

**TRIBHUVAN UNIVERSITY**

**INSTITUTE OF ENGINEERING**

**THAPATHALI CAMPUS**

**A Final Project Report**

**On**

**Police Record Management System (PRMS)**

**Submitted By:**

Aayush Pathak (THA077BEI002)

Bikash Pandey (THA077BEI011)

Bishal Giri (THA077BEI014)

Safal Karki (THA077BEI036)

**Submitted To:**

Department of Electronics and Computer Engineering

Thapathali Campus

Kathmandu, Nepal

**Under the Supervision of**

Er. Saroj Shakya

October, 2021

# DECLARATION

We hereby declare that the project work report entitled “Police Record Management System(PRMS)” submitted for the fulfilment of the requirements for the course of CT 401 Computer Programming is our original work and the Project Work Report has not formed the basis for the award of any degree, diploma, or other similar titles.

Signature

Name of Student: Safal Karki

Signature

Name of Student: Bishal Giri

Signature

Name of Student: Bikash Pandey

Signature

Name of Student: Aayush Pathak

# ABSTRACT

Proper data management of various nature of convicts and crimes is a crucial job for law enforcement agencies all over the world. Lack of data management will create unnecessary hurdles in investigations and legal proceedings. This project intends to implement a convenient system of keeping track of legal records used by law enforcement agencies in a digital way with the help of C programming language. This project also aims to explore the various features of C and its application in real life. Furthermore, this project is an opportunity for us to enhance our skills in C programming.

*Keywords: C, data management, convict, law enforcement*

# INTRODUCTION

## **1.1 Background**

Law enforcement agencies have existed ever since the first human civilization was established. A record system is commonly implemented by law enforcement agencies for various purposes from investigating crimes to legal proceedings. Such record consists of all the information needed to identify and categorize crime reports in a systematic and convenient way. With the rapid innovation of technologies in the world and digitalized approaches being more and more common in almost every sector, the need for implementing digitalized data management system in law enforcing agencies was realized.

## **1.2 Problem statement**

Lack of records can be a primary problem in solving cases by police. There are numerous examples in which a police case was unnecessarily tedious to solve just because of lack of records. Thus to keep the record of criminal this project will be helpful.

## **1.3 Objectives**

This project aims to fulfil the following objective:

* To develop an application that keeps record of numerous convicts along with their identifiable information and their nature of crime
* To help the policeman to keep their valuable information and the data for the longer period of time
* To ensure the accuracy of the data
* To reduce the redundancy
* Facilitate interaction and sharing of information among
* To facilitate the interaction and sharing the information among the police departments, districts, state/headquarters and other police agencies

## **1.4 Application**

The project we purpose has following applications:

* Storing convict’s info in a well-organized manner
* Keeps criminal info for long time
* Reduce manual and redundant record keeping

## **1.5 Project features**

This project will have following features:

1. Adding record
2. Searching record
3. Displaying record
4. Deleting record

## **1.6 Feasibility Analysis**

### **1.6.1 Economic Feasibility**

As it is a concept of software and the hardware required is the PC only which is present anywhere nowadays. So, this project is economically feasible as almost every resource used while creating this project will be free of cost.



### **1.6.2 Technical Feasibility**

This application is developed with the help of standard C libraries. As c is a user friendly language so a program written in c will also be user friendly and it can be easily installed  
in any pc.



### **1.6.3 Operational Feasibility**

This application requires a compatible compiler for C(msvc, gcc, clang, etc.)

## **System requirements**

* Processor: i3 and above
* RAM: Minimum 1 GB
* OS: Windows

# 2 LITERATURE REVIEW

As we studied about the present running situations and the problems running on, we came to find that the issue of managing a police record is the burning issue and needs to be implemented and solved as soon as possible. So, we began to research, and we went through various mediums to search how we can solve them and now after consulting various books and various online pages we came to have an idea that using C programming we could solve this problem. Good knowledge of following parts related to C programming is necessary for the making of the project.

## **2.1 File handling**

In programming, we may require some specific input data to be generated several numbers of times. Sometimes, it is not enough to only display the data on the console. The data to be displayed may be very large, and only a limited amount of data can be displayed on the console, and since the memory is volatile, it is impossible to recover the programmatically generated data again and again. However, if we need to do so, we may store it onto the local file system which is non-volatile and can be accessed every time. Here, comes the need of file handling in C.

File handling in C enables us to create, update, read, and delete the files stored on the local file system through our C program. The following operations can be performed on a file.

* Creation of the new file
* Opening an existing file
* Reading from the file
* Writing to the file
* Deleting the file

## **2.2 Array and Pointer**

Arrays a kind of data structure that can store a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Pointers in C are easy and fun to learn. Some C programming tasks are performed more easily with pointers, and other tasks, such as dynamic memory allocation, cannot be performed without using pointers. So, it becomes necessary to learn pointers to become a perfect C programmer. Let's start learning them in simple and easy steps.

## **2.3 Functions**

A function is a group of statements that together perform a task. Every C program has at least one function, which is **main()**, and all the most trivial programs can define additional functions.

You can divide up your code into separate functions. How you divide up your code among different functions is up to you, but logically the division is such that each function performs a specific task.

A function **declaration** tells the compiler about a function's name, return type, and parameters. A function **definition** provides the actual body of the function.

# 3 Methodology

The working of our code is displayed in following chart as far as possible:

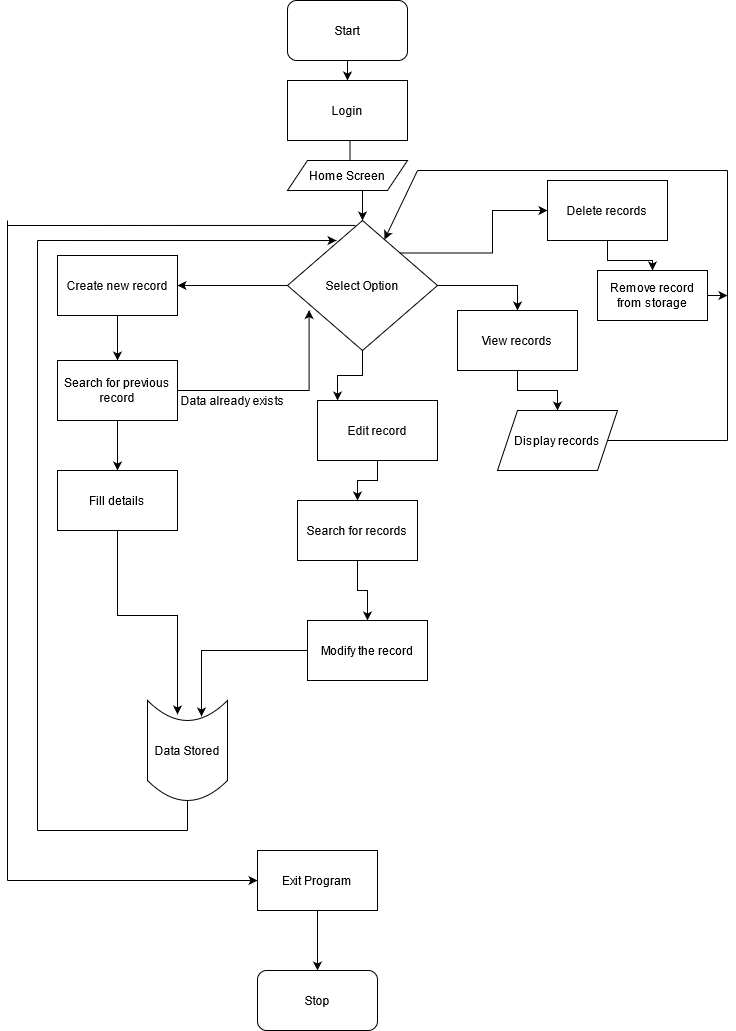


Fig: working of system

## **Login**

The login system takes username and password from the user. A certain combination of username and password is already defined in the system. It uses conditional logic. The program will only continue if the user enters the correct username and password.

## **Home Screen**

After the user has successfully logged in, the system displays a home screen from which the user can continue to choose which function to implement. There are multiple choices from which user can continue from here. On the basis of our choice we can add or display or delete or search records.

## **Add new record**

If the user chooses “Add new record” option, the system executes the function add new record. Then the user can enter multiple data. Most of them are strings and integers. But before adding the new record, the system searches if that record was already in the database.

## **3.4 Search record**

The system can search for the records which already exist in the database. The system returns the searched value from the database.

## **Edit record**

After a record is added, the user can edit the already existing record. The modified record will be saved in the database.

## **View record**

The system can display whichever record the user wants by choosing the view record option. The system basically displays the existing record to the screen the user has searched for.

## **Delete record**

If the user chooses the delete record option, the system allows the user to search for a record and delete it from the database forever.

# Implementation and Result

## **4.1 Implementation details:**

This program was developed to fulfill the needs of police to keep the criminal record in a systematic and time feasible way. To build this program, we used library functions like windows.h, stdio.h, string.h, time.h, conio.h and stdlib.h. There are total 7 user defined functions and many library functions made to make the work more understandable and easier. Here we have made a function for each individual functioning of our work such as loading screen, login, add record, search record, edit record, delete record, view record etc.

