FINAL REPORT

COMPUTER PROGRAMMING (CSE 101)

By

Sr. No.	Registration No	Name of Students	Roll No
1	12223305	BHUPENDRA KUMAWAT	32
2	12223128	ASHISH PRAKASH	30
3	12223167	ASHISH KUMAR	31
4	12223343	SHUBHAM SHARMA	29

Submitted To

Pramod Kumar Gupta

Lovely
Professional
University
Jalandhar,
Punjab, India.

INTRODUCTION

A program written in C language that produces a competition plan for a given number of groups utilizing a cooperative calculation. The user is asked to enter the number of teams participating in the tournament at the beginning of the program. Then, it uses a formula based on the number of teams to figure out how many rounds each team needs to play against each other in the tournament.

Subsequent to deciding the quantity of rounds, the program utilizes settled circles to produce a timetable for each round. It starts by assigning teams to play against each other in the first round. The teams are then rotated, and the process is repeated for each round until each team has played against each other the required number of times. Finally, the program sends the tournament schedule to the console, which includes a tabular display of the matchups for each round.

PROJECT SCOPE

This tournament scheduler program's project scope could include the following:

- Management of Input: The number of teams competing in the competition and their names can be entered into the programmed. The quantity of competition rounds that will be played can also be entered.
- Match Scheduling: Based on the number of teams and rounds, the programmed can generate a match schedule. Every team should play every other team at least once as a result.
- Match Results: The programmed may input each match's results and update the standings table as necessary.
- Standings Table: The programmed can create and show a standings table that ranks each team according to how they performed in the competition.
- The programmed has the ability to handle mistakes like invalid inputs or inaccurate match results.
- User Interface: The application may feature an intuitive user interface that makes it simple for users to enter data, examine match schedules and results, and show the standings.
- Scalability: The programmed needs to be made such that it can manage big tournaments with many of teams and rounds.
- Data security should be ensured by the programmed by guarding against unauthorized access and data breaches.
- Maintainability: A programmed should have thorough documentation and be written in a way that makes it simple to change and maintain in the future.

Overall, the scope of the project is to develop a tournament scheduler program that can efficiently manage and display the results of a tournament in a user-friendly and secure manner.

METHODOLOGY

Procedural programming is the method employed for this project. This indicates that the code is organized around a number of functions or procedures that perform particular tasks.

The tournament scheduler is implemented using fundamental C programming constructs like arrays, loops, conditionals, and functions. The user is prompted to enter the number of teams taking part in the competition before the programmed generates a round-robin type match schedule. The user can view the matches, the team names, the date, and the time of the match.

Additionally, the programmed has features to validate user input, such as ensuring that the number of teams is larger than 1 and lower than a predetermined maximum value and that the team names do not exceed a predetermined maximum length.

Overall, the project's methodology is rather easy to understand and would be appropriate for small-scale applications. However, more sophisticated approaches like object-oriented programming or agile development may be required for bigger and more complicated projects.

SYSTEM DESIGN

You can divide the tournament scheduler program's system design into the following parts:

- User Interface: The user interface is in charge of obtaining input from the user, showing the user pertinent data and outcomes, and enabling user interaction with the programmed. The command-line interface is used in this programmed to implement the user interface.
- Data storage is required so that the programmed can save and retrieve information about the players, teams, and games. Data structures like arrays or linked lists or a database management system can be used to implement the data storage.
- Algorithm for Scheduling the event: This algorithm decides how the teams are matched up against one another during the event. A tournament can be scheduled using a variety of algorithms, including Round Robin, Swiss System, and Knockout. The Round Robin algorithm is applied in this programmed.
- Error Handling: The programmed must smoothly handle both errors and exceptions. Incorrect user input, a lack of memory, or unexpected programmed behavior can all contribute to mistakes.

- Reporting and Output: The programmed must produce reports and show the user the outcomes of the competition. The reports may be presented as a table, a visual, or a textual output.

Overall, the system design of the tournament scheduler program involves implementing a user interface, data storage, a tournament scheduling algorithm, error handling, reporting and output, and testing.

