MINI PROJECT REPORT

Raj Gupta

November 16, 2022

Abstract

This report contains the all information about my mini project in c language, named Second-hand bookselling and buying management system.

Contents

1 Introduction

I made the second hand book buying and selling management system.In this project the seller can register book and user can buy these books. Click Here to open the Code.

2 Statistical Information

Starting date/time- 11th Nov.2022/20:00
Starting date/time- 15th Nov.2022/20:00
Total Time Required- 2-3 HR a day
Total line of code- 396 lines
Number of functions -8 functions

3 Function Description

The *Developerintro*() is to show the developer information.
The *mainmenu*() is used to give choice and according to that choice the other functions are performed.
The *SellerBook*() is used to store the book by the sellers.
The *BuyABook*() is used by the buyer to buy the books.
The *search*() is used to search the book and the author name and matches with the books registered by the seller.
The *BuyersDisplay*() is used to display the books bought.
The*SellersDisplay*() is used to get the information about the books registered by the seller.
The*setcolor* is used to set change the color to desired one.

4 code in C

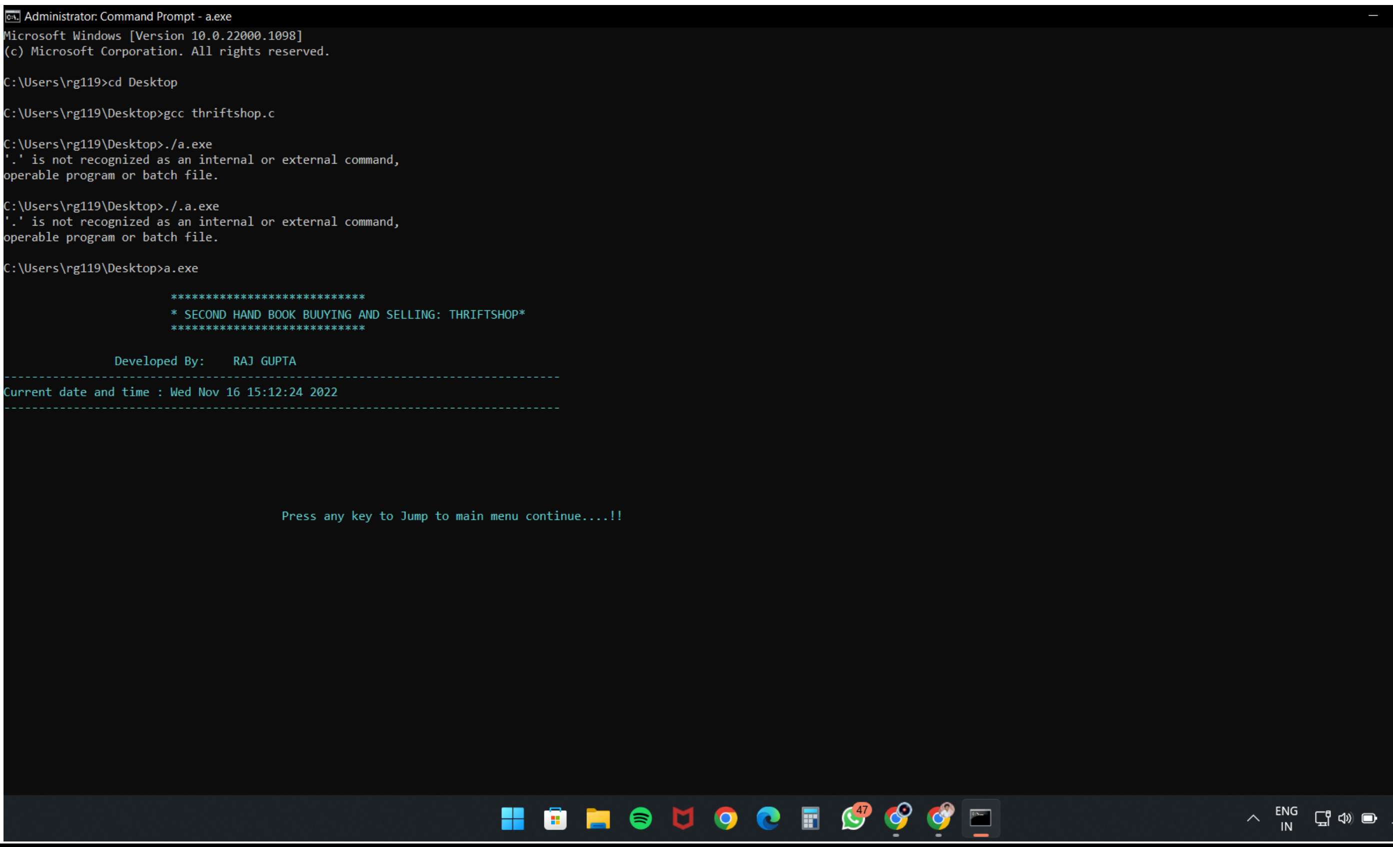
code:

```c
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <windows.h>
#include <time.h>
#include <ctype.h>
#include <string.h>


// Function Declaration
void Developer_intro();
void main_menu();
void Seller_Book();
void Buy_A_Book();
int search();
void Buyers_Display();
void Sellers_Display();

void setcolor(int ForgC)
{
        WORD wColor;
        HANDLE hStdOut = GetStdHandle(STD_OUTPUT_HANDLE);
        CONSOLE_SCREEN_BUFFER_INFO csbi;

        if (GetConsoleScreenBufferInfo(hStdOut, &csbi))
        {
                wColor = (csbi.wAttributes & 0xB0) + (ForgC & 0x0B);
                //        SetConsoleTextAttributes(hStdOut,wColor);
                SetConsoleTextAttribute(hStdOut, wColor);
        }
}

struct Seller_Details // STRUCTURE DECLARATION
{

        char Name[20];
        char Name_of_Book[10];
        char Author_name[10];
        char Address[25];
        char Email[20];
        char Phoneno[10];
        char Price[5];

};
```

```
struct Buyer_Details // STRUCTURE DECLARATION
{

        char Name[20];
        char Name_of_Book[10];
        char Author_name[10];
        char Address[25];
        char Email[20];
        char Phoneno[15];

};

// start of Main function
int main()
{
        Developer_intro();
        main_menu();
        return 0;
}
// End of the Main function

// For developer Introduction
void Developer_intro()
{       time_t t;
        time(&t);
        setcolor(15);
        printf("\n\t\t\t**************************");
        printf("\n\t\t\t* SECOND HAND BOOK BUUYING AND SELLING: THRIFTSHOP*")
        printf("\n\t\t\t**************************");
        printf("\n\n\t\tDeveloped By:");
        printf("\t RAJ GUPTA\n");
        for(int i=0;i<80;i++)
                printf("-");
            printf("\nCurrent date and time : %s",ctime(&t));
            for(int i=0;i<80;i++)
                printf("-");
        printf("\n\n\n\n\n\n\n\t\t\t\t\tPress any key to Jump to main menu co
        getchar();
}

// For Displaying the main menu

void main_menu()
{
        time_t t;
        time(&t);
        short int choice;
        while (choice != 5)
        {
```