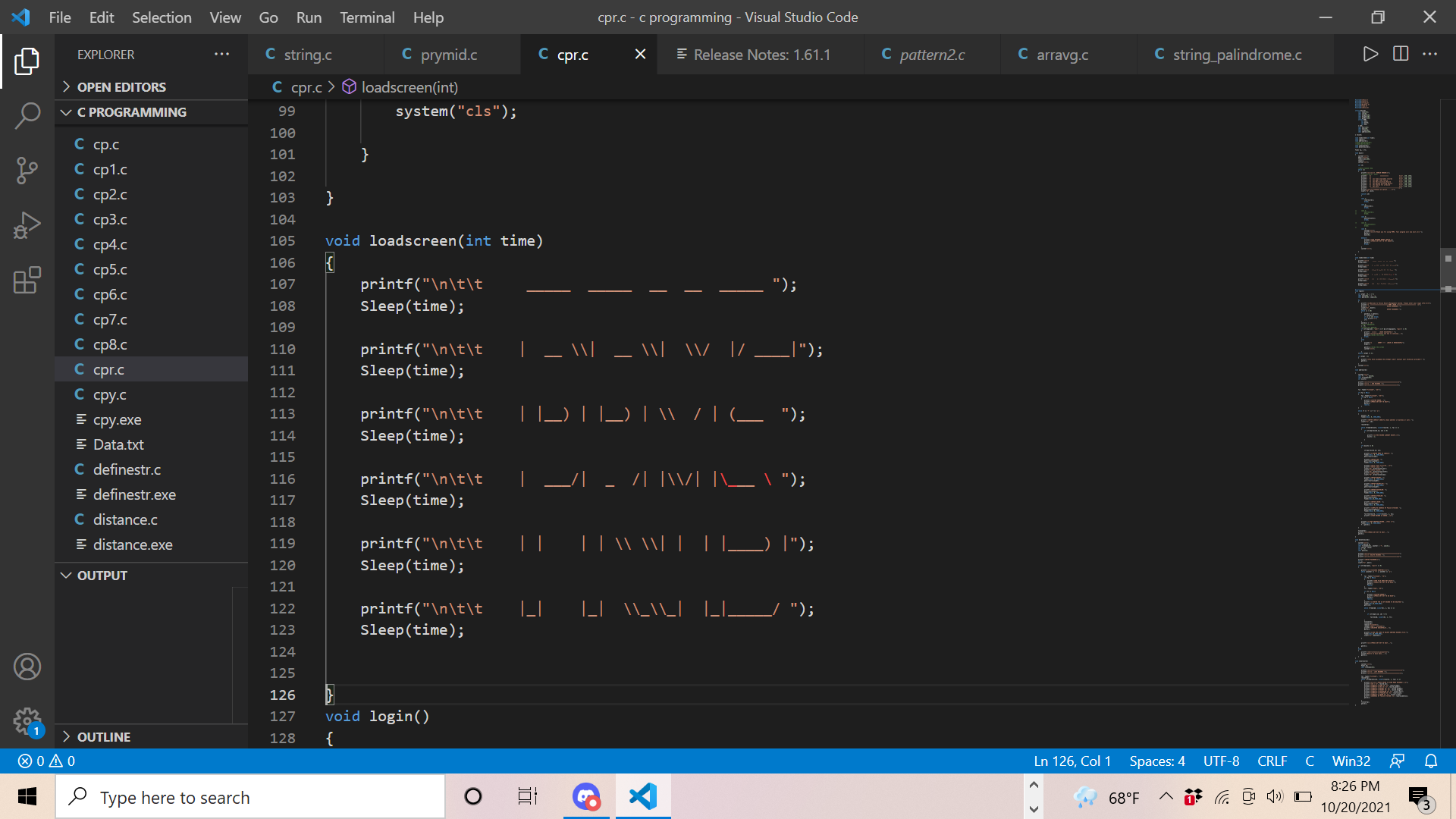


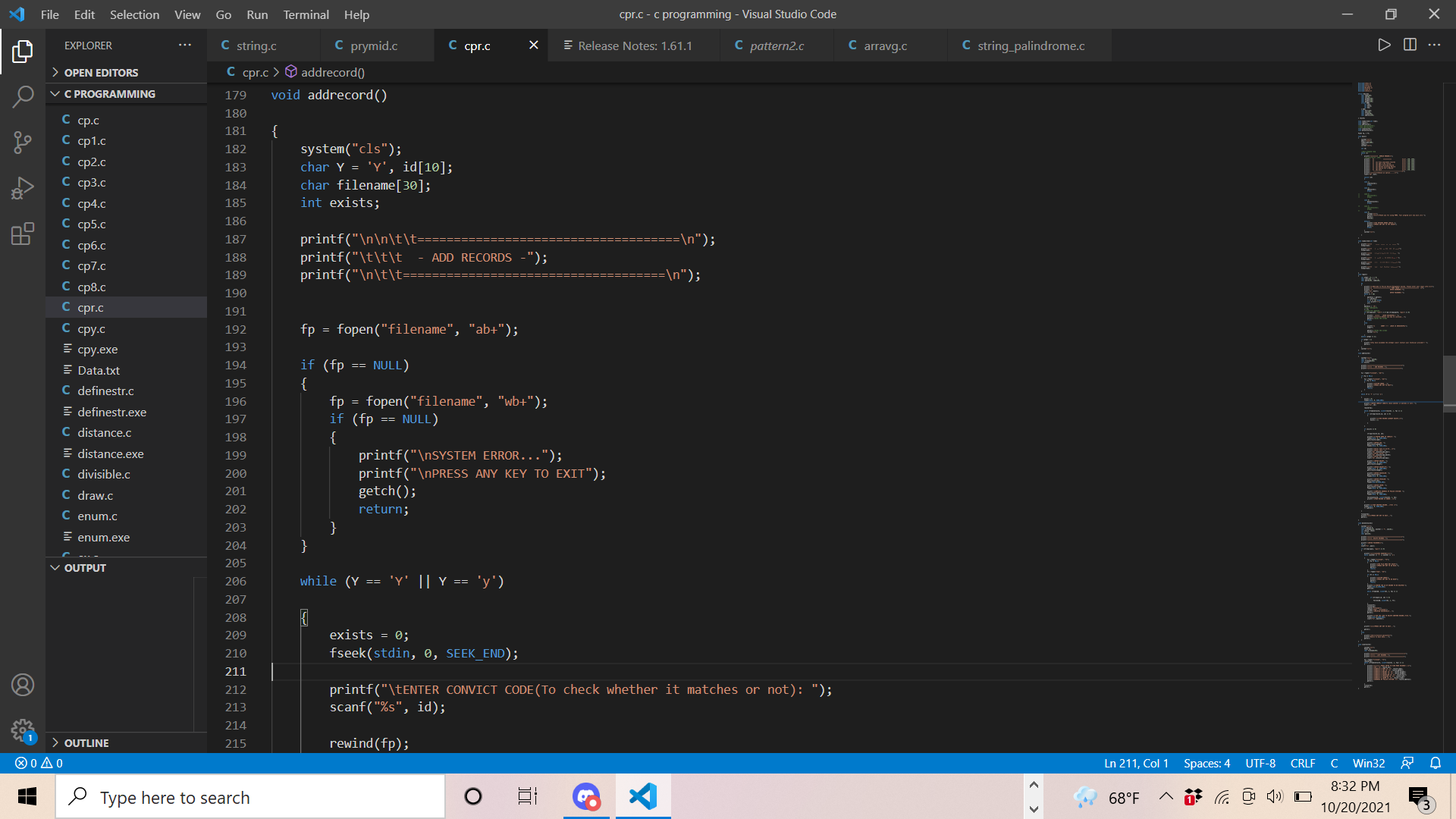
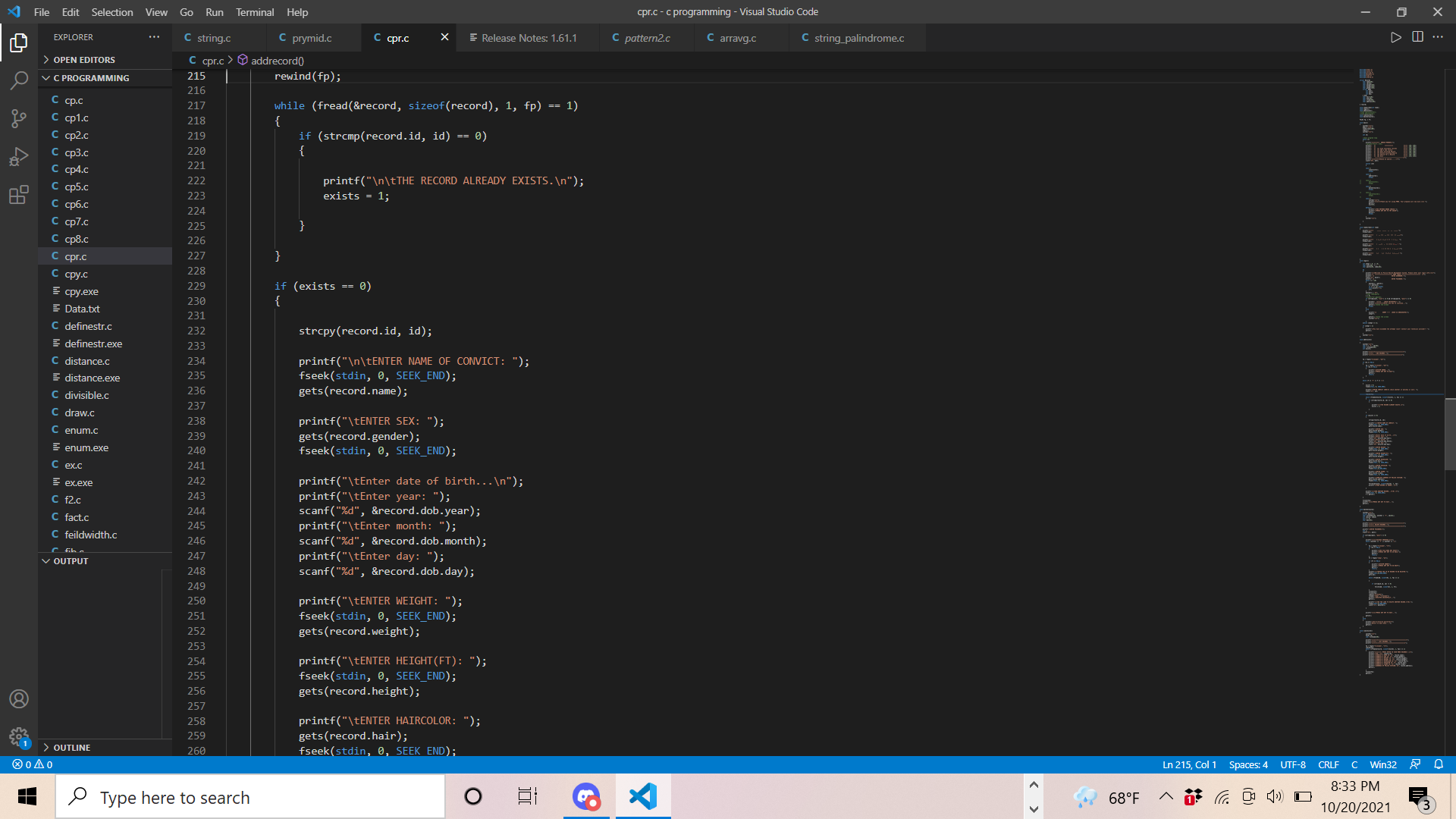
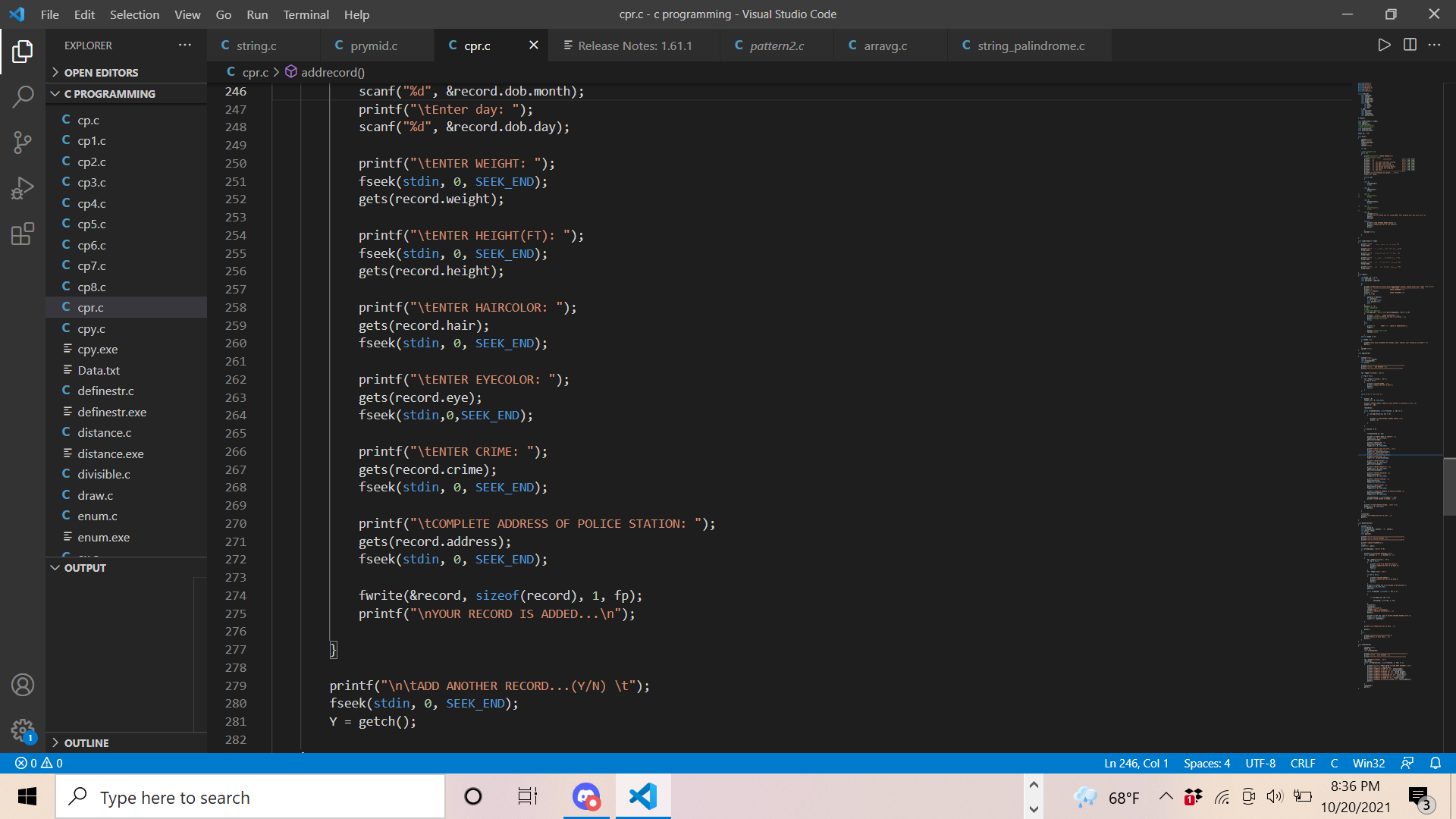
Fig 4.1.1: load screen function

The lines of code shown above when run using a compiler it results a welcome screen. The main objective of this function is to make our welcome screen attractive.