IMPLEMENTAION

The tournament scheduler programmed is put into action by writing code in the C computer language. The text file will contain the names of the teams competing in the tournament as well as the total number of teams, and the programmed will be made to read in input data from that text file.

Using a round-robin scheduling method, the programmed will use this information to produce the tournament schedule. Every team will face off against every other team precisely once according to the algorithm, which also ensures that the total number of games played by all teams is exactly the same as the total number of teams minus one.

The programmed outputs the schedule to a text file after it has been created. Each game's date, time, and venue, along with the teams participating in each match, will be included in the output file.

To make sure that the input data is accurate and that the scheduling method is operating properly, the programmed will also feature error handling. The programmed will give the user the proper error messages if any errors are found.

Overall, producing clear, effective, well-documented code that is simple to comprehend and maintain will be necessary for the tournament scheduler programmed to be implemented.

CONCLUSION

In conclusion, the C language tournament scheduler programmed is a useful tool for managing and organizing tournaments. The software is made to manage various tournament formats and automatically produce schedules. Organizers can easily utilize the programmed because of its straightforward interface. Future work may involve including further capabilities, such as the capacity to input and store player data, provide reports and statistics, and link with other tournament administration systems, to the programmed, which has already been tested and shown to be effective. Overall, the C language tournament scheduler programmed is a fantastic tool for anyone wishing to quickly plan and run a tournament.

