'else', 'switch', 'while', and 'for' loops to control the execution flow of a program.
4. Functions: Functions are modular blocks of code that perform specific tasks, promoting code reusability and organization.
5. Arrays: Arrays allow storing multiple elements of the same data type in a contiguous memory block.
6. Pointers: Pointers provide the capability to access and modify memory locations directly, enabling dynamic memory allocation and efficient data manipulation.
7. Structures: Structures group related data items into a single unit, enabling encapsulation and data organization.
8. Input/Output Operations: C provides functions for reading input from the keyboard and writing output to the console or files.
9. File Handling: C allows opening, reading, writing, and closing files, essential for data persistence and storage.
10. Memory Management: C's dynamic memory allocation techniques, such as 'malloc' and 'free', enable allocation and deallocation of memory during program execution.

Mastering C programming requires hands-on practice and exploration of these concepts. Various resources, including tutorials, books, and online courses, can guide learners through the process.

4

4

**Code**

#include <stdio.h>

#include <stdlib.h>

**struct** batsman

{

**char** name[25];

**int** runs, score, balls, toruns, tobal, ones, twos, threes, fours, sixes;

**int** max\_six, max\_run, max\_four;

**float** str;

} pl1[100], pl3;

**struct** bowler

{

**char** name[25];

**int** runsgv, wkttkn, overs;

**int** max\_w;

**float** econ;

} pl2[100], pl4;

**int** main()

{

**int** plno, choice;

**int** i, n, m;

    printf("Enter the Batsman detail:\n");

    printf("Enter the number of batsman:\n");

    scanf("%d", &m);

**for** (i = 0; i < m; i++)

    {

        printf("Enter name of batsman%d:\n", i + 1);

        scanf("%s", pl1[i].name);

        printf("Enter the number of ones scored by player%d:\n ", i + 1);

        scanf("%d", &pl1[i].ones);

        printf("Enter the number of twos scored by player%d:\n ", i + 1);

        scanf("%d", &pl1[i].twos);

        printf("Enter the number of threes scored by player%d:\n ", i + 1);

        scanf("%d", &pl1[i].threes);

5

        printf("Enter the number of fours scored by player%d:\n ", i + 1);

        scanf("%d", &pl1[i].fours);

        printf("Enter the number of sixes scored by player%d:\n ", i + 1);

        scanf("%d", &pl1[i].sixes);

        printf("Enter the balls played by the player%d:\n", i + 1);

        scanf("%d", &pl1[i].balls);

    }

    printf("\nEnter the bowlers details:\n");

    printf("Enter the number of bowlers:\n");

    scanf("%d", &n);

**for** (i = 0; i < n; i++)

    {

        printf("\nEnter name of bowler%d:", i + 1);

        scanf("%s", pl2[i].name);

        printf("Enter the runs given by the bowler%d:\n ", i + 1);

        scanf("%d", &pl2[i].runsgv);

        printf("Enter the overs bowled by the bowler%d:\n", i + 1);

        scanf("%d", &pl2[i].overs);

        printf("Enter the wickets taken by the bowler%d\n", i + 1);

        scanf("%d", &pl2[i].wkttkn);

    }

    printf("Thank you all details are recorded\n");

**do**

    {

6

        printf("Enter the choice:\n 1)Batsman detail:\n 2)Bowlers detail:\n 3)Match summary:\n 4)Record:\n 5)Exit\n ");

        scanf("%d", &choice);

**switch** (choice)

        {

**case** 1:

            printf("Enter the batsman number to see his details\n");

            scanf("%d", &plno);

            plno--;

            printf("                       Player Detail\n");

            printf("= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =\n");

            printf(" Batsman        runs           balls        fours       sixes         sr   \n");

            printf("= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =\n");

            pl1[plno].runs = (1 \* pl1[plno].ones) + (2 \* pl1[plno].twos) + (3 \* pl1[plno].threes) + (4 \* pl1[plno].fours) + (6 \* pl1[plno].sixes);

            pl1[plno].str = (pl1[plno].runs \* 100.00) / pl1[plno].balls;

            printf(" %-15s %-14d %-13d %-11d %-11d %-9.2f\n\n", pl1[plno].name, pl1[plno].runs, pl1[plno].balls, pl1[plno].fours, pl1[plno].sixes, pl1[plno].str);

**break**;

**case** 2:

            printf("Enter the bowlers number to see his details\n");

            scanf("%d", &plno);

            plno--;

            printf("                         Player Detail\n  ");

printf(" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = \n");

            printf(" Bowler        overs           runs        wicket       economy\n");

7

            printf(" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = \n");

**for** (i = 0; i < n; i++)

            {

                pl2[plno].econ = pl2[plno].runsgv / pl2[plno].overs;

                printf(" %-15s %-14d %-13d %-11d %-11.2f\n\n", pl2[plno].name, pl2[plno].overs, pl2[plno].runsgv, pl2[plno].wkttkn, pl2[plno].econ);

            }

**break**;

**case** 3:

            printf("                     Match summary\n");

            printf(" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = \n");

            printf(" Batsman        runs           balls        fours       sixes         sr   \n");

            printf(" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = \n");

**for** (i = 0; i < 1; i++)

            {

                pl1[i].runs = (1 \* pl1[i].ones) + (2 \* pl1[i].twos) + (3 \* pl1[i].threes) + (4 \* pl1[i].fours) + (6 \* pl1[i].sixes);

                pl3.toruns += pl1[i].runs;

                pl1[i].str = (pl1[i].runs \* 100.00) / pl1[i].balls;

                printf(" %-15s %-14d %-13d %-11d %-11d %-9.2f\n\n", pl1[i].name, pl1[i].runs, pl1[i].balls, pl1[i].fours, pl1[i].sixes, pl1[i].str);