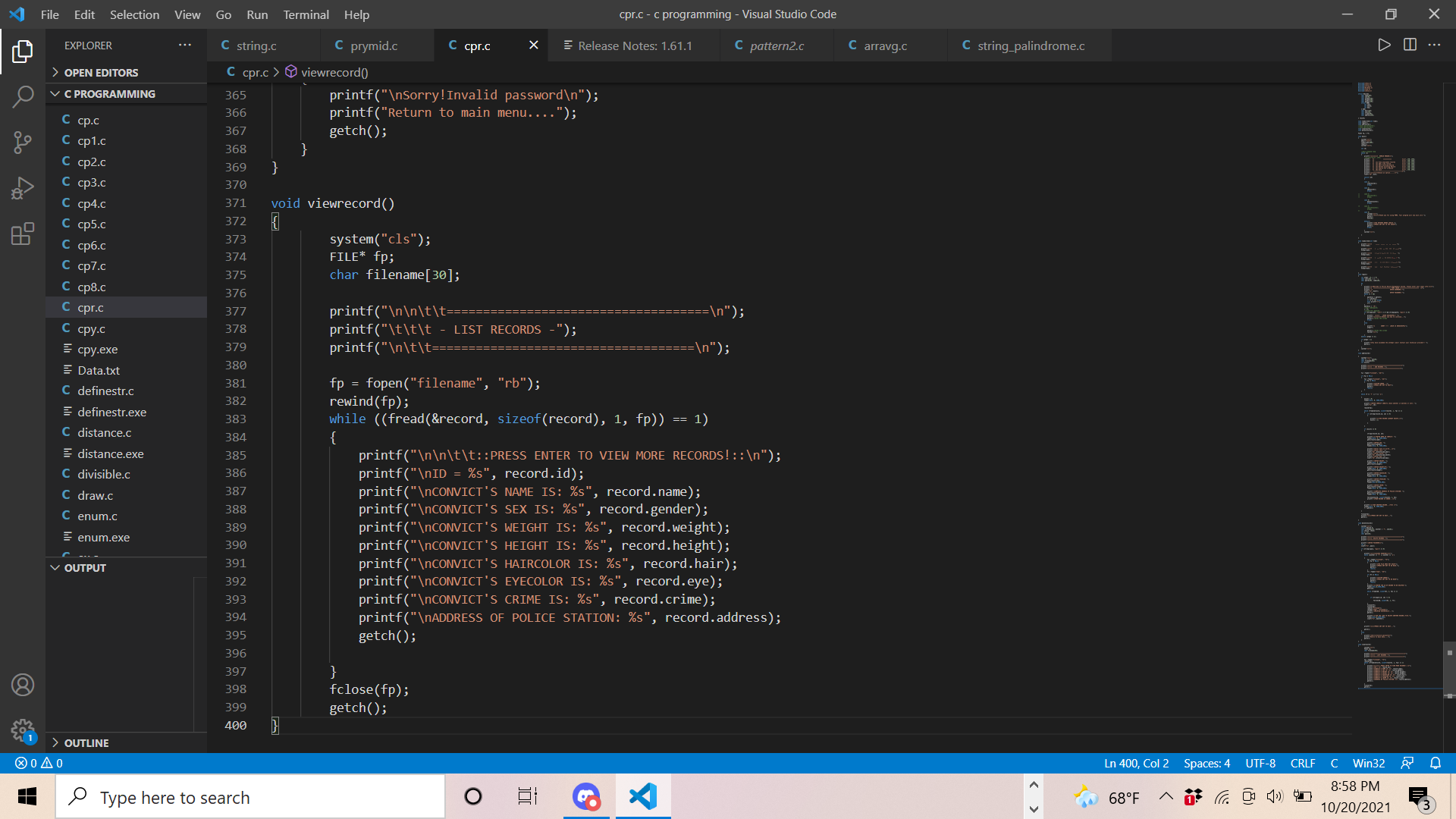


Fig 4.1.2: login function

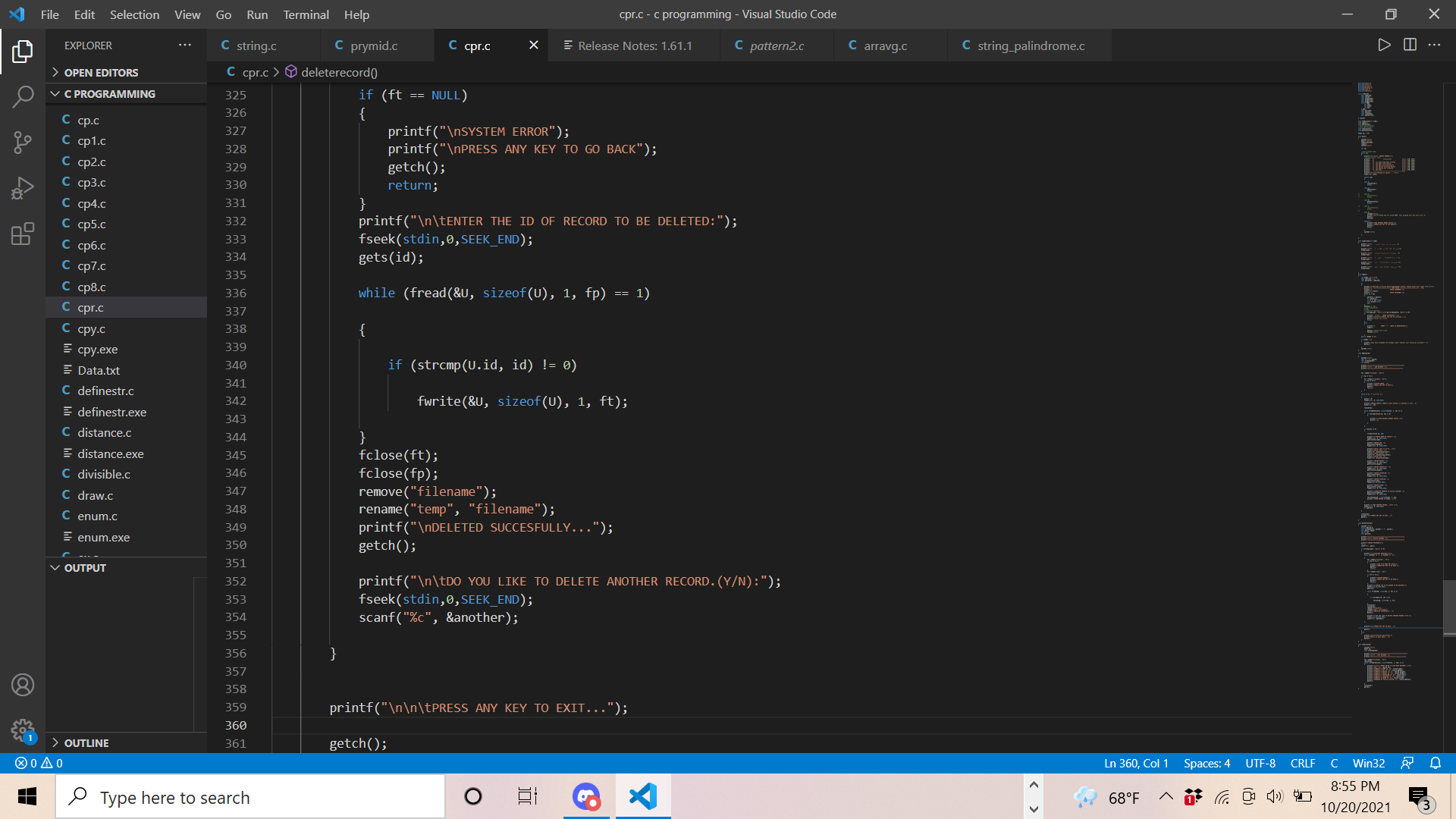
The above lines of code is used for login. On running the code in a compiler it will ask the user to enter the username and password and if the combination of username and password matches with our initial combination then only it will open the main menu. The main objective of this function is to maintain confidentiality.

    
Fig 4.1.3: add record function

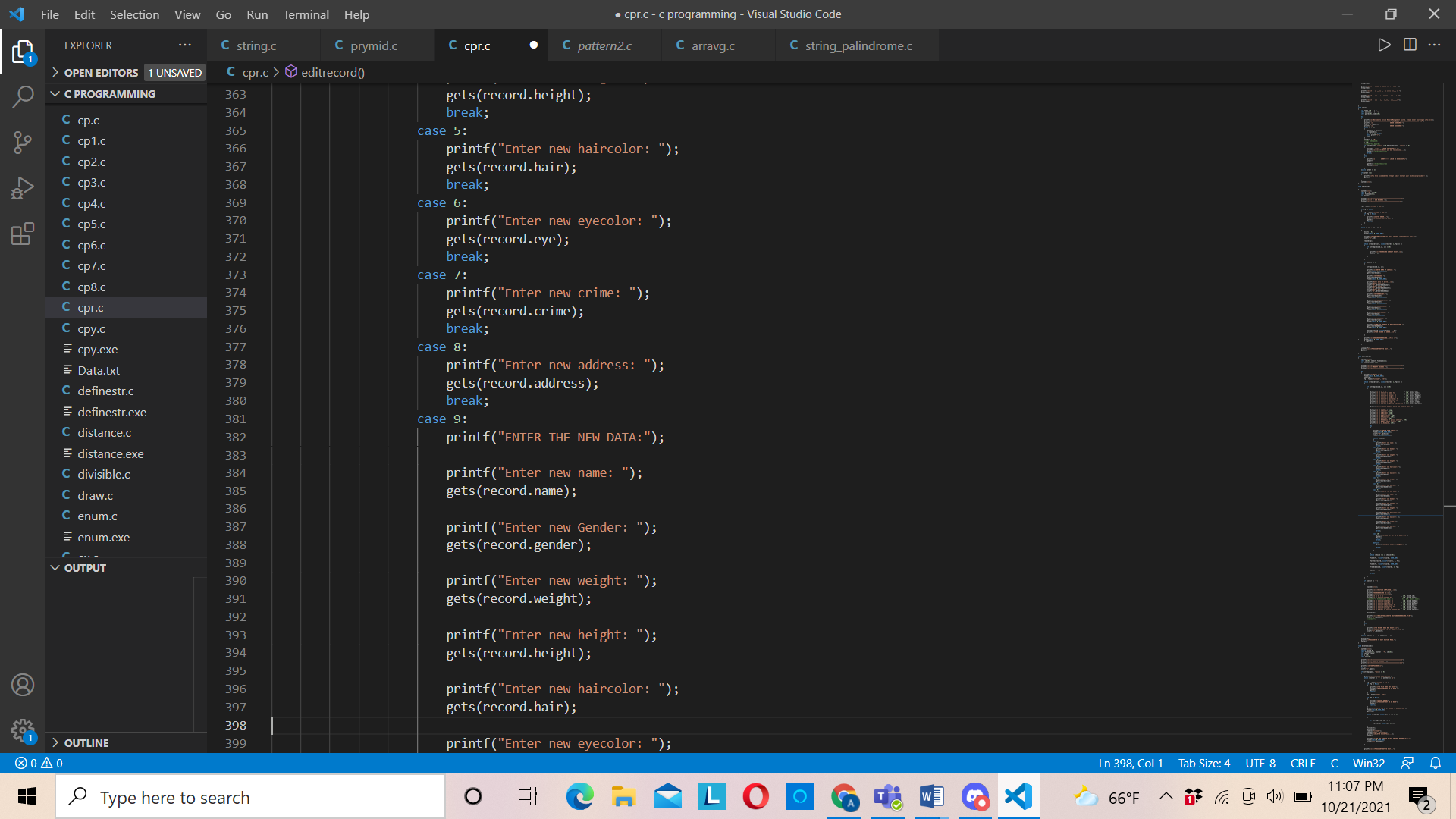
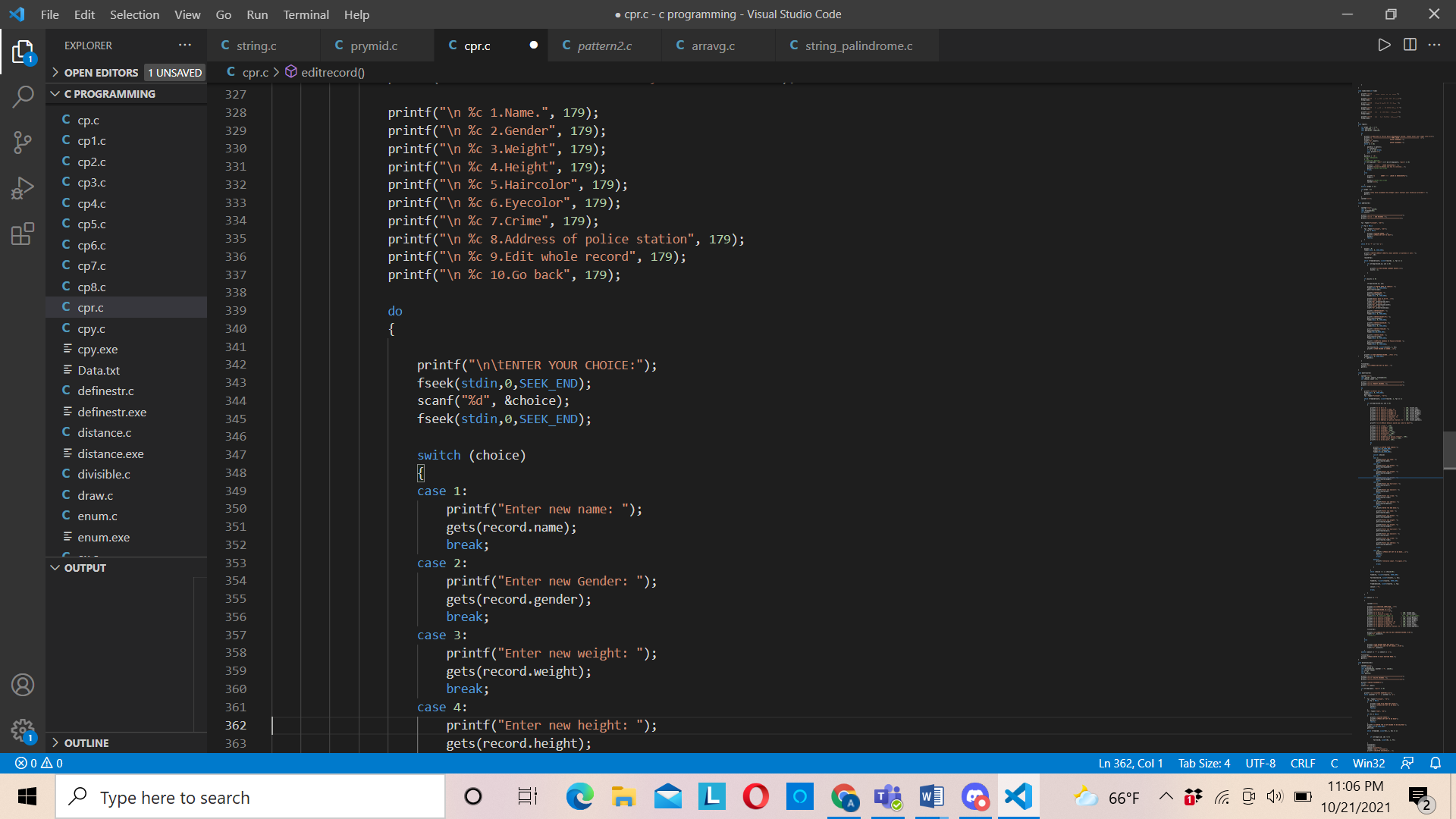
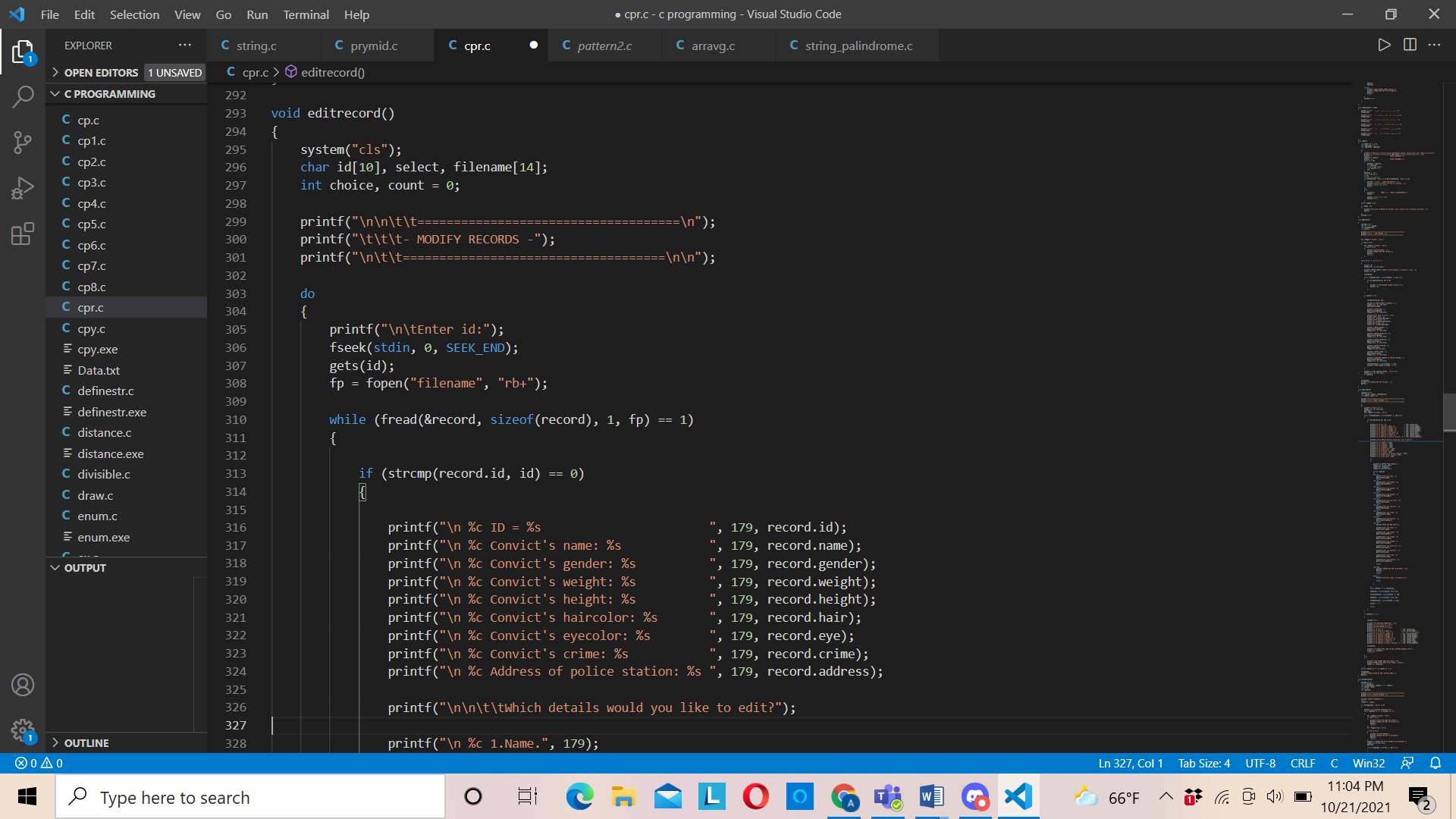
The above line of code is used to add the record in our system. At first the system checks either the record of criminal exist or not. If the record exist then it will be opened to change or add some new information and if the record dosenot exist then the new file will be opened. The main objective of this function is to add the new record which helps to make our record updated and make it reliable.

  
Fig 4.1.4: view record function

The lines of code shown above when run with the help of a compiler helps to view a record that is already entered in our file. It list all the record that is recorded in our file serially with their basic information associated with them. The main objective of this function is to view all the record that is in our file at a single time in serial manner which will make the system efficent.

  
Fig 4.1.5 Delete record function

The above line of code is used to delete the record which was recorded previously. It is very important feature that is needed in a police record management system to make it flexible because the acused may be proved innocent after sometime at that time the record becomes useless. So, delete record is needed. To make it secure the password protection is given to it. The main objective of this function is to make our program flexible.