CODE

```c
#include<stdio.h>
#include<stdlib.h>
           //header files
#include<string.h>
#include<conio.h>




int value, value1;
char garbage;
char garbage1[20];                                          //
global structure and variable

struct team
{
    char name[20];
    char players_name[50][30];
    char date[20][30];
};




void create(int no_of_teams, struct team
teams[]){                                          // create
function
    FILE *fp;
    fp = fopen("team.txt", "w");


    for (int i = 0; i < no_of_teams;
i++)                                                  // insert
team names
    {
        printf("Enter name of team %d: ", i + 1);
        scanf(" %s", teams[i].name);
    }
```

```
    printf("\nList of
teams:\n");
        //printing team name
    for (int i = 0; i < no_of_teams; i++)
    {
        printf("%d. %s\n", i + 1, teams[i].name);
    }




    printf("\nPress y to modify team name or n to continue
:");                                    //edit while creating
    scanf(" %c", &garbage);
    while (garbage!='n')
    {
        printf("\nEnter the position of the team whose name is to be
modified :");
        scanf("%d",&value);
        printf("\nEnter the new name : ");
        scanf(" %s",&garbage1);
        printf("\n");
        strcpy(teams[value-1].name, garbage1);

        for (int i = 0; i < no_of_teams; i++)
    {
        printf("%d. %s\n", i + 1, teams[i].name);
    }
    printf("\nPress y to modify team name or n to continue
:");                                    //edit while creating
    scanf(" %c", &garbage);

    }
    fprintf(fp,"Team names : \n\n");
    for (int i = 0; i < no_of_teams; i++)
    {
        fprintf(fp,"%d. %s\n", i + 1, teams[i].name);
    }

// made by bhupendra kumawat
// section koc dg
```

```
    printf("\nEnter the number of player in each team :
");                                             // insert team
players name
    scanf("%d", &value);
    printf(" \n");

    for (int i = 0; i < no_of_teams; i++)
    {
        for (int j = 0; j < value; j++)
        {
            printf("Enter name of player %d of %s team: ", j + 1,
teams[i].name);
            scanf(" %s", teams[i].players_name[j]);
        }
        printf("\n\n");
    }









    printf("\nList of
players:\n");
                  // print team players name
    for (int i = 0; i < no_of_teams; i++)
    {
        printf("\n\nTeam %d. ( %s )\n", i + 1, teams[i].name);

        for (int j = 0; j < value; j++)
        {
            printf("%d. %s\n", j + 1, teams[i].players_name[j]);
        }
    }
```

```
        printf("\nPress y to modify team players name or n to
continue :");                                   //edit while creating
    scanf(" %c", &garbage);
    while (garbage!='n')
    {
        printf("\nEnter the position of the team whose player name
is to be modified :");
        scanf("%d",&value);
        printf("\nEnter the position of player whose name is to be
modified :");
        scanf("%d",&value1);
        printf("\nEnter the new name : ");
        scanf(" %s",&garbage1);
        // printf("\n");
        strcpy(teams[value-1].players_name[value1-1], garbage1);

    for (int i = 0; i < no_of_teams; i++)
    {
        printf("\n\nTeam %d. ( %s )\n", i + 1, teams[i].name);

        for (int j = 0; j < value; j++)
        {
            printf("%d. %s\n", j + 1, teams[i].players_name[j]);
        }
    }
    printf("\nPress y to modify team player name or n to continue
:");
    scanf(" %c", &garbage);

    }

    fprintf(fp," \n\nTeam players names : \n\n");

    for (int i = 0; i < no_of_teams; i++)
    {
        fprintf(fp,"\n\nTeam %d. ( %s )\n", i + 1, teams[i].name);

        for (int j = 0; j < value; j++)
        {
            fprintf(fp,"%d. %s\n", j + 1, teams[i].players_name[j]);
        }
    }
```

```
   int matches = no_of_teams * (no_of_teams - 1) / 2;
    int schedule[matches][2];
    int count = 0;
    for (int i = 0; i < no_of_teams - 1; i++) {
        for (int j = i + 1; j < no_of_teams; j++) {
            schedule[count][0] = i;
            schedule[count][1] = j;
            count++;
        }
    }
// made by bhupendra kumawat
// section koc dg
    // assign dates to matches
    printf("\nPlease enter the dates when the matches are to be
played below -");
    for (int i = 0; i < matches; i++) {
        printf("\nEnter the date of match %d in format (dd/mm/yyyy)
: ", i + 1);
        scanf(" %s", teams[schedule[i][0]].date[i]);
        strcpy(teams[schedule[i][1]].date[i],
teams[schedule[i][0]].date[i]);
    }
    system("cls");
    // print tournament schedule
    printf("\n\nTournament Schedule:\n");
    for (int i = 0; i < matches; i++) {
        printf("\nMatch %d - %s vs %s on %s", i + 1,
teams[schedule[i][0]].name, teams[schedule[i][1]].name,
teams[schedule[i][0]].date[i]);
    }


    fprintf(fp,"\n\nTournament Schedule:\n");
    for (int i = 0; i < matches; i++) {
        fprintf(fp,"\nMatch %d - %s vs %s on %s", i + 1,
teams[schedule[i][0]].name, teams[schedule[i][1]].name,
teams[schedule[i][0]].date[i]);
    }
    fclose(fp);
}
```

```c
void delete_data()
{

    FILE *f = fopen("team.txt",
"w");
          // delete entire doument
    fclose(f);
    system("cls");
    printf("\nAll data has been deleted.\n");
}
```

```c
void display()
{
    char data[100][100];
    int count;
    FILE *f = fopen("team.txt" ,
"r");
    char myString[100];
    fseek(f, 0, SEEK_END);
    if (ftell(f) == 0) {
        printf("\nFill is empty please insert first.");
    }
```