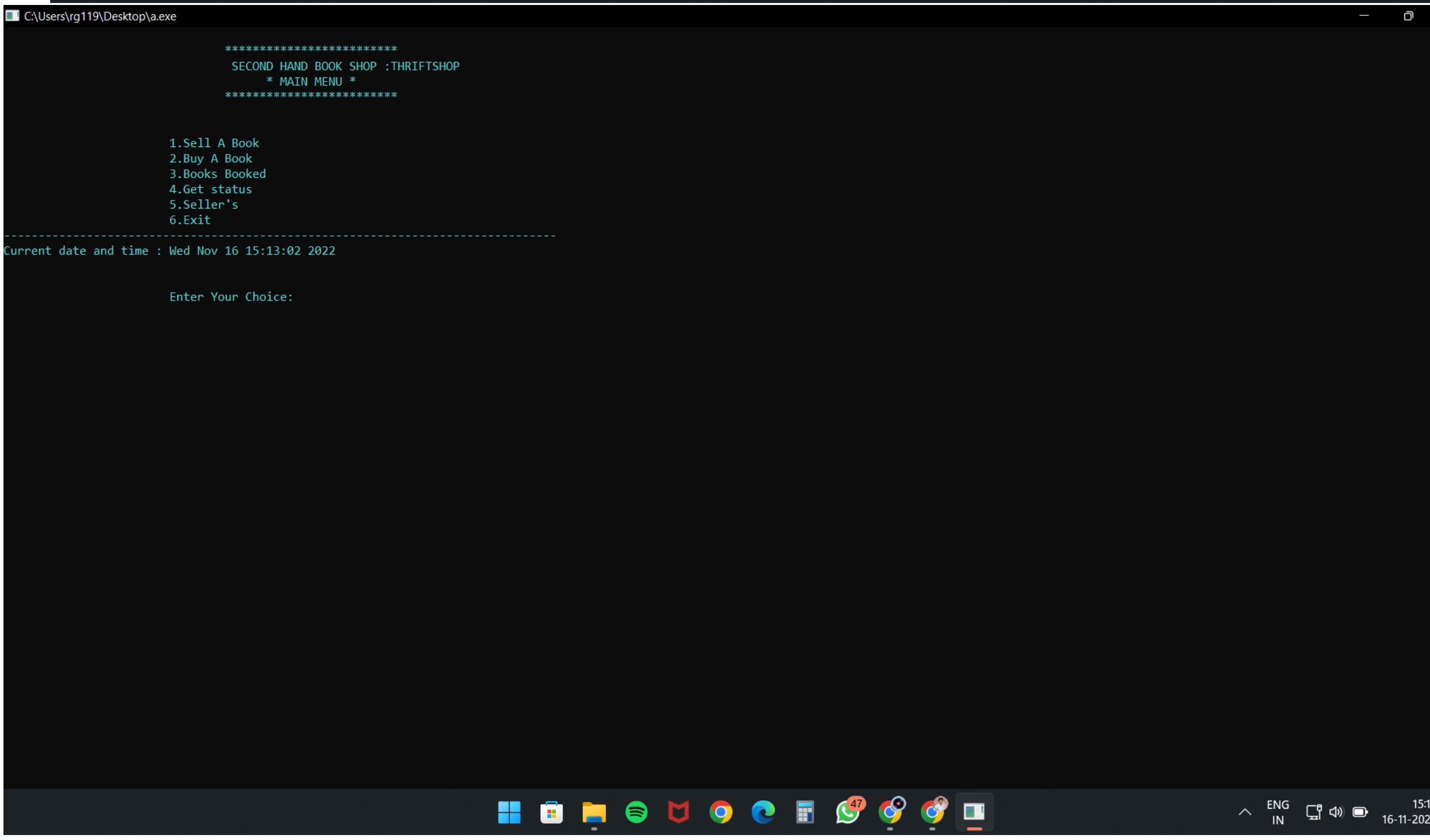
```
    system("cls");
    printf("\n\t\t\t\t*************************");
    printf("\n\t\t\t\t SECOND HAND BOOK SHOP : THRIFTSHOP ");
    printf("\n\t\t\t\t       * MAIN MENU *");
    printf("\n\t\t\t\t*************************");
    printf("\n\n\n\t\t\t1. Sell A Book");
    printf("\n\t\t\t2.Buy A Book");
    printf("\n\t\t\t3.Books Booked ");
    printf("\n\t\t\t4.Get status");
    printf("\n\t\t\t5.Seller's");
    printf("\n\t\t\t6.Exit\n");


    for(int i=0;i<80;i++)
    printf("-");
printf("\nCurrent date and time : %s",ctime(&t));
    printf("\n\n\t\t\tEnter Your Choice: ");
    scanf("%hu", &choice);


    switch (choice)
    {

    case 1:
            Seller_Book();
            break;

    case 2:
            Buy_A_Book();
            break;

    case 3:
            Buyers_Display();
            break;

    case 4: // Delivery();
            break;

    case 5:
            Sellers_Display();
            break;
    case 6:
            system("cls");
            printf("\n\n\t *****THANK YOU*****");
            printf("\n\t FOR TRUSTING OUR SERVICE");
            //      Sleep(2000);
            exit(0);
            break;

    default:
```

```
                {

                        printf("\n\n\t\t\tWrong␣choice.....!!!");
                        printf("\n\t\t\tPress␣any␣key␣to␣␣␣continue....!!");

                        getch();
                }
                }
        }
}

// End of displaying main meu


// Seller function
void Seller_Book()
{
        FILE *f;
        char test;
        f = fopen("sellers_details.txt", "a+");
        if (f == 0)
        {
                f = fopen("sellers_details.txt", "w+");
                system("cls");
                printf("Please␣hold␣on␣while␣we␣set␣our␣database␣in␣your␣com
                printf("\n␣Process␣completed␣press␣any␣key␣to␣continue!!␣");
                getch();
        }
        while (1)
        {       struct Seller_Details input;


                system("cls");
                printf("\n␣Enter␣Seller␣Details:");
                printf("\n**************************\n");
                printf("Enter␣Name:\n");
                scanf("%s", input.Name);
                fflush(stdin);
                printf("Enter␣Subject␣of␣Book:\n");
                scanf("%s", input.Name_of_Book);
                fflush(stdin);
                printf("Enter␣Author␣Name:\n");
                fgets(input.Author_name, 10, stdin);
                printf("Enter␣Address:\n");
                fgets(input.Address, 25, stdin);
                fflush(stdin);
                printf("Enter␣Email:\n");
                scanf("␣%s", input.Email);
                printf("Enter␣Phone␣Number:\n");