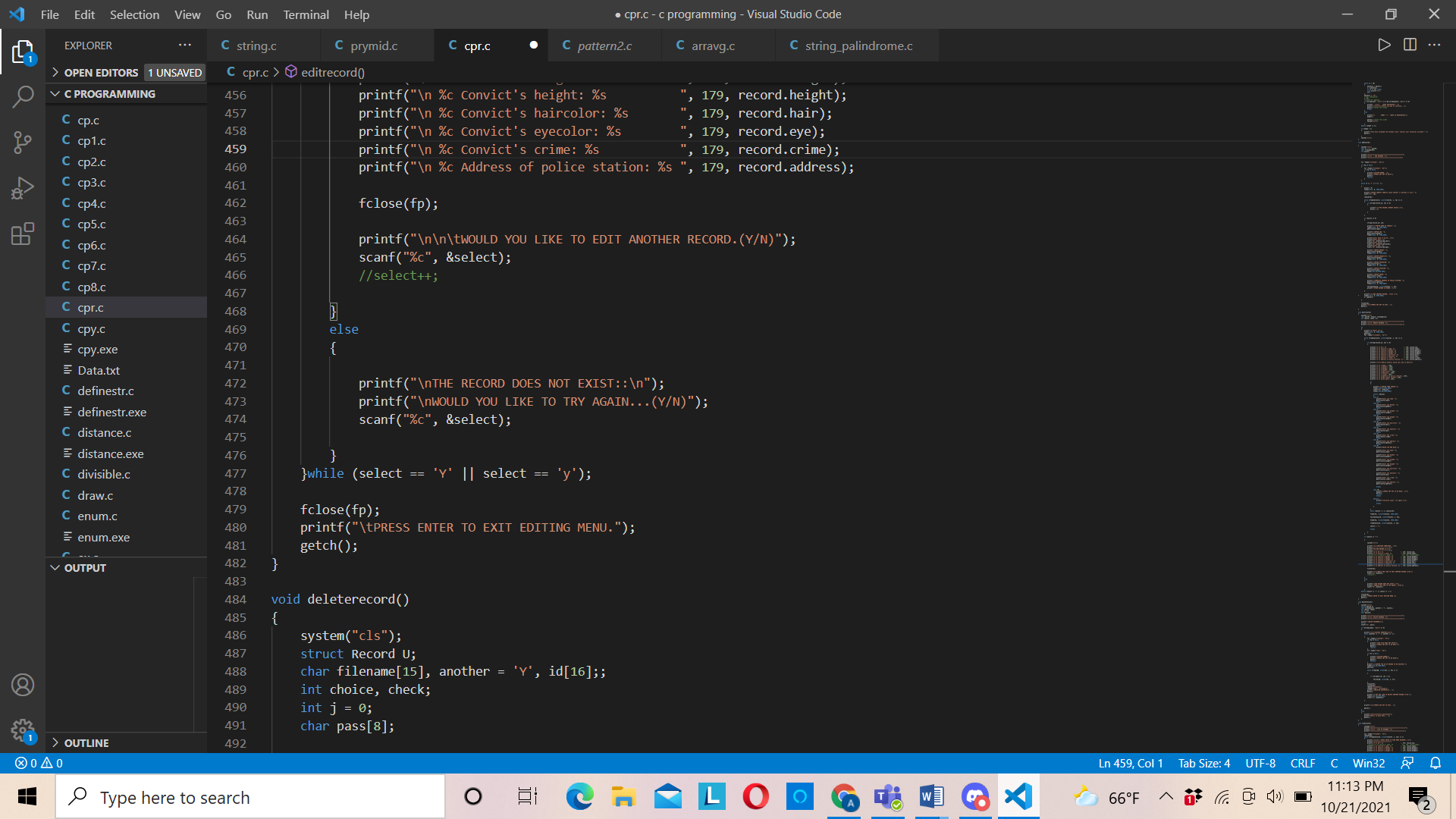
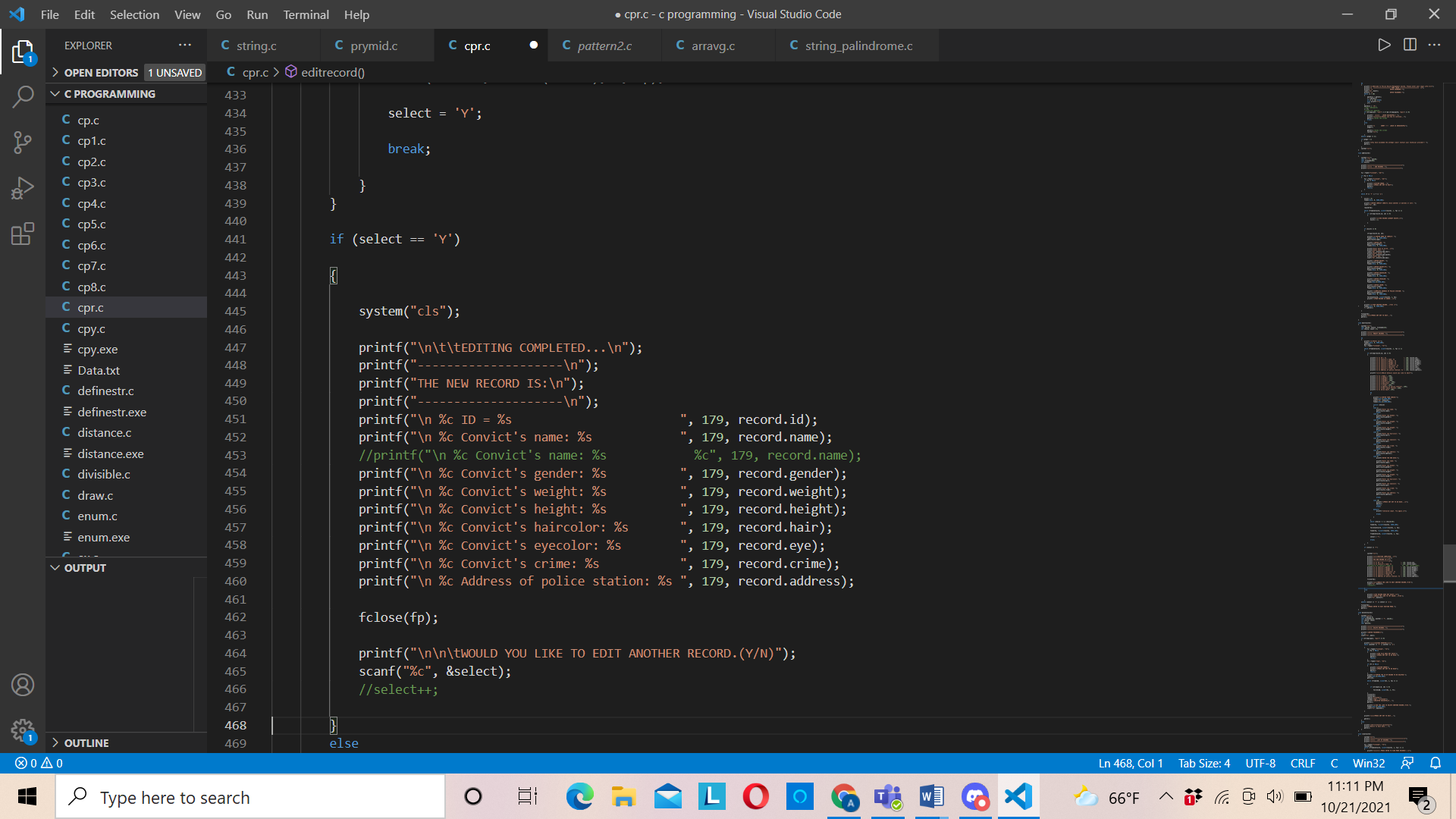
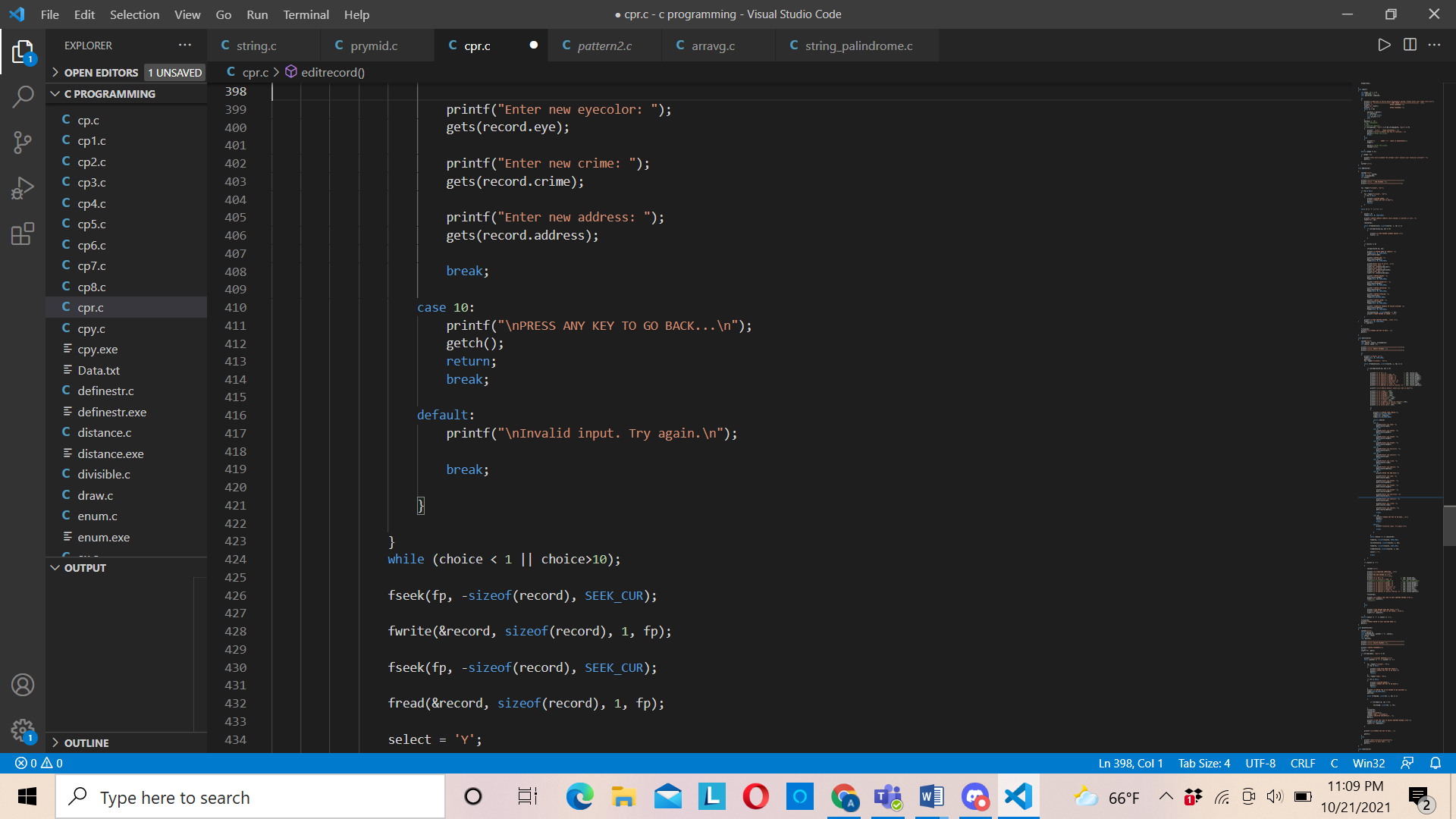
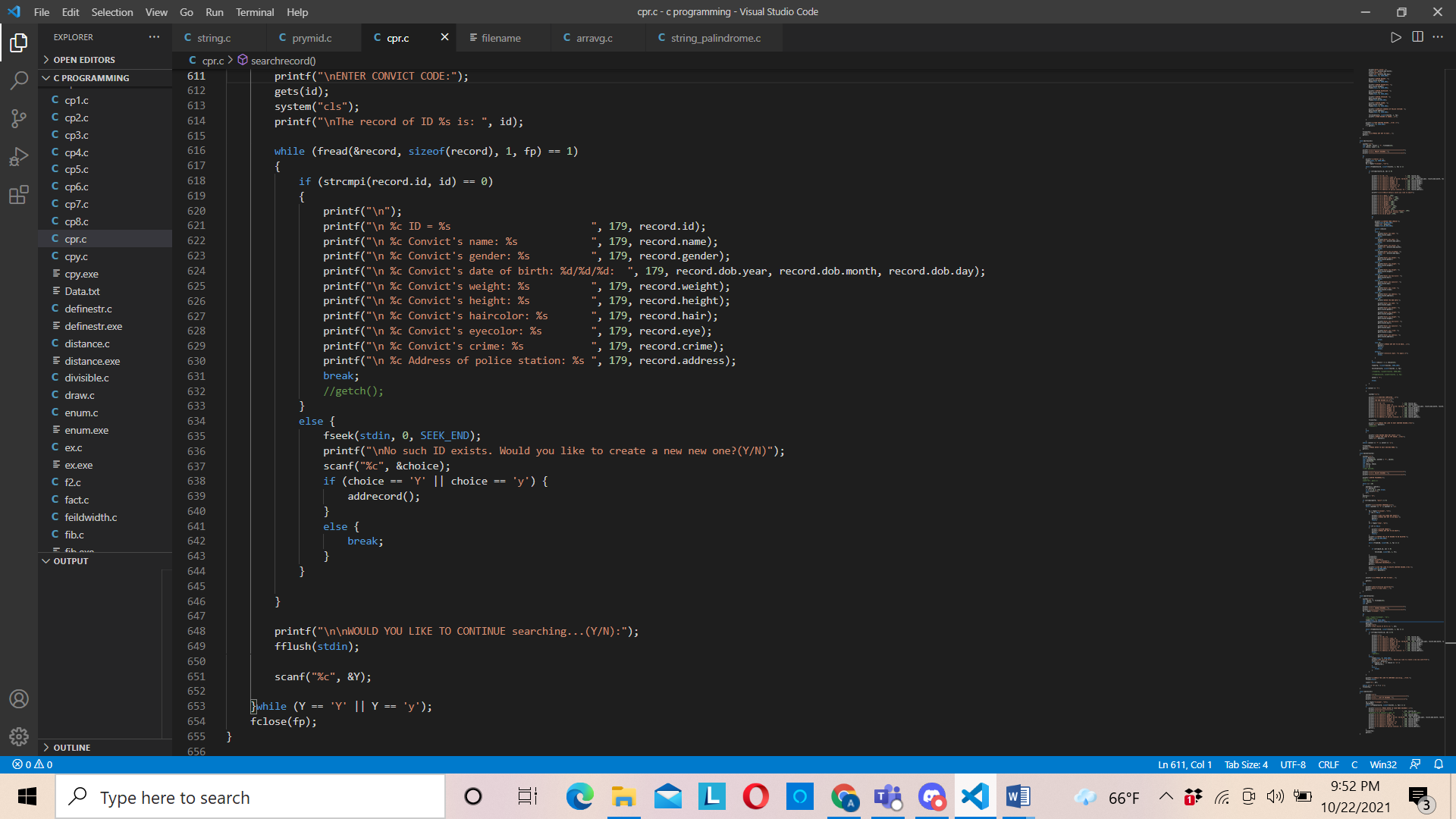
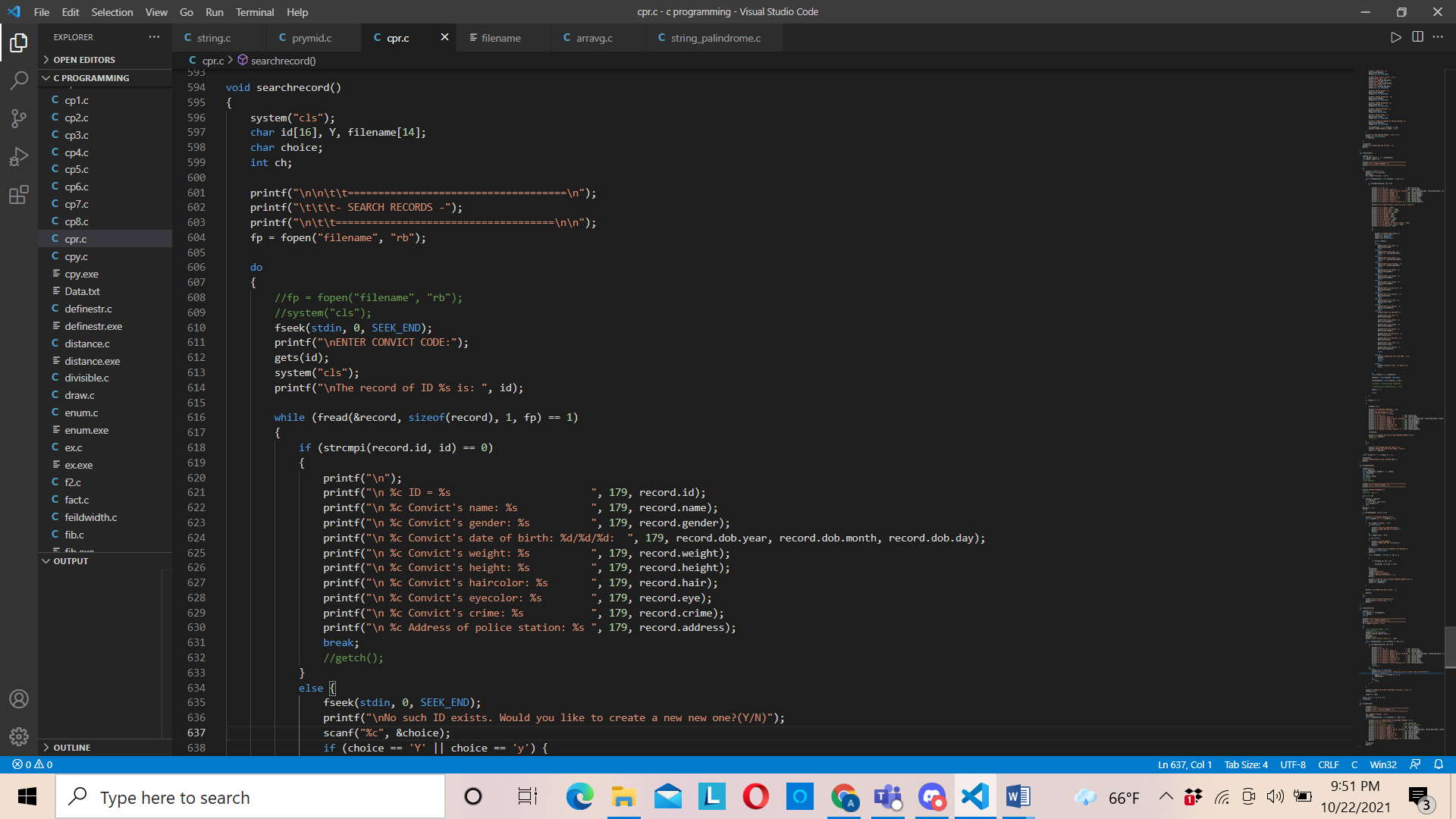