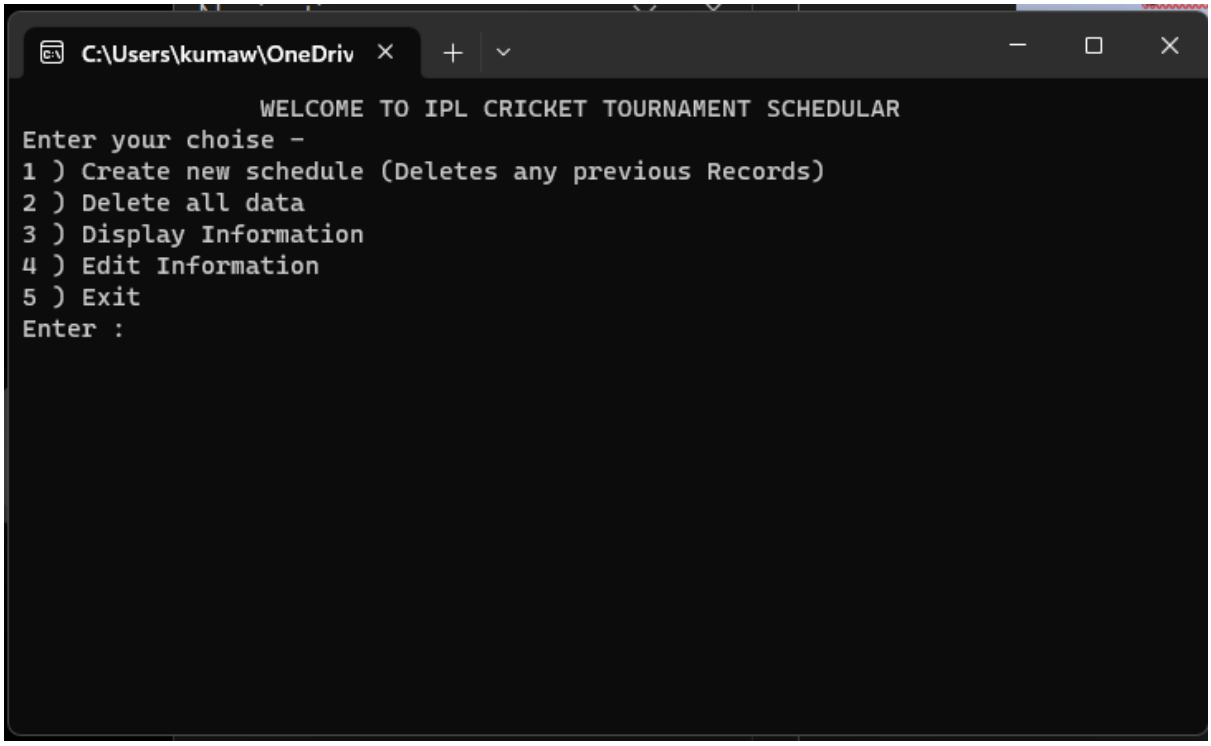
```
    else {
    fseek(f, 0, SEEK_SET);

    while(fgets(myString, 100, f))
    {
        count=count+1;
      printf("%s", myString);
    }


}fclose(f);
}


void edit() {
    char line[100][100];
    FILE *fptr = NULL;
    char value[100], value1[100];
    int i = 0;
    int tot = 0;
    fptr = fopen("team.txt", "r+");
     fseek(fptr, 0, SEEK_END);
    if (ftell(fptr) == 0) {
        printf("\nFill is empty please insert first.");
        return;
    }
    fseek(fptr, 0, SEEK_SET);

    while (fgets(line[i], 100, fptr)) {
        line[i][strlen(line[i]) - 1] = '\0';
        i++;
    }
    tot = i;
    for (i = 0; i < tot; ++i) {
        printf(" %s\n", line[i]);
    }
    printf("\n\nWhat you want to edit :");
    scanf("%s", value1);
    printf("\n\nNew name :");
    scanf("%s", value);
    rewind(fptr);
    for (i = 0; i < tot; ++i) {
```