                scanf("%s", input.Phoneno);
                fflush(stdin);
                printf(”Enter␣Price(\’x\’Rs):\n”);
                scanf(”%s”, input.Price);
                fflush(stdin);
                fwrite(&input, sizeof(struct Seller_Details), 1, f);
                fputs(”\n”, f);
                printf(”\n\n1␣Book␣is␣successfully␣Added!!”);
                printf(”\n␣Press␣esc␣key␣to␣exit,␣␣any␣other␣key␣to␣add␣anoth
                test = getche();
                if (test == 27)
                        break;
        }
        fclose(f);
}

// Seller function end

// Seller display function
void Sellers_Display()
{
        FILE *f;

        char test;
struct Seller_Details input;
        if ((f = fopen(”sellers_details.txt”, ”r”)) == NULL)
        {
                exit(0);
        }

                        system(”cls”);
                        printf(”NAME\t␣”);
                        printf(”subject\t”);
                        printf(”Author\t”);
                        printf(”\tADDRESS␣”);
                        printf(”\tEMAIL␣”);
                        printf(”\tPHONENUMBER␣”);
                        printf(”\t␣PRICE␣\n”);

                        for (int i = 0; i < 118; i++)
                            printf(”–”);
while (1)
{


                        while (fread(&input, sizeof(struct Seller_Details), 1
                        {
                            printf(”\n%s␣\t\t␣%s\t\t␣%s\t\t␣%s\t\t␣%s\t\t
                        }
                        printf(”\n␣Press␣esc␣key␣to␣exit”);
                test = getche();
        if (test == 27)
            break;
}
            fclose(f);

    }
// Seller display function ends


// Buyer display function
void Buyers_Display()
{
    struct Buyer_Details inp;
    FILE *f;
    char test;
    if ((f = fopen("Buyers_details.txt", "r")) == NULL)
    {
        exit(0);
    }

            system("cls");
            printf("NAME\t␣");
            printf("subject\t");
            printf("Author\t");
            printf("\tADDRESS␣");
            printf("\tEMAIL␣");
            printf("\tPHONENUMBER\n");


            for (int i = 0; i < 118; i++)
                printf("-");
while (1)
{


            while (fread(&inp, sizeof(struct Buyer_Details), 1, f
            {
                printf("\n%s␣\t\t␣%s\t\t␣%s\t\t␣%s\t\t␣%s\t\t
            }
            printf("\n␣Press␣esc␣key␣to␣exit");
        test = getche();
        if (test == 27)
            break;
}
            fclose(f);
}


// Buyer display function ends
// Buyer function
void Buy_A_Book()
{
struct Buyer_Details inp;
        FILE *f;
        char test;
        f = fopen("Buyers_details.txt", "a+");
        if (f == 0)