Fig 4.1.6: edit record function

The above lines of code when run with the help of compiler will help to edit the information that we have previously entered. We may edit certain part of the infromation or whole information on the basis of the option that will be given to us. The main objective of this function is to increase the accuracy of our program.

  
Fig 4.1.7: Search function

The above line of code when compiled and run it enable us to search the criminal record by is id and if the record doesnot exist then we can add a new one or we can leave the screen abd go to main menu. The main objective of this function is to make our program time efficient.

As we go through the program, we first get an interface saying welcome to the police record management login page and the program will continue only if the user enter correct username and password then only the program will run. To make the information confidential login is required at first. After we enter correct username and password, the main menu is displayed where we get many options such as delete record, edit record, add record, search record, etc.  
On the basis of the option we entered we can add, edit, delete, view, search.

## **4.2 Result**

While running the code the results that were obtained are presented below

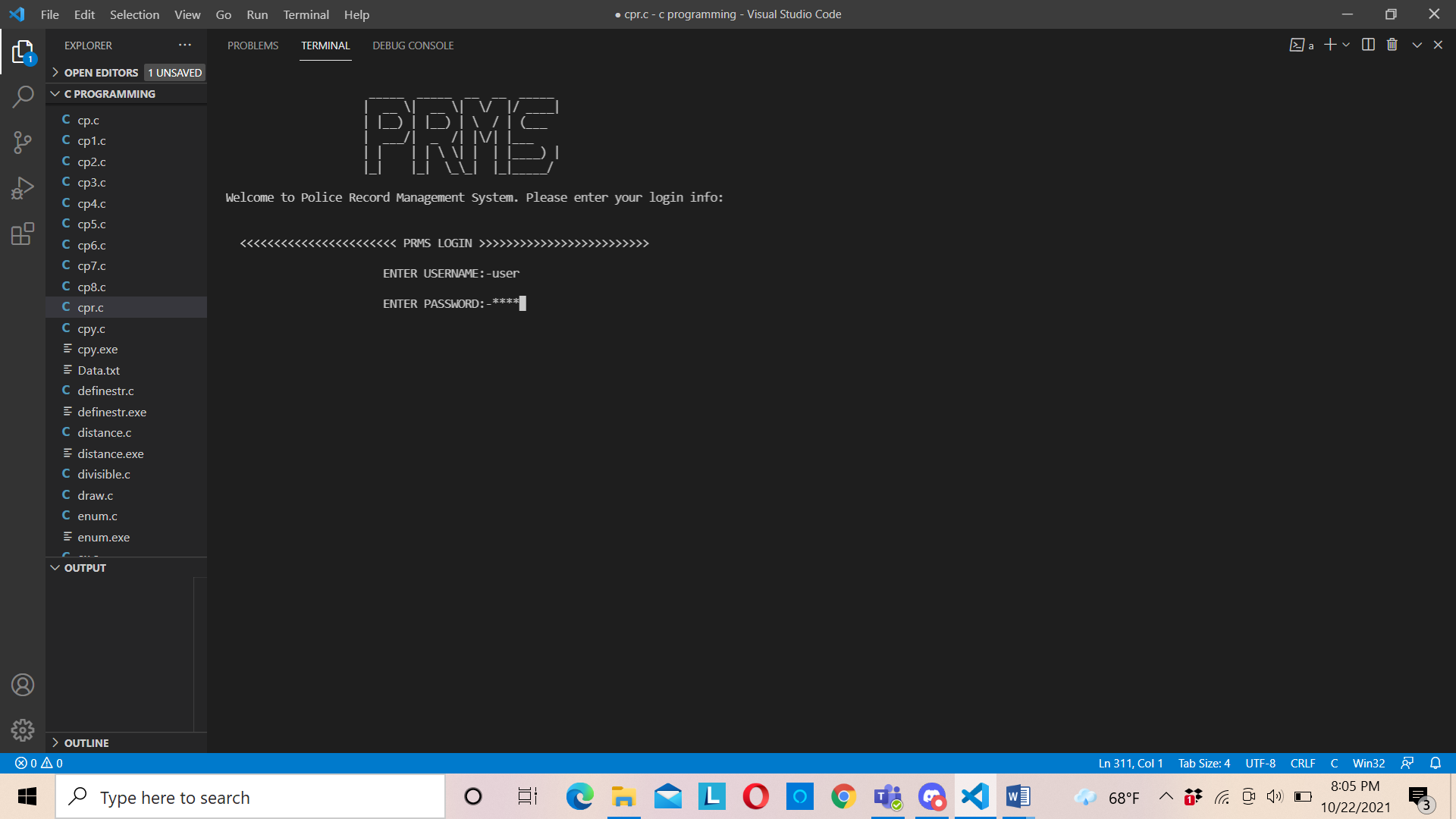
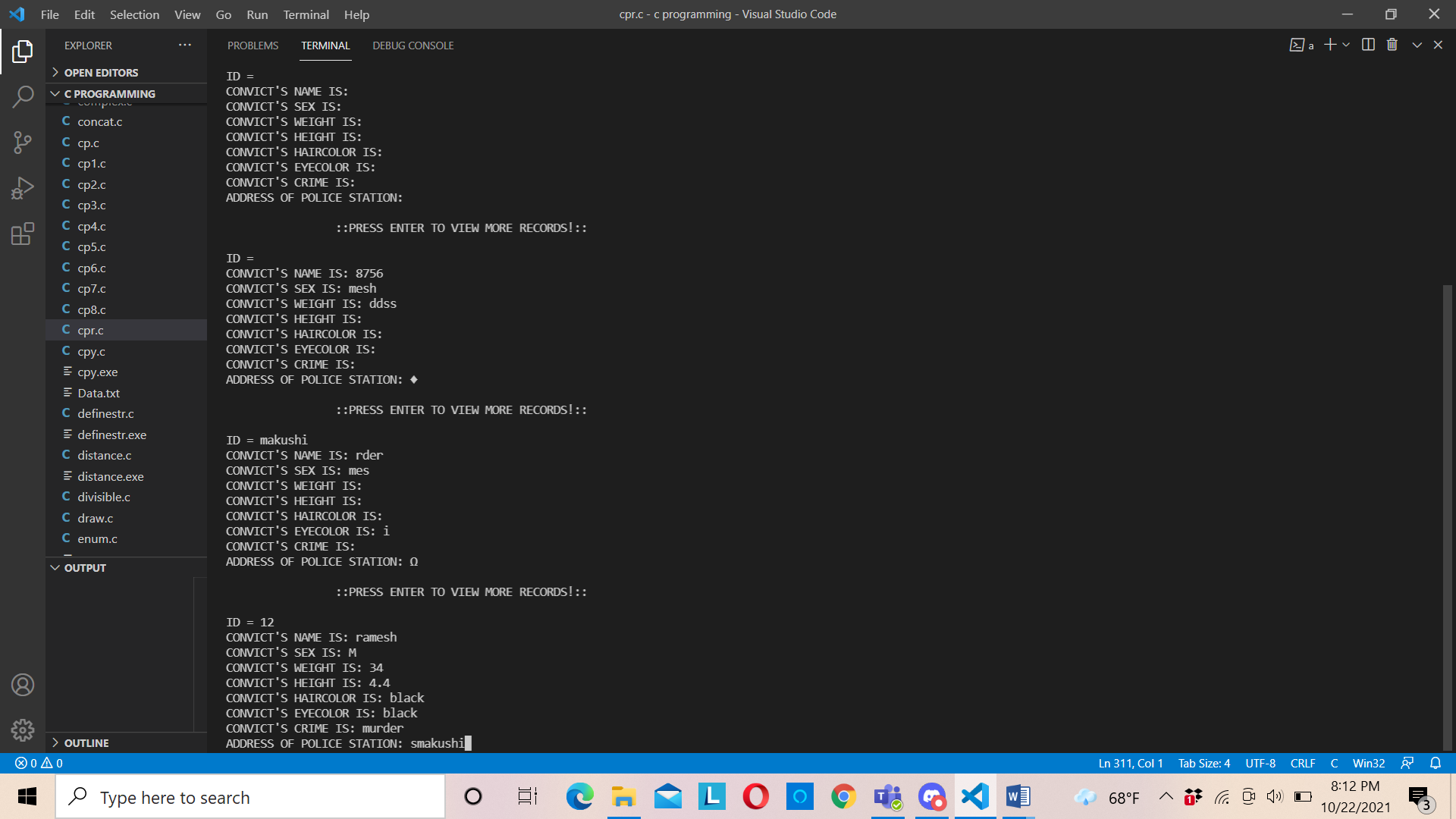
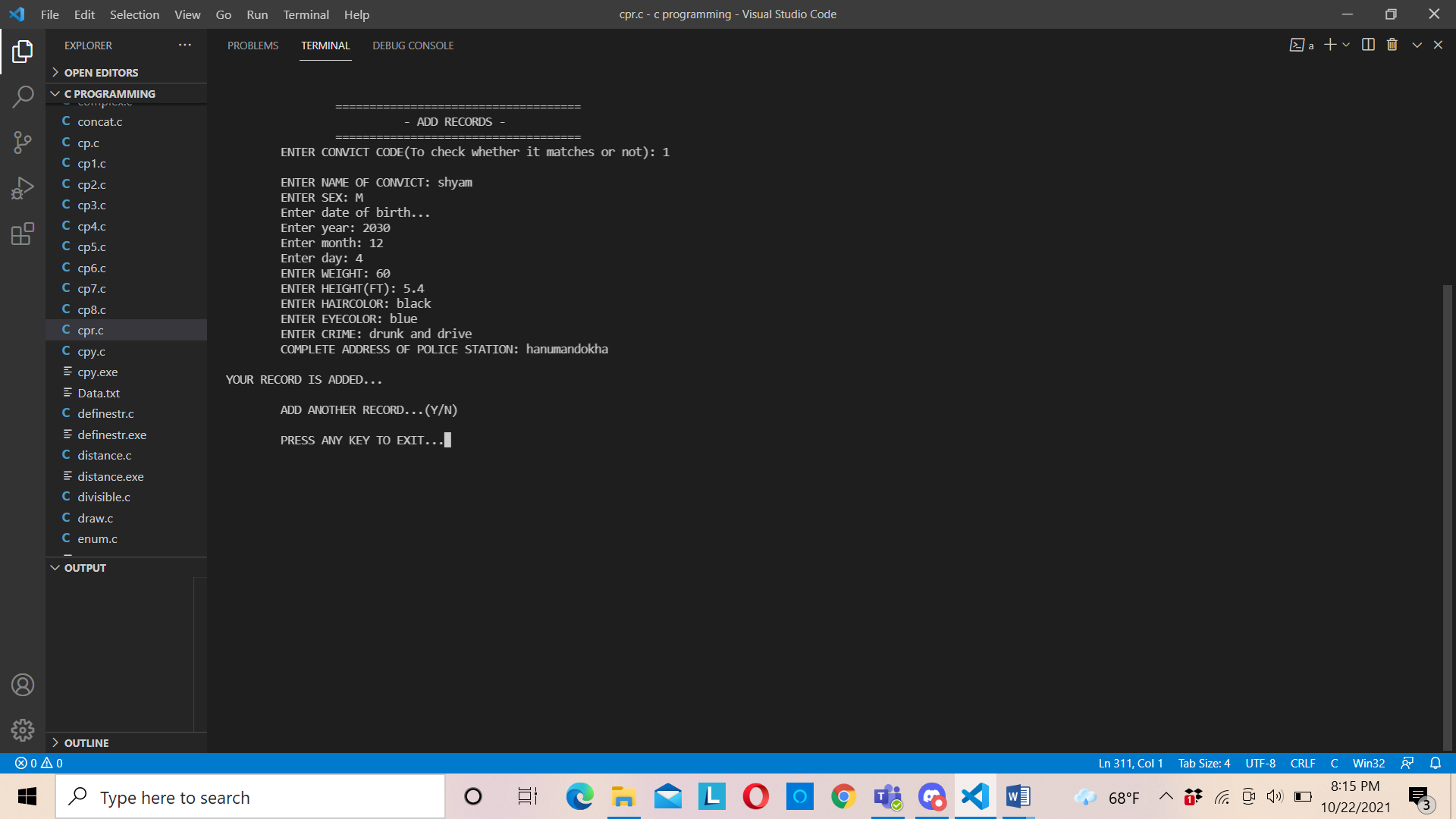
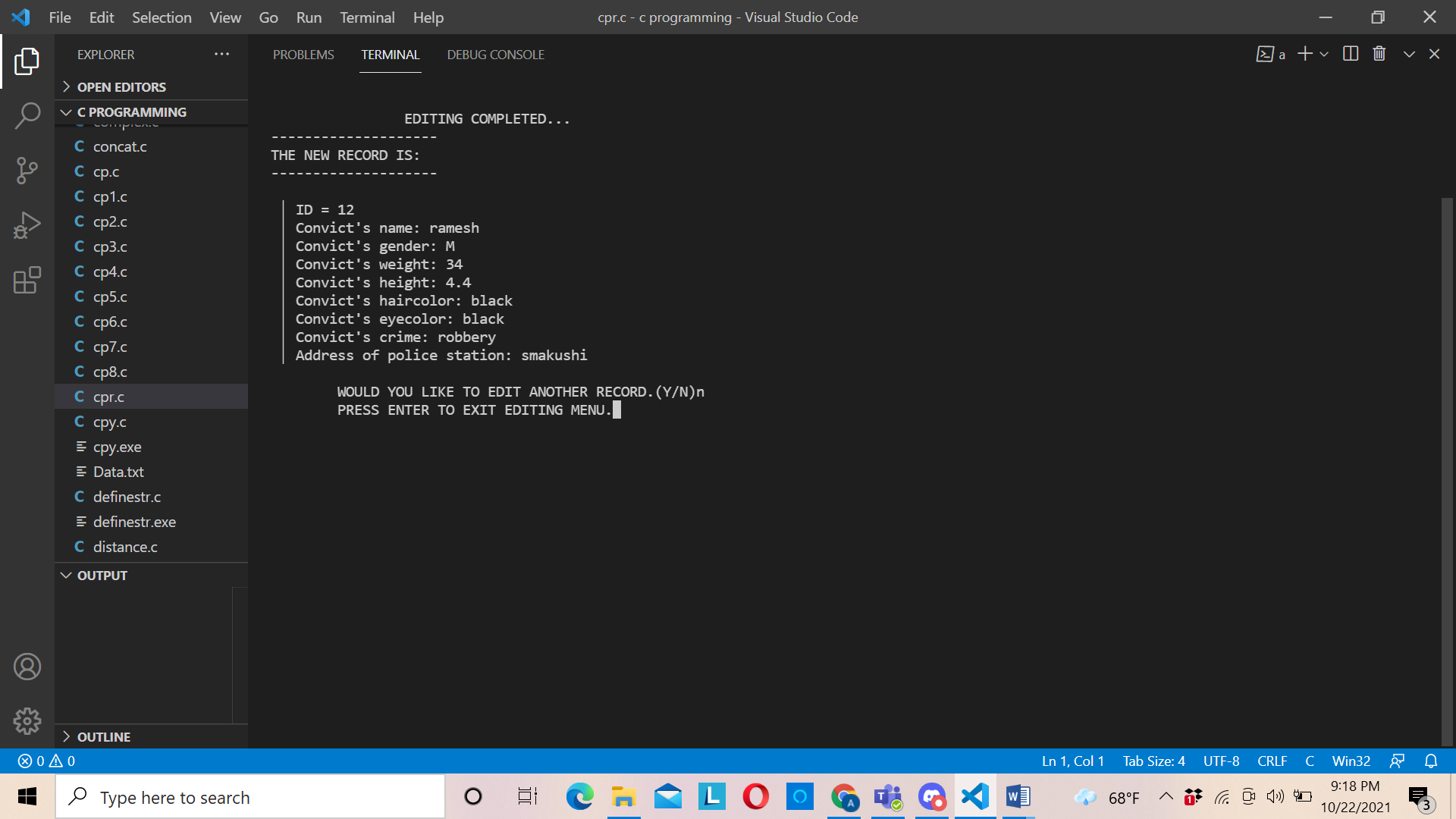
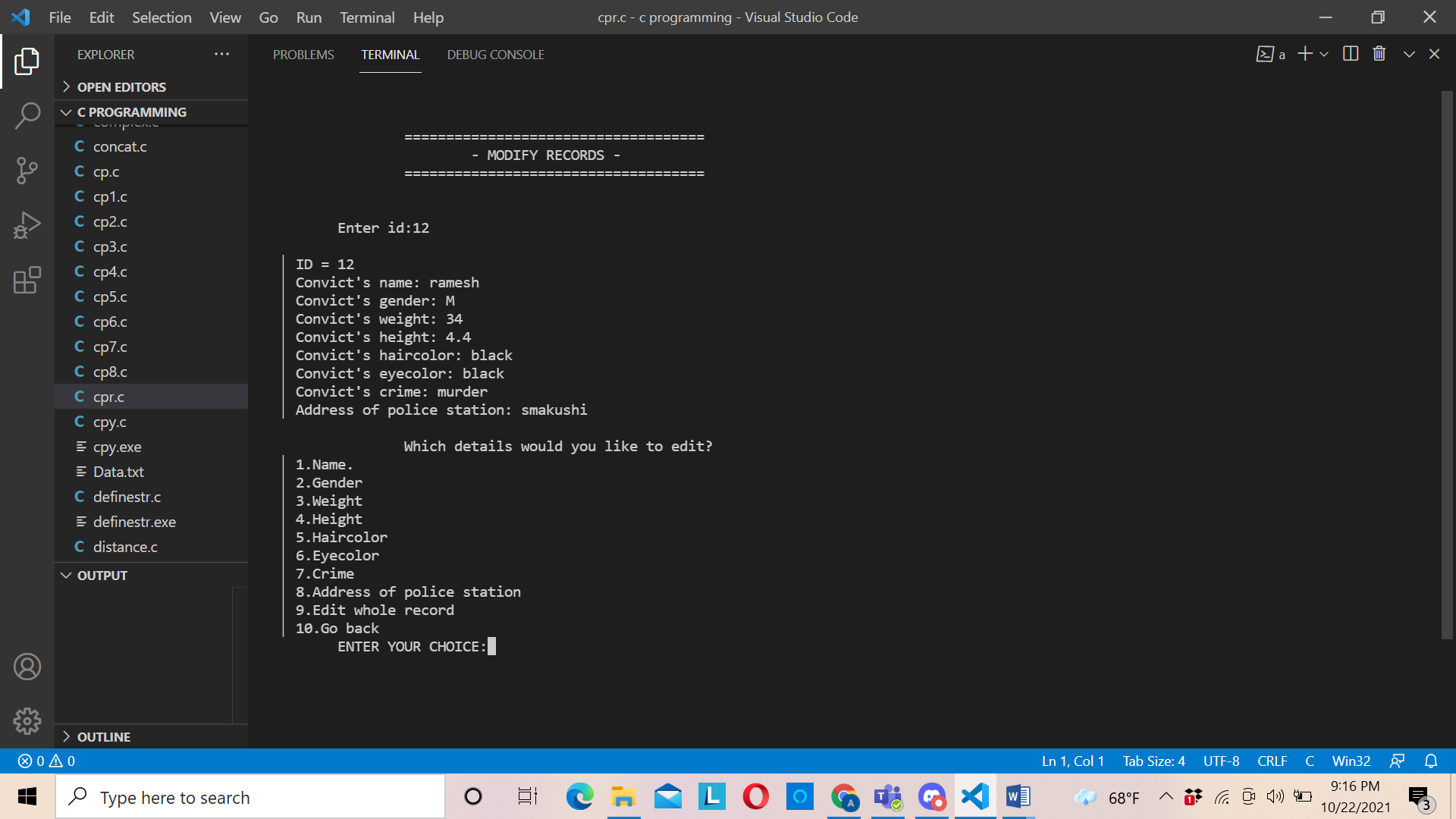
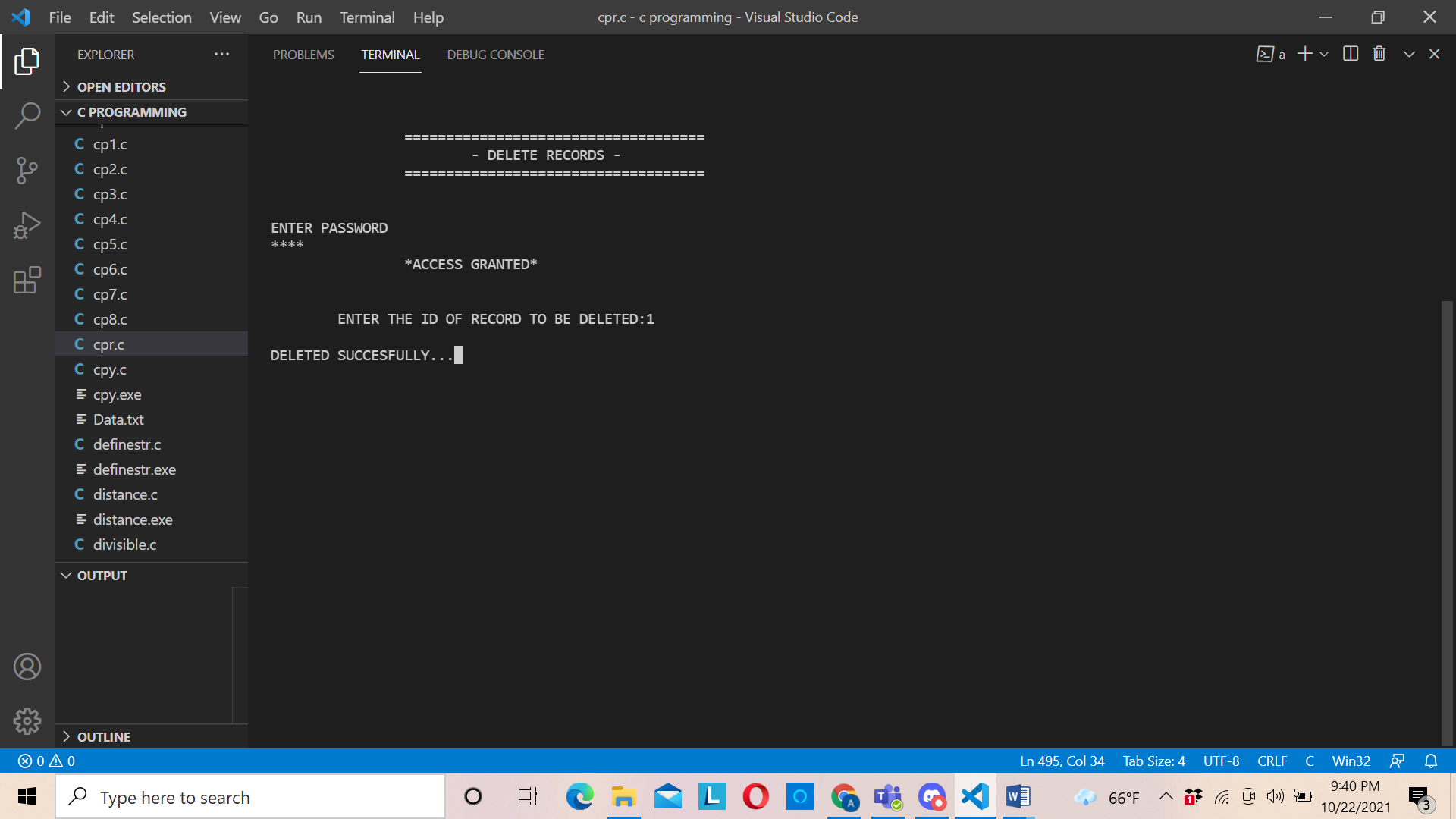
  