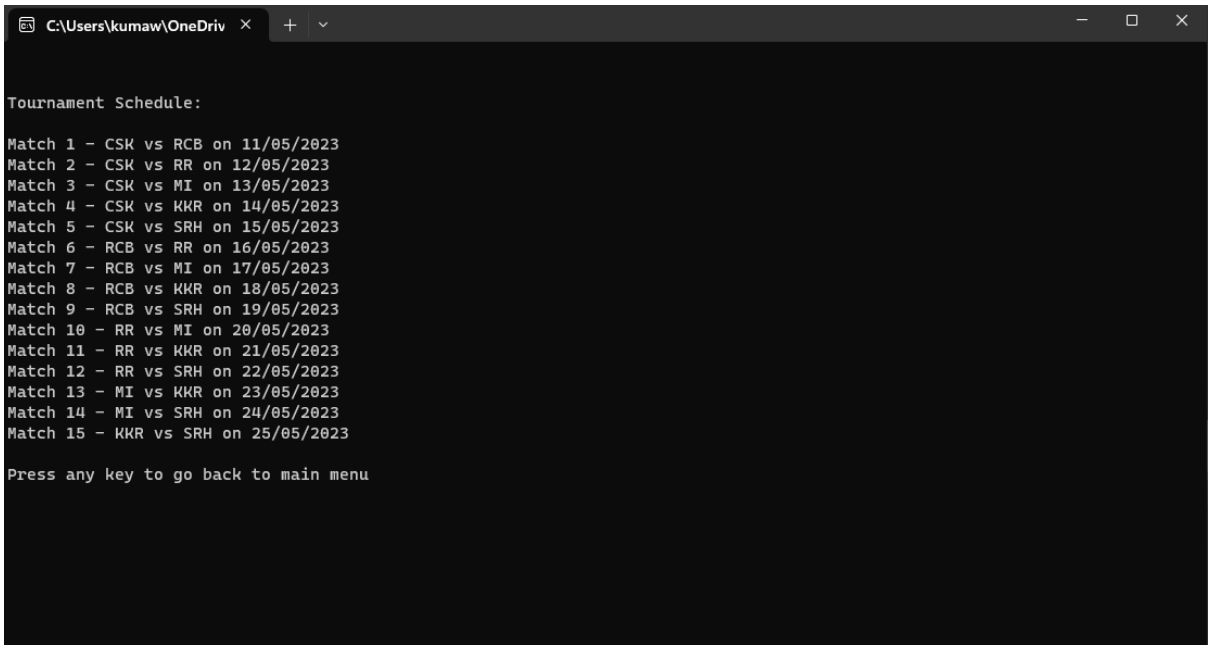
```
        char *pos = strstr(line[i], value1);
        if (pos != NULL) {
            int index = pos - line[i];
            char new_line[100];
            sprintf(new_line, "%.*s%s%s", index, line[i], value, pos
+ strlen(value1));
            strcpy(line[i], new_line);
        }
        fprintf(fptr, "%s\n", line[i]);
    }
    fclose(fptr);
    printf("\n\nSuccessfully updated the file!\n");
}









// banner function
void Banner()
{

    // welcome Banner
    system("cls");
    printf("\t\tWELCOME TO IPL CRICKET TOURNAMENT SCHEDULAR");
    printf("\nEnter your choise - ");
    printf("\n1 ) Create new schedule (Deletes any previous
Records)");
    printf("\n2 ) Delete all data");
    printf("\n3 ) Display Information");
    printf("\n4 ) Edit Information");
    printf("\n5 ) Exit");

    printf("\nEnter : ");
    scanf("%d", &value);
// made by bhupendra kumawat
// section koc dg
```