        {
                f = fopen("Buyers_details.txt", "w+");
                system("cls");
                printf("Please␣hold␣on␣while␣we␣set␣our␣database␣in␣your␣comp
                printf("\n␣Process␣completed␣press␣any␣key␣to␣continue!!␣");
                getch();
        }
        while (1)
        {
                system("cls");
                printf("\n␣Enter␣Buyer␣Details:");
                printf("\n**************************\n");
                int flag = search();
                if (flag == 0)
                {
                        printf("\n␣Enter␣Name:\n");
                        scanf("%s", inp.Name);
                        fflush(stdin);
                        printf("Enter␣Address:\n");
                        scanf("%s", inp.Address);
                        printf("Enter␣Phone␣Number:\n");
                        scanf("%s", inp.Phoneno);
                        printf("Enter␣Email:\n");
                        scanf("␣%s", inp.Email);
                        fflush(stdin);
                        fwrite(&inp, sizeof(struct Buyer_Details), 1, f);
                }
                printf("\n\nBook␣is␣successfully␣booked!!");
                printf("\n␣Press␣esc␣key␣to␣exit,␣␣any␣other␣key␣to␣add␣anoth
                test = getche();
                if (test == 27)
                        break;
        }
        fclose(f);
}
// Buyer function ends

//search function starts
int search()
{
        struct Buyer_Details inp;
        struct Seller_Details input;
        system("cls");
        FILE *f;
        char name_of_book[20];
        char Author_name[20];
        int flag = 1;
        int a, b;

        f = fopen("sellers_details.txt", "r+");
        if (f == 0)
        {
                exit(0);
        }
        printf("Enter the Book name: \n");
        scanf("%s", name_of_book);
        fflush(stdin);
        printf("Enter the Author name: \n");
        scanf("%s", Author_name);
        while (fread(&input, sizeof(struct Seller_Details), 1, f) == 1)
        {
                a = strcmp(input.Name_of_Book, name_of_book);
                b = strcmp(input.Author_name, Author_name);
                if ((a && b) != 0)
                {
                        flag = 0;
                        printf("\n\tBook Found\n ");
                           strcpy(inp.Name_of_Book, name_of_book);
                        strcpy(inp.Author_name, Author_name);
                        flag = 0;
                        break;
                }
        }
        if (flag == 1)
        {
                printf("\n\tRequested Book could not be found!");
        }
        fclose(f);
        return flag;
}


//search function ends
```