 fig4.2.1: welcome screen

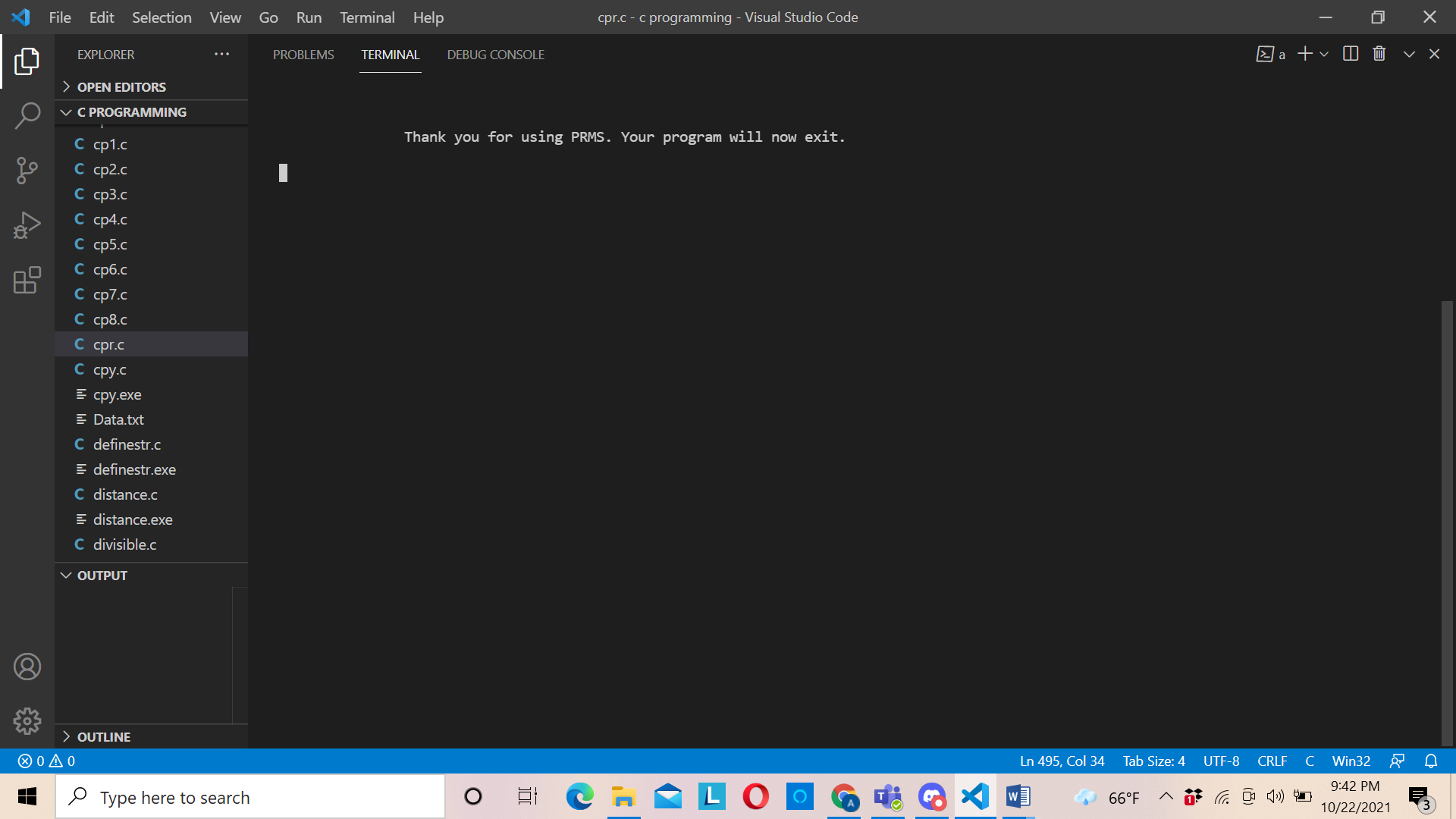
  
 fig4.2.2: main menu

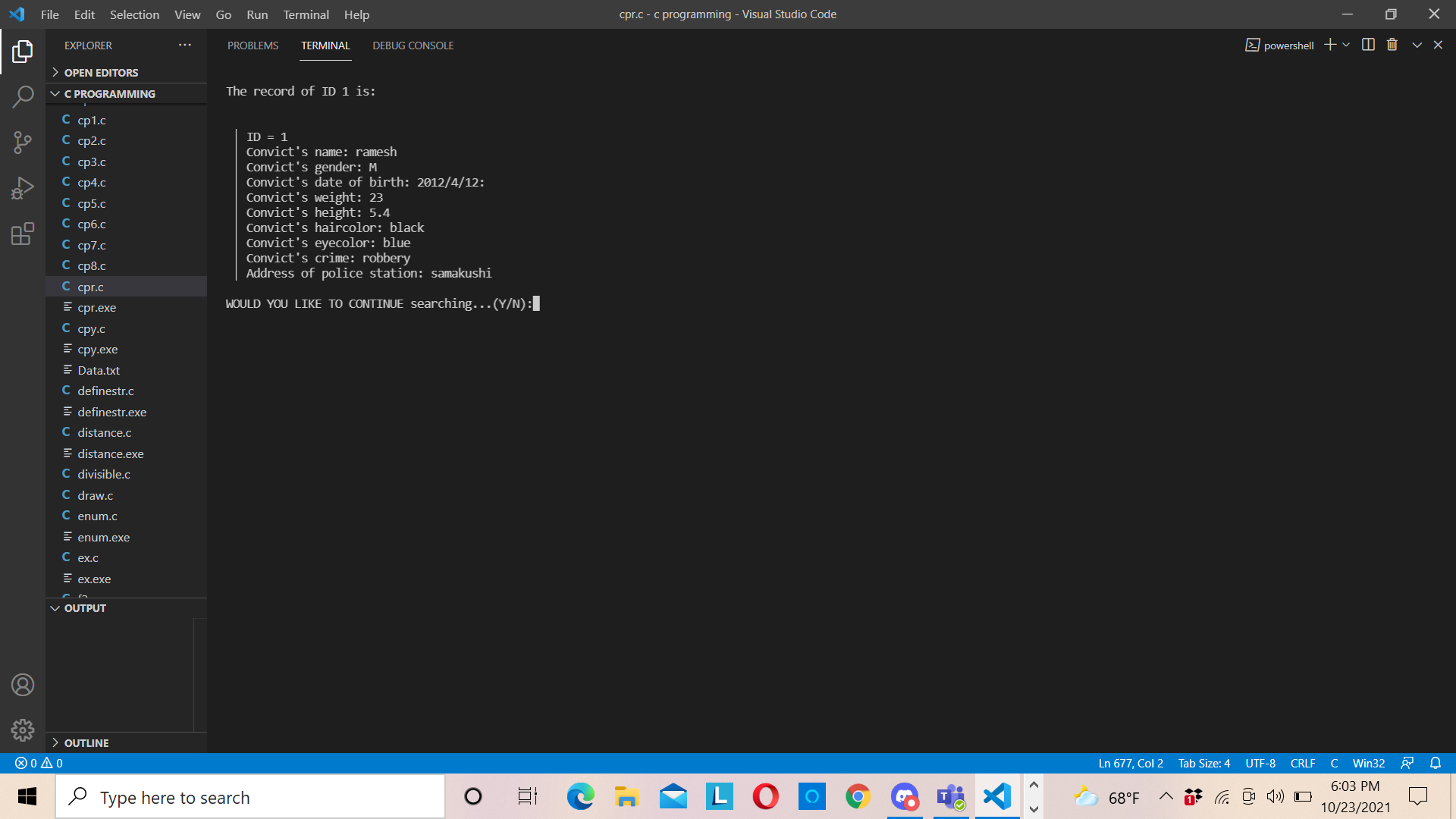
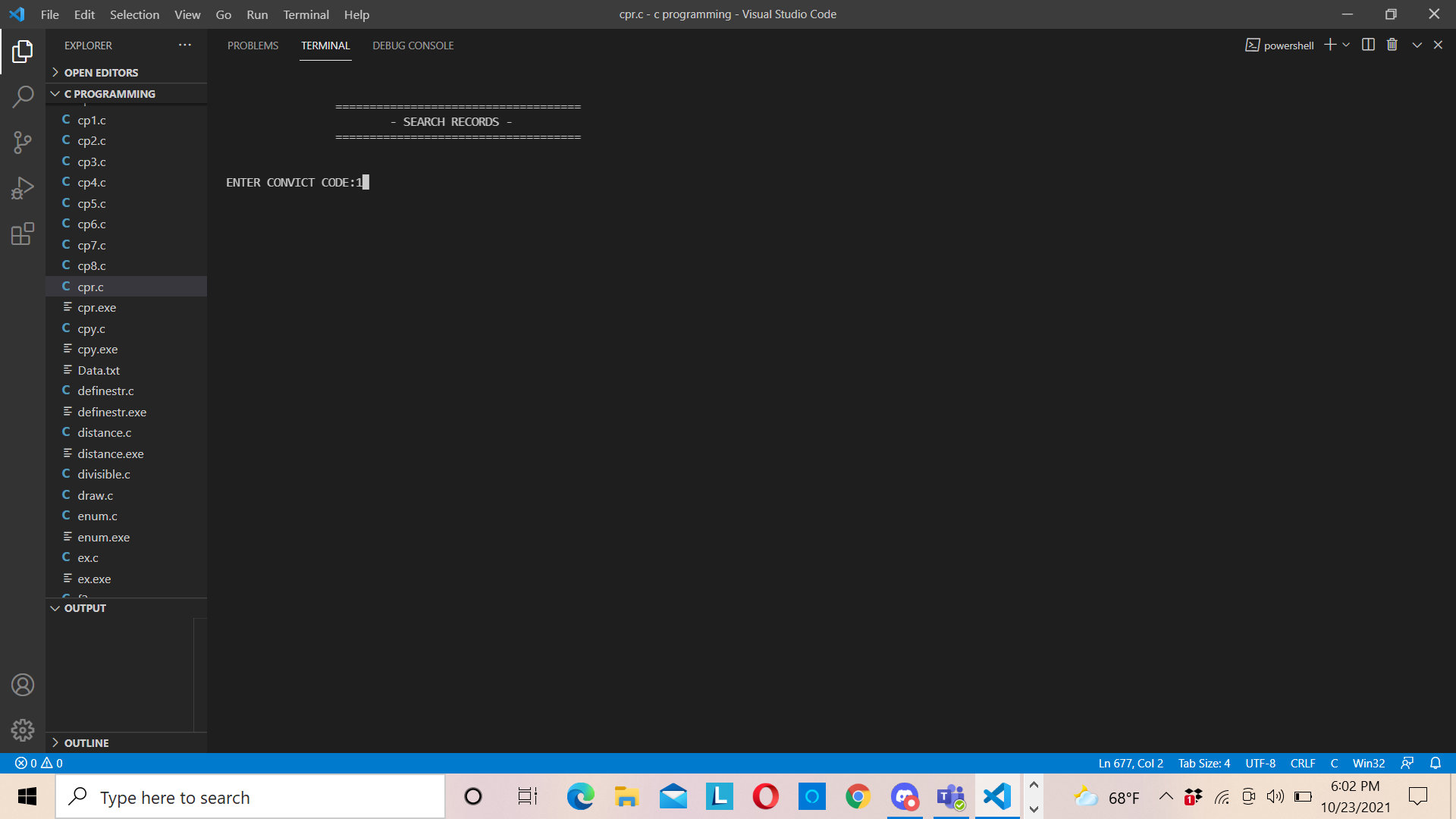
  
 fig: view record

  
 fig4.2.3: add record

  
 fig4.2.4: edit record

  
 fig4.2.5: delete record

  
 fig4.2.6: exit screen

  
 fig4.2.7 search record

# Conclusion and Further Work

## **5.1 Conclusion**

In this way the project was completed with the coordination of our team and help from various sources and under the supervision of our great teacher. This project took us about 5 weeks to complete and be fully working.

## **5.2 Further works:**

This program is not perfect it has various aspects of improvement that can be done to meet the needs as required. Some of the areas might be as follows

* We might make it user to user data interface as a separate datasheet for each user who log in using id and password.
* We can make backspace to delete the wrong letter we enter before pressing enter button.
* We can make the GUI more user friendly
* We can encrypt data for more security

# References

1. File Handling in C, available: <https://data-flair.training/blogs/file-handling-in-c/#:~:text=File%20handling%20in%20C%20refers,for%20future%20reference%20and%20analysis>.
2. C-Arrays, available: <https://www.javatpoint.com/c-array#:~:text=An%20array%20is%20defined%20as,%2C%20double%2C%20float%2C%20etc>.
3. C-pointer, available: <https://www.javatpoint.com/c-pointers#:~:text=The%20pointer%20in%20C%20language,a%20pointer%20is%202%20byte>.
4. Records management, available: <https://en.wikipedia.org/wiki/Records_management>

# Appendix

The source code of the program can also be found in [our github repository](https://github.com/SafalKarkey/first-sem-C-project)

#include<stdio.h>

#include<conio.h>

#include<string.h>

#include<Windows.h>

#include<time.h>

#include<stdlib.h>

*struct* Record{

*char* name[30];

*char* id[10];

*char* weight[10];

*char* height[10];

*char* gender[10];

*struct* DOB{

*int* year;

*int* month;

*int* day;

    } dob;

*char* hair[10];

*char* eye[10];

*char* crime[30];

*char* address[30];

} record;

*void* loadscreen(*int* *time*);

*void* login();

*void* addrecord();

*void* searchrecord();

*void* editrecord();

*void* viewrecord();

*void* deleterecord();

FILE\* fp, \* ft;

*void* main()

{

    system("cls");

    //mkdir("file");

    loadscreen(100);

    login();

    system("cls");

*int* ch;

    //main program loop

    while (1)

    {

        printf("\n\n\n\n\t        ##MAIN MENU##\n");

        printf("   %c            =============              %c\n", 179, 179);

        printf("   %c                                       %c\n", 179, 179);

        printf("   %c  [1] View available records           %c\n", 179, 179);

        printf("   %c  [2] Add a new record                 %c\n", 179, 179);

        printf("   %c  [3] Edit existing Record             %c\n", 179, 179);

        printf("   %c  [4] Delete existing Record           %c\n", 179, 179);

        printf("   %c  [5] Search for a Record              %c\n", 179, 179);

        printf("   %c  [6] Exit                             %c\n", 179, 179);

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

        printf("\n\n\t\t\tChoose an option.....:\t");

        scanf("%d", &ch);

        switch (ch)

        {

        case 1:

            viewrecord();

            break;

        case 2:

            addrecord();

            break;

        case 3:

            editrecord();

            break;

        case 4:

            deleterecord();

            break;

        case 5:

            searchrecord();

            break;

        case 6:

            system("cls");

            printf("\n\n\t\tThank you for using PRMS. Your program will now exit.\n\n ");

            loadscreen(100);

            Sleep(50);

            exit(0);

        default:

            printf("\nYOU ENTERED WRONG CHOICE.");

            printf("\nPRESS ANY KEY TO TRY AGAIN");

            getch();

            break;

        }

        system("cls");

    }

}

*void* loadscreen(*int* *time*)

{

    printf("\n\t\t     \_\_\_\_\_  \_\_\_\_\_  \_\_  \_\_  \_\_\_\_\_ ");

    Sleep(*time*);

    printf("\n\t\t    |  \_\_ \\|  \_\_ \\|  \\/  |/ \_\_\_\_|");

    Sleep(*time*);

    printf("\n\t\t    | |\_\_) | |\_\_) | \\  / | (\_\_\_  ");

    Sleep(*time*);

    printf("\n\t\t    |  \_\_\_/|  \_  /| |\\/| |\\\_\_\_ \\ ");

    Sleep(*time*);

    printf("\n\t\t    | |    | | \\ \\| |  | |\_\_\_\_) |");

    Sleep(*time*);

    printf("\n\t\t    |\_|    |\_|  \\\_\\\_|  |\_|\_\_\_\_\_/ ");

    Sleep(*time*);

    printf(" (c) 2021");

}

*void* login()

{

*int* atmpt = 0, i = 0;

*char* user[10], c;

*char* pword[10], code[10];

    do

    {

        printf("\n\nWelcome to Police Record Management System. Please enter your login info:\n\n");

        printf("\n  <<<<<<<<<<<<<<<<<<<<<<< PRMS LOGIN >>>>>>>>>>>>>>>>>>>>>>>>>  \n");

        printf(" \n                       ENTER USERNAME:-");

        scanf("%s", &user);

        printf(" \n                       ENTER PASSWORD:-");

        while (i < 10)