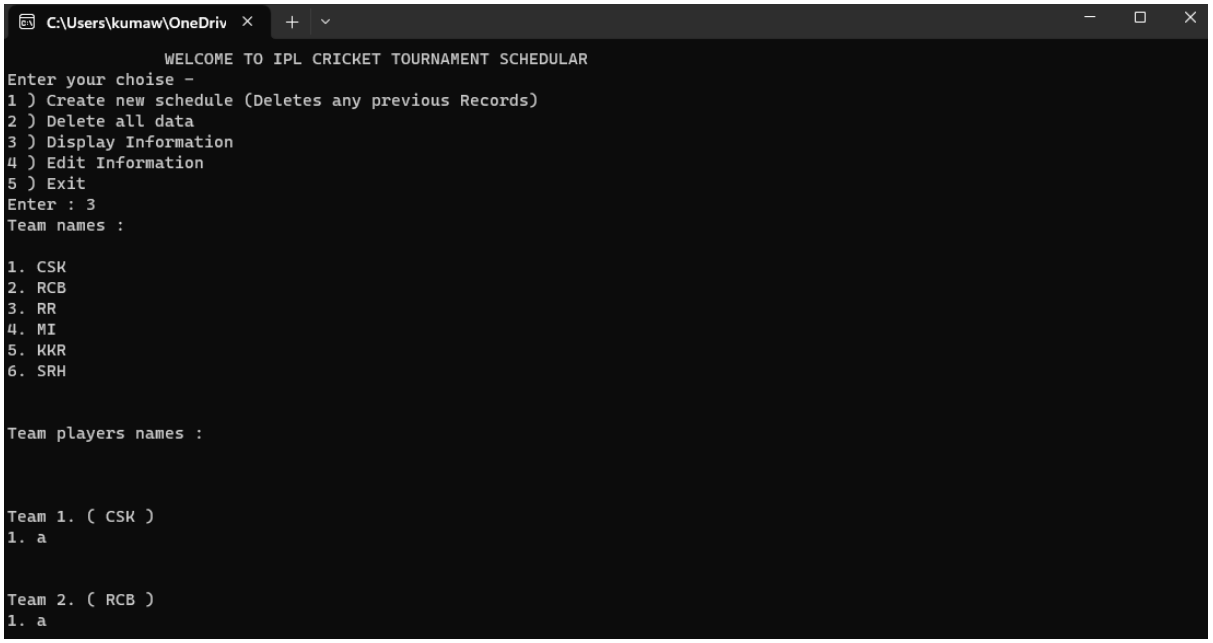
```
    switch (value)
    {
    case 1:
    {
        // system("cls");
        int no_of_teams;
        printf("Enter the number of teams: ");
        scanf("%d", &no_of_teams);
        struct team teams[no_of_teams];
        create(no_of_teams, teams);
        break;
    }
    case 2:
        delete_data();
        break;
    case 3:
        display();
        break;
    case 4:
        edit();
        break;
    case 5:
         printf(“Exiting….”);
        exit(1);
        break;
    default:{
        printf("\nInvalid input");
        break;
    }}
}
```

```c
int
main()
                          //main function
{
    while (1)
    {
      Banner();
      printf("\n\nPress any key to go back to main menu");
      getch();
    }


    return 0;
}
```

OUTPUT


C:\Users\kumaw\OneDriv
WELCOME TO IPL CRICKET TOURNAMENT SCHEDULAR
Enter your choise -
1) Create new schedule (Deletes any previous Records)
2) Delete all data
3) Display Information
4) Edit Information
5) Exit
Enter :


C:\Users\kumaw\OneDriv
Tournament Schedule:
Match 1 - CSK vs RCB on 11/05/2023
Match 2 - CSK vs RR on 12/05/2023
Match 3 - CSK vs MI on 13/05/2023
Match 4 - CSK vs KKR on 14/05/2023
Match 5 - CSK vs SRH on 15/05/2023
Match 6 - RCB vs RR on 16/05/2023
Match 7 - RCB vs MI on 17/05/2023
Match 8 - RCB vs KKR on 18/05/2023
Match 9 - RCB vs SRH on 19/05/2023
Match 10 - RR vs MI on 20/05/2023
Match 11 - RR vs KKR on 21/05/2023
Match 12 - RR vs SRH on 22/05/2023
Match 13 - MI vs KKR on 23/05/2023
Match 14 - MI vs SRH on 24/05/2023
Match 15 - KKR vs SRH on 25/05/2023
Press any key to go back to main menu


C:\Users\kumaw\OneDriv
WELCOME TO IPL CRICKET TOURNAMENT SCHEDULAR
Enter your choise -
1) Create new schedule (Deletes any previous Records)
2) Delete all data
3) Display Information
4) Edit Information
5) Exit
Enter : 3
Team names :
1. CSK
2. RCB
3. RR
4. MI
5. KKR
6. SRH
Team players names :
Team 1. (CSK)
1. a
Team 2. (RCB)
1. a