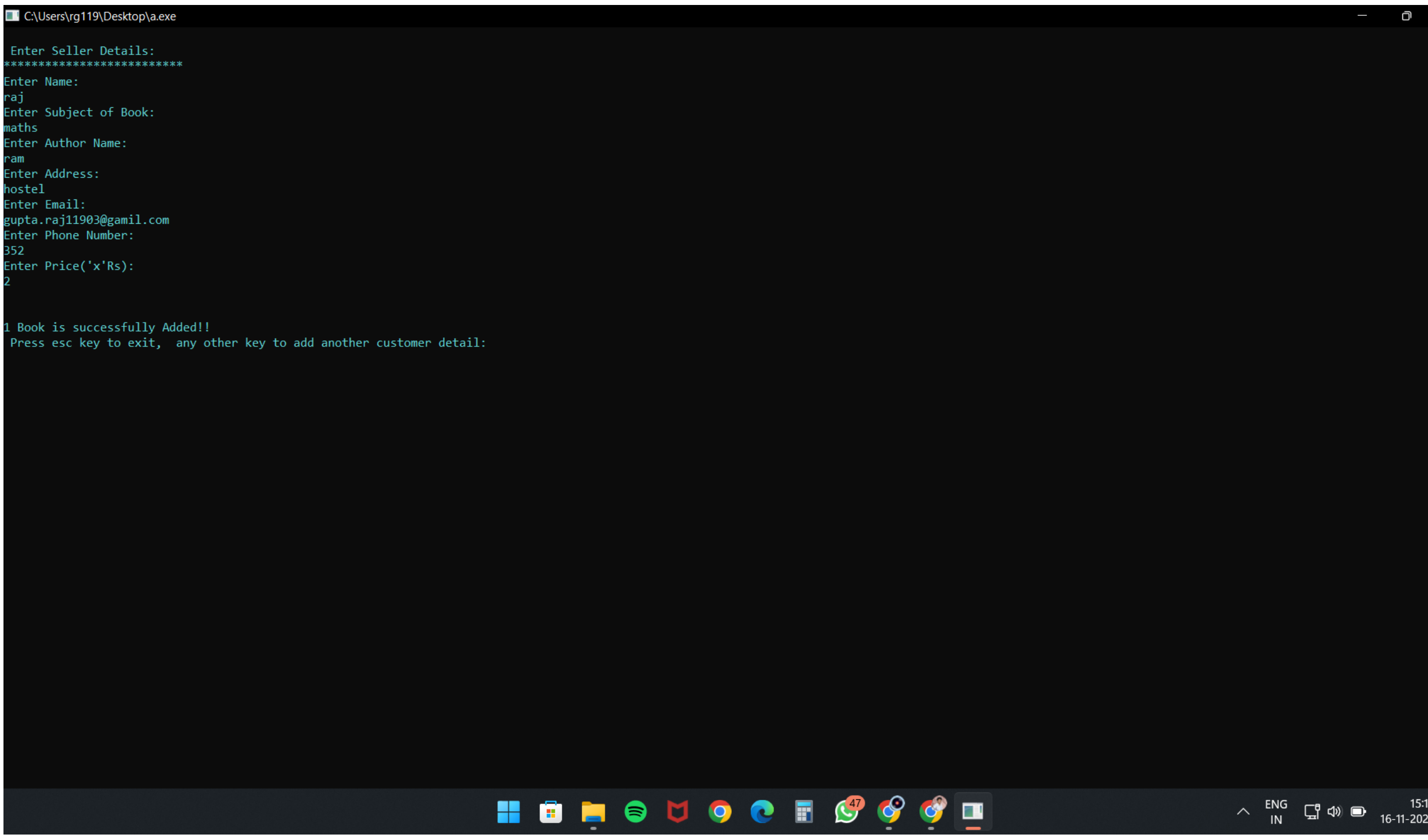
Administrator: Command Prompt - a.exe

```
Microsoft Windows [Version 10.0.22000.1098]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rg119>cd Desktop

C:\Users\rg119\Desktop>gcc thriftshop.c

C:\Users\rg119\Desktop>./a.exe
'.' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\rg119\Desktop>./.a.exe
'.' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\rg119\Desktop>a.exe

                        ****************************
                        * SECOND HAND BOOK BUUYING AND SELLING: THRIFTSHOP*
                        ****************************

                Developed By:    RAJ GUPTA
--------------------------------------------------------------------------------
Current date and time : Wed Nov 16 15:12:24 2022
--------------------------------------------------------------------------------



                                        Press any key to Jump to main menu continue....!!
```

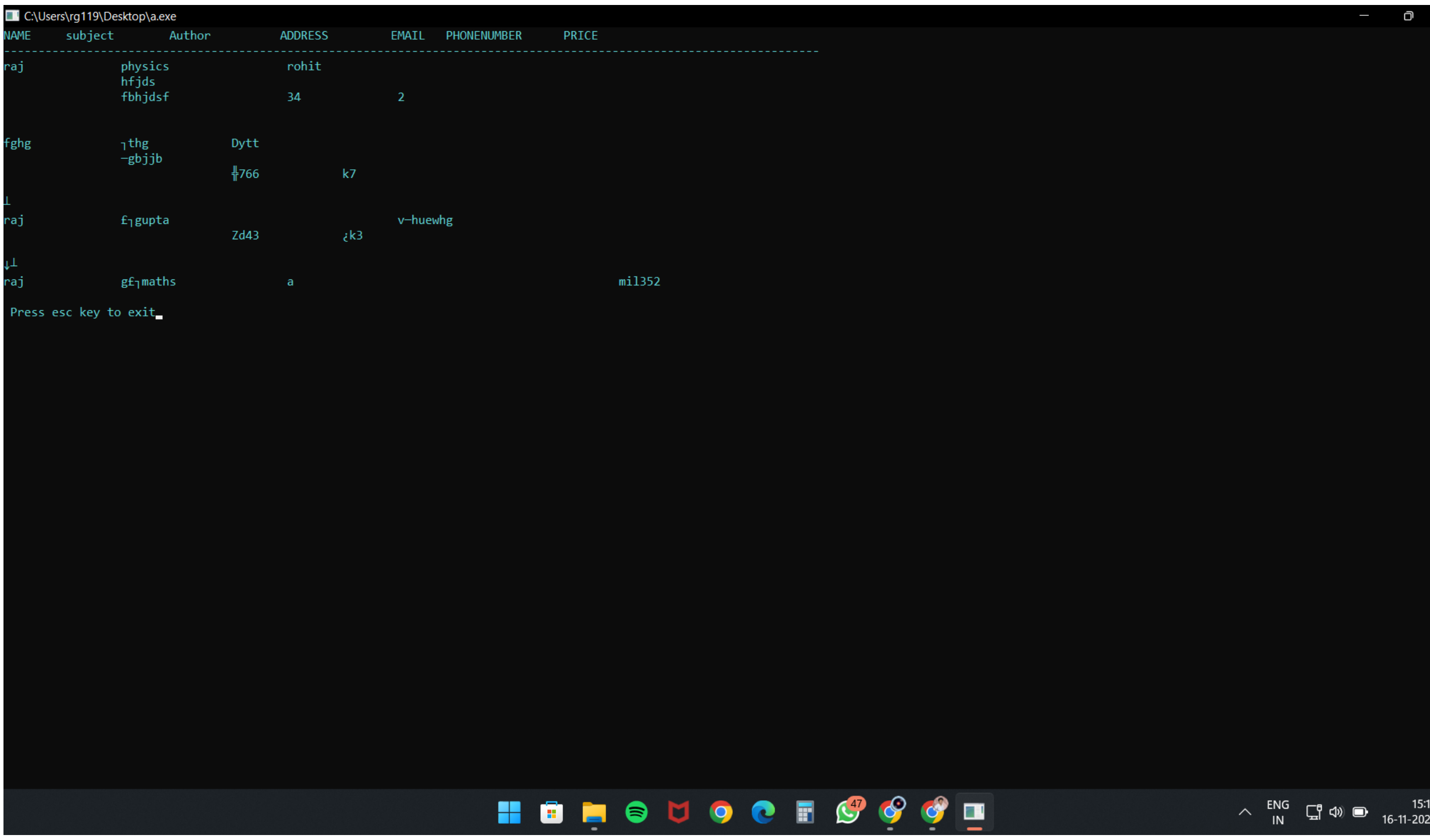


C:\Users\rg119\Desktop\a.exe

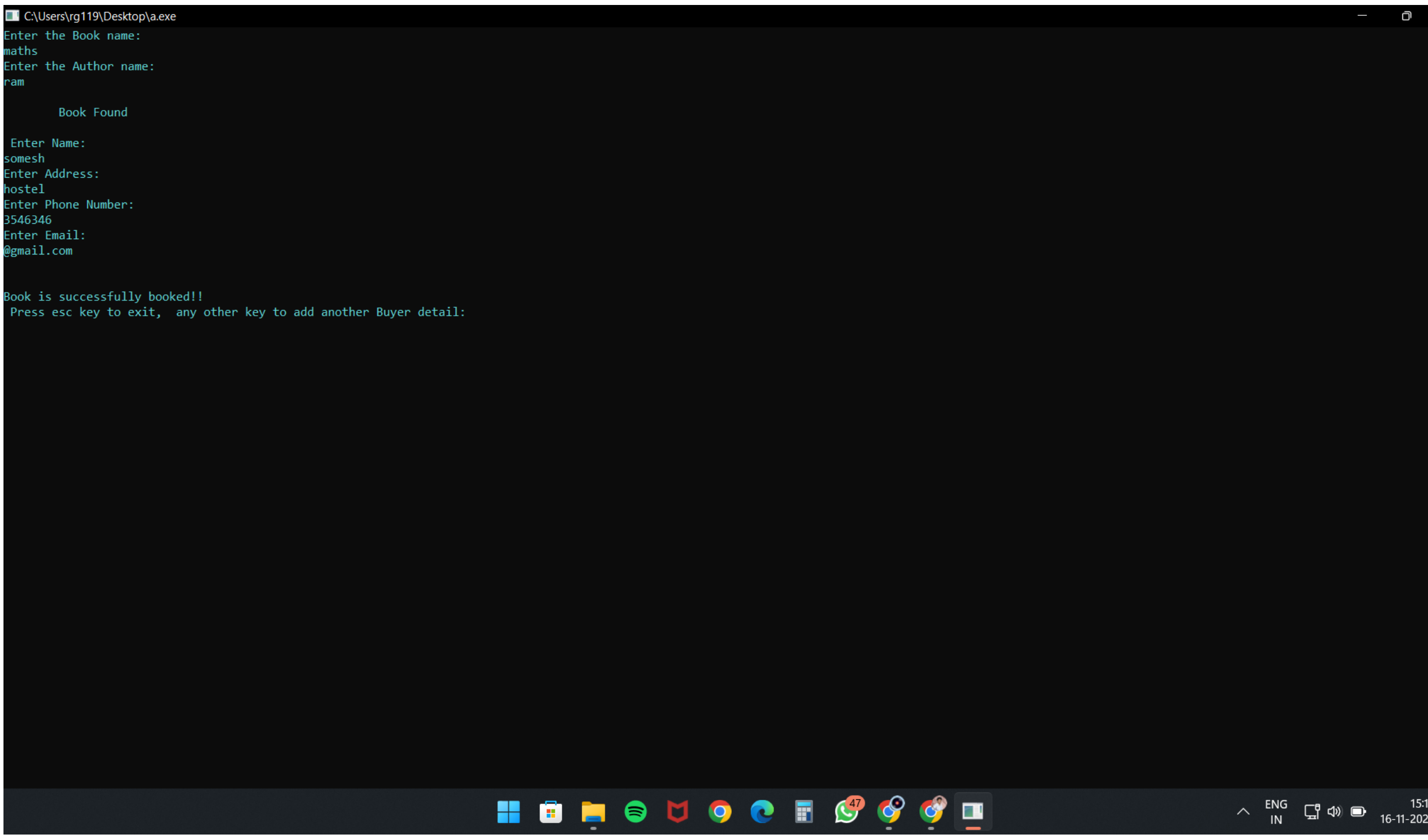
```
                                ************************
                                 SECOND HAND BOOK SHOP :THRIFTSHOP
                                      * MAIN MENU *
                                ************************


                        1.Sell A Book
                        2.Buy A Book
                        3.Books Booked
                        4.Get status
                        5.Seller's
                        6.Exit
--------------------------------------------------------------------------------
Current date and time : Wed Nov 16 15:13:02 2022


                        Enter Your Choice:
```



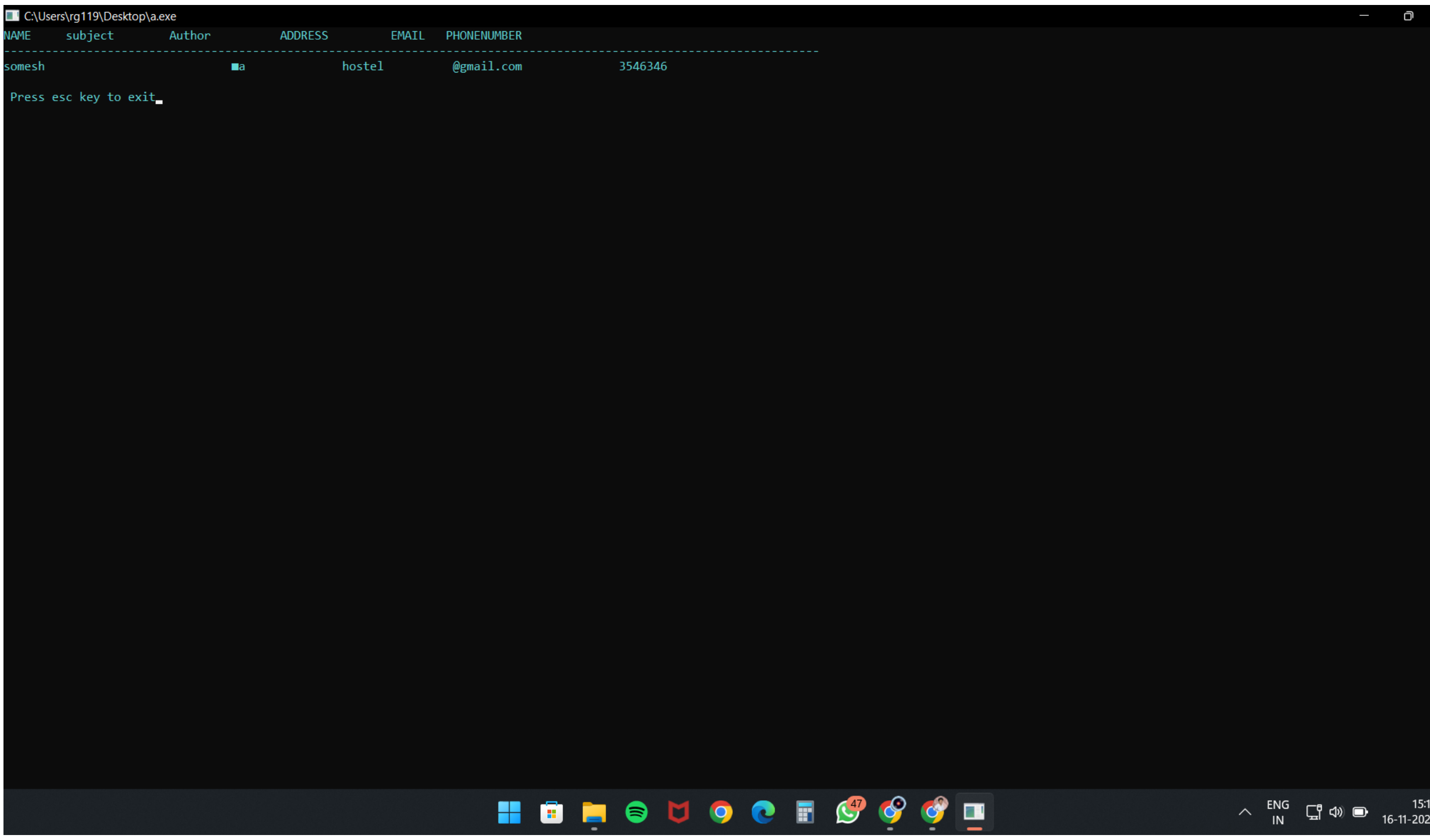
```
C:\Users\rg119\Desktop\a.exe

 Enter Seller Details:
**************************
Enter Name:
raj
Enter Subject of Book:
maths
Enter Author Name:
ram
Enter Address:
hostel
Enter Email:
gupta.raj11903@gamil.com
Enter Phone Number:
352
Enter Price('x'Rs):
2


1 Book is successfully Added!!
 Press esc key to exit,  any other key to add another customer detail:
```



```
C:\Users\rg119\Desktop\a.exe
NAME     subject        Author          ADDRESS         EMAIL   PHONENUMBER      PRICE
----------------------------------------------------------------------------------------------------------------------
raj              physics                 rohit
                 hfjds
                 fbhjdsf                 34              2


fghg             ┐thg           Dytt
                 ─gbjjb
                                ╬766            k7

┴
raj              £┐gupta                                 v─huewhg
                                Zd43            ¿k3

↓┴
raj              g£┐maths                a                                              mil352

 Press esc key to exit_
```



```
C:\Users\rg119\Desktop\a.exe
Enter the Book name:
maths
Enter the Author name:
ram

        Book Found

 Enter Name:
somesh
Enter Address:
hostel
Enter Phone Number:
3546346
Enter Email:
@gmail.com


Book is successfully booked!!
 Press esc key to exit,  any other key to add another Buyer detail:
```