        {

            pword[i] = getch();

            c = pword[i];

            if (c == 13 || i > 8) break;

            else printf("\*");

            i++;

        }

        pword[i] = '\0';

        i = 0;

        if (strcmp(user, "user") == 0 && strcmp(pword, "pass") == 0)

        {

            printf("  \n\n\n    LOGIN SUCCESSFUL!! ");

            printf("\n\n\n\t\t\tEnter any key to continue...");

            getch();

            break;

        }

        else

        {

            printf("\n        SORRY !!!!  LOGIN IS UNSUCESSFUL");

            atmpt++;

            getch();

            system("cls");

        }

    }while (atmpt <= 2);

    if (atmpt > 2)

    {

        printf("\nYou have exceeded the attempt limit! Contact your technical provider!! ");

        getch();

    }

    system("cls");

}

*void* addrecord()

{

    system("cls");

*char* Y = 'Y', id[10];

*char* filename[30];

*int* exists;

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

    printf("\t\t\t  - ADD RECORDS -");

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

    fp = fopen("filename", "ab+");

    if (fp == NULL)

    {

        fp = fopen("filename", "wb+");

        if (fp == NULL)

        {

            printf("\nSYSTEM ERROR...");

            printf("\nPRESS ANY KEY TO EXIT");

            getch();

            return;

        }

    }

    while (Y == 'Y' || Y == 'y')

    {

        exists = 0;

        fseek(stdin, 0, SEEK\_END);

        printf("\tENTER CONVICT CODE(To check whether it matches or not): ");

        scanf("%s", id);

        rewind(fp);

        while (fread(&record, sizeof(record), 1, fp) == 1)

        {

            if (strcmp(record.id, id) == 0)

            {

                printf("\n\tTHE RECORD ALREADY EXISTS.\n");

                exists = 1;

            }

        }

        if (exists == 0)

        {

            strcpy(record.id, id);

            printf("\n\tENTER NAME OF CONVICT: ");

            fseek(stdin, 0, SEEK\_END);

            gets(record.name);

            printf("\tENTER SEX: ");

            gets(record.gender);

            fseek(stdin, 0, SEEK\_END);

            printf("Enter date of birth...\n");

            printf("Enter year: ");

            scanf("%d", &record.dob.year);

            printf("Enter month: ");

            scanf("%d", &record.dob.month);

            printf("Enter day: ");

            scanf("%d", &record.dob.day);

            fseek(stdin, 0, SEEK\_END);

            printf("\tENTER WEIGHT: ");

            gets(record.weight);

            fseek(stdin, 0, SEEK\_END);

            printf("\tENTER HEIGHT(FT): ");

            gets(record.height);

            fseek(stdin, 0, SEEK\_END);

            printf("\tENTER HAIRCOLOR: ");

            gets(record.hair);

            fseek(stdin, 0, SEEK\_END);

            printf("\tENTER EYECOLOR: ");

            gets(record.eye);

            fseek(stdin,0,SEEK\_END);

            printf("\tENTER CRIME: ");

            gets(record.crime);

            fseek(stdin, 0, SEEK\_END);

            printf("\tCOMPLETE ADDRESS OF POLICE STATION: ");

            gets(record.address);

            fseek(stdin, 0, SEEK\_END);

            fwrite(&record, sizeof(record), 1, fp);

            printf("\nYOUR RECORD IS ADDED...\n");

        }

        printf("\n\tADD ANOTHER RECORD...(Y/N) \t");

        fseek(stdin, 0, SEEK\_END);

        Y = getch();

    }

    fclose(fp);

    printf("\n\n\tPRESS ANY KEY TO RETURN TO PREVIOUS MENU...");

    getch();

}

*void* editrecord()

{

    system("cls");

*char* id[10], select = 'Y', filename[14];

*int* choice, count = 0;

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

    printf("\t\t\t- MODIFY RECORDS -");

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

    do

    {

        printf("\n\tEnter id:");

        fseek(stdin, 0, SEEK\_END);

        gets(id);

        fp = fopen("filename", "rb+");

        while (fread(&record, sizeof(record), 1, fp) == 1)

        {

            if (strcmp(record.id, id) == 0)

            {

                printf("\n %c ID = %s                       ", 179, record.id);

                printf("\n %c Convict's name: %s            ", 179, record.name);

                printf("\n %c Convict's date of birth: %d/%d/%d  ", 179, record.dob.year, record.dob.month, record.dob.day);

                printf("\n %c Convict's gender: %s          ", 179, record.gender);

                printf("\n %c Convict's weight: %s          ", 179, record.weight);

                printf("\n %c Convict's height: %s          ", 179, record.height);

                printf("\n %c Convict's haircolor: %s       ", 179, record.hair);

                printf("\n %c Convict's eyecolor: %s        ", 179, record.eye);

                printf("\n %c Convict's crime: %s           ", 179, record.crime);

                printf("\n %c Address of police station: %s ", 179, record.address);

                printf("\n\n\t\tWhich details would you like to edit?");

                printf("\n %c 1.Name.", 179);

                printf("\n %c 2.Birth year ", 179);

                printf("\n %c 3.Birth month ", 179);

                printf("\n %c 4.Birth day ", 179);

                printf("\n %c 5.Gender", 179);

                printf("\n %c 6.Weight", 179);

                printf("\n %c 7.Height", 179);

                printf("\n %c 8.Haircolor", 179);

                printf("\n %c 9.Eyecolor", 179);

                printf("\n %c 10.Crime", 179);

                printf("\n %c 11.Address of police station", 179);

                printf("\n %c 12.Edit whole record", 179);

                printf("\n %c 13.Go back", 179);

                do

                {

                    printf("\n\tENTER YOUR CHOICE:");

                    fseek(stdin,0,SEEK\_END);

                    scanf("%d", &choice);

                    fseek(stdin,0,SEEK\_END);

                    switch (choice)

                    {

                    case 1:

                        printf("Enter new name: ");

                        gets(record.name);

                        break;

                    case 2:

                        printf("Enter new year: ");

                        scanf("%d", &record.dob.year);

                        break;

                    case 3:

                        printf("Enter new month: ");

                        scanf("%d", &record.dob.month);

                        break;

                    case 4:

                        printf("Enter new birthday: ");

                        scanf("%d", &record.dob.day);

                        break;

                    case 5:

                        printf("Enter new Gender: ");

                        gets(record.gender);

                        break;

                    case 6:

                        printf("Enter new weight: ");

                        gets(record.weight);

                        break;

                    case 7:

                        printf("Enter new height: ");

                        gets(record.height);

                        break;

                    case 8:

                        printf("Enter new haircolor: ");

                        gets(record.hair);

                        break;

                    case 9:

                        printf("Enter new eyecolor: ");

                        gets(record.eye);

                        break;

                    case 10:

                        printf("Enter new crime: ");

                        gets(record.crime);

                        break;

                    case 11:

                        printf("Enter new address: ");

                        gets(record.address);

                        break;

                    case 12:

                        printf("ENTER THE NEW DATA:");

                        printf("Enter new name: ");

                        gets(record.name);

                        printf("Enter new Gender: ");

                        gets(record.gender);

                        printf("Enter new weight: ");

                        gets(record.weight);

                        printf("Enter new height: ");

                        gets(record.height);

                        printf("Enter new haircolor: ");

                        gets(record.hair);

                        printf("Enter new eyecolor: ");

                        gets(record.eye);

                        printf("Enter new crime: ");

                        gets(record.crime);

                        printf("Enter new address: ");

                        gets(record.address);

                        break;

                    case 13:

                        printf("\nPRESS ANY KEY TO GO BACK...\n");

                        getch();

                        return;

                        break;

                    default:

                        printf("\nInvalid input. Try again.\n");

                        break;

                    }

                }while (choice < 1 || choice>13);

                fseek(fp, -sizeof(record), SEEK\_CUR);

                fwrite(&record, sizeof(record), 1, fp);

                //fseek(fp, -sizeof(record), SEEK\_CUR);

                //fread(&record, sizeof(record), 1, fp);

                select = 'Y';

                break;

            }

        }

        if (select == 'Y')

        {

            system("cls");

            printf("\n\t\tEDITING COMPLETED...\n");

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

            printf("THE NEW RECORD IS:\n");

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

            printf("\n %c ID = %s                       ", 179, record.id);

            printf("\n %c Convict's name: %s            ", 179, record.name);

            printf("\n %c Convict's date of birth: %d/%d/%d  ", 179, record.dob.year, record.dob.month, record.dob.day);

            printf("\n %c Convict's gender: %s          ", 179, record.gender);

            printf("\n %c Convict's weight: %s          ", 179, record.weight);

            printf("\n %c Convict's height: %s          ", 179, record.height);

            printf("\n %c Convict's haircolor: %s       ", 179, record.hair);

            printf("\n %c Convict's eyecolor: %s        ", 179, record.eye);

            printf("\n %c Convict's crime: %s           ", 179, record.crime);

            printf("\n %c Address of police station: %s ", 179, record.address);

            fclose(fp);

            printf("\n\n\tWOULD YOU LIKE TO EDIT ANOTHER RECORD.(Y/N)");

            scanf("%c", &select);

            //select++;

        }

        else

        {

            printf("\nTHE RECORD DOES NOT EXIST::\n");

            printf("\nWOULD YOU LIKE TO TRY AGAIN...(Y/N)");

            scanf("%c", &select);

        }

    }while (select == 'Y' || select == 'y');

    fclose(fp);

    printf("\tPRESS ENTER TO EXIT EDITING MENU.");

    getch();

}

*void* deleterecord()

{

    system("cls");

*struct* Record U;

*char* filename[15], another = 'Y', id[16];

*char* pword[10];

*char* c = ' ';

*int* choice, check;

*int* j = 0;

*int* i = 0;

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

    printf("\t\t\t- DELETE RECORDS -");

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

    printf("\nENTER PASSWORD\n");

    /\*int i;

    scanf("%s", pass);\*/

    while (i < 10)

    {

        pword[i] = getch();

        c = pword[i];

        if (c == 13 || i>8) break;

        else printf("\*");

        i++;

    }

    pword[i] = '\0';

    i = 0;

    if (strcmpi(pword, "pass") == 0)

    {

        printf("\n\t\t\*ACCESS GRANTED\*\n\n");

        while (another == 'Y' || another == 'y')

        {

            fp = fopen("filename", "rb");

            if (fp == NULL)

            {

                printf("\nTHE FILE DOES NOT EXIST");

                printf("\nPRESS ANY KEY TO GO BACK.");

                getch();

                return;

            }

            ft = fopen("temp", "wb");

            if (ft == NULL)

            {

                printf("\nSYSTEM ERROR");

                printf("\nPRESS ANY KEY TO GO BACK");

                getch();

                return;

            }

            printf("\n\tENTER THE ID OF RECORD TO BE DELETED:");

            fseek(stdin,0,SEEK\_END);

            gets(id);

            while (fread(&U, sizeof(U), 1, fp) == 1)

            {

                if (strcmp(U.id, id) != 0)

                    fwrite(&U, sizeof(U), 1, ft);

            }

            fclose(ft);

            fclose(fp);

            remove("filename");

            rename("temp", "filename");

            printf("\nDELETED SUCCESFULLY...");

            getch();

            printf("\n\tDO YOU LIKE TO DELETE ANOTHER RECORD.(Y/N):");

            fseek(stdin,0,SEEK\_END);

            scanf("%c", &another);

        }

        printf("\n\n\tPRESS ANY KEY TO EXIT...");

        getch();

    }

    else

    {

        printf("\nSorry!Invalid password\n");

        printf("Return to main menu....");

        getch();

    }

}

*void* searchrecord()

{

    system("cls");

*char* id[16], Y, filename[14];

*char* addnew;

*int* ch;

*int* flag = 0;

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

    printf("\t\t\t- SEARCH RECORDS -");

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

    fp = fopen("filename", "rb");

    do

    {

        fseek(stdin, 0, SEEK\_END);

        printf("\nENTER CONVICT CODE:");

        gets(id);

        system("cls");

        fseek(stdin, 0, SEEK\_END);

        while (fread(&record, sizeof(record), 1, fp) == 1)

        {

            if (strcmpi(record.id, id) == 0)

            {

                printf("\nThe record of ID %s is: \n", id);

                printf("\n");

                printf("\n %c ID = %s                       ", 179, record.id);

                printf("\n %c Convict's name: %s            ", 179, record.name);

                printf("\n %c Convict's gender: %s          ", 179, record.gender);

                printf("\n %c Convict's date of birth: %d/%d/%d:  ", 179, record.dob.year, record.dob.month, record.dob.day);

                printf("\n %c Convict's weight: %s          ", 179, record.weight);

                printf("\n %c Convict's height: %s          ", 179, record.height);

                printf("\n %c Convict's haircolor: %s       ", 179, record.hair);

                printf("\n %c Convict's eyecolor: %s        ", 179, record.eye);

                printf("\n %c Convict's crime: %s           ", 179, record.crime);

                printf("\n %c Address of police station: %s ", 179, record.address);

                flag = 1;

            }

        }

        if (flag == 0) {

            printf("\nNo such record exists. Would you like to add new record?(Y/N) \n");

            scanf("%c", &addnew);

            if (addnew == 'Y' || addnew == 'y') {

                addrecord();

            }

            system("cls");

        }

        fp = fopen("filename", "rb");

        printf("\n\nWOULD YOU LIKE TO CONTINUE searching...(Y/N):");

        fflush(stdin);

        scanf("%c", &Y);

    }while (Y == 'Y' || Y == 'y');

    fclose(fp);

}

*void* viewrecord()

{

        system("cls");

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

        printf("\t\t\t - LIST OF RECORDS -");

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

        fp = fopen("filename", "rb");

        rewind(fp);

        while ((fread(&record, sizeof(record), 1, fp)) == 1)

        {

            printf("\n\n\t\t::PRESS ENTER TO VIEW MORE RECORDS!::\n");

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

            printf("\n %c ID = %s                       ", 179, record.id);

            printf("\n %c Convict's name: %s            ", 179, record.name);

            printf("\n %c Convict's gender: %s          ", 179, record.gender);

            printf("\n %c Convict's date of birth: %d/%d/%d:  ", 179, record.dob.year, record.dob.month, record.dob.day);

            printf("\n %c Convict's weight: %s          ", 179, record.weight);

            printf("\n %c Convict's height: %s          ", 179, record.height);

            printf("\n %c Convict's haircolor: %s       ", 179, record.hair);

            printf("\n %c Convict's eyecolor: %s        ", 179, record.eye);

            printf("\n %c Convict's crime: %s           ", 179, record.crime);

            printf("\n %c Address of police station: %s ", 179, record.address);

            getch();

        }

        fclose(fp);

        getch();

}