            }

            printf("TOTAL RUNS:%d\n\n", pl3.toruns);

            printf("\n\n");

            printf("= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =  = = = = = = = = =  = = = = = \n");

            printf(" Bowler        overs           runs        wicket       economy\n");

            printf(" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =\n");

**for** (i = 0; i < n; i++)

            {

8

                pl2[i].econ = pl2[i].runsgv / pl2[i].overs;

                printf(" %-15s %-14d %-13d %-11d %-11.2f\n\n\n", pl2[i].name, pl2[i].overs, pl2[i].runsgv, pl2[i].wkttkn, pl2[i].econ);

            }

**break**;

**case** 4:

            pl3.max\_run = 0, pl4.max\_w = 0, pl3.max\_four = 0, pl3.max\_six = 0;

**for** (i = 0; i < m; i++)

            {

                pl1[i].runs = (1 \* pl1[i].ones) + (2 \* pl1[i].twos) + (3 \* pl1[i].threes) + (4 \* pl1[i].fours) + (6 \* pl1[i].sixes);

**if** (pl3.max\_run < pl1[i].runs)

                {

                    pl3.max\_run = pl1[i].runs;

                }

**if** (pl3.max\_six < pl1[i].sixes)

                {

                    pl3.max\_six = pl1[i].sixes;

                }

**if** (pl3.max\_four < pl1[i].fours)

                {

                    pl3.max\_four = pl1[i].fours;

                }

**if** (pl4.max\_w < pl2[i].wkttkn)

                {

                    pl4.max\_w = pl2[i].wkttkn;

                }

            }

            printf("Highest runs scored by the batsman:%d\n", pl3.max\_run);

9

            printf("Maximum fours scored by the batsman:%d\n", pl3.max\_four);

            printf("Maximum sixes scored by the batsman%d:\n", pl3.max\_six);

            printf("Maximum wickets taken by the bowler:%d\n", pl4.max\_w);

**break**;

**case** 5:

            exit(1);

**default**:

            printf("Enter the correct choice\n");

**break**;

        }

    } **while** (choice != 5);

**return** 0;

}

**Result**

10

**Output :**

Enter the Batsman detail:

Enter the number of batsman:

4

Enter name of batsman1:

Rohit

Enter the number of ones scored by player1:

1

Enter the number of twos scored by player1:

2

Enter the number of threes scored by player1:

3

Enter the number of fours scored by player1:

4

Enter the number of sixes scored by player1:

6

Enter the balls played by the player1:

50

Enter name of batsman2:

Virat

Enter the number of ones scored by player2:

1

Enter the number of twos scored by player2:

4

Enter the number of threes scored by player2:

2

Enter the number of fours scored by player2:

2

Enter the number of sixes scored by player2:

1

Enter the balls played by the player2:

33

Enter name of batsman3:

11

Hardik

Enter the number of ones scored by player3:

4

Enter the number of twos scored by player3:

1

Enter the number of threes scored by player3:

1

Enter the number of fours scored by player3:

1

Enter the number of sixes scored by player3:

1

Enter the balls played by the player3:

35

Enter name of batsman4:

Shikhar

Enter the number of ones scored by player4:

1

Enter the number of twos scored by player4:

4

Enter the number of threes scored by player4:

2

Enter the number of fours scored by player4:

2

Enter the number of sixes scored by player4:

2

Enter the balls played by the player4:

24

Enter the bowlers details:

Enter the number of bowlers:

2

Enter name of bowler1:Kunal

Enter the runs given by the bowler1:

23

Enter the overs bowled by the bowler1:

12

3

Enter the wickets taken by the bowler1

0

Enter name of bowler2:Jadeja

Enter the runs given by the bowler2:

43

Enter the overs bowled by the bowler2:

20

Enter the wickets taken by the bowler2

0

Thank you all details are recorded

Enter the choice:

1)Batsman detail:

2)Bowlers detail:

3)Match summary:

4)Record:

5)Exit

Enter the batsman number to see his details

1

**A white paper with black text

Description automatically generated**

13

**A screenshot of a sports report

Description automatically generated**

14

Conclusion

In this mini-project, we have successfully developed a C program to manage cricket scorecards. The program allows users to create new scorecards, enter innings information, track runs, wickets, overs, and extras, and display the current scorecard summary. The program also calculates additional statistics such as batsman strike rate and bowler economy.

The program was developed using the C programming language, which is a powerful and versatile language that is well-suited for developing simple applications. The program employs various concepts of C programming, including variables, arrays, pointers, functions, structures, and file handling.

The development of this program has provided us with valuable experience in programming and problem-solving. We have learned how to design and implement a data structure to store cricket scorecard information, how to write functions to manipulate the data, and how to interact with the user through a user interface.

We believe that this program is a valuable learning tool for anyone interested in learning C programming or in developing cricket scorecard management applications. The program is relatively simple to understand and modify, and it can be easily extended to include additional features such as support for multiple matches and tournaments.

Future Works

The current version of the program focuses on managing basic scorecard information. In the future, we plan to extend the program to include additional features such as:

1. Support for multiple matches and tournaments
2. Graphical user interface (GUI) for improved user interaction
3. Live score updates during matches
4. Statistical analysis of match data

We believe that these enhancements will make the program more comprehensive and useful for cricket enthusiasts and scorekeepers.

Overall, we are satisfied with the outcome of this mini-project. We have learned a great deal about C programming and have developed a useful application that can be used to manage cricket scorecards. We encourage others to use and extend this program to further their own learning and development.

References

15

<https://bard.google.com/chat>

https://www.javatpoint.com/cricket-score-sheet-in-c

<https://www.freebookcentre.net/Language/Free-C-Programming-Books-Download-1.htm>

https://freecomputerbooks.com/C-Programming-Wikibook.html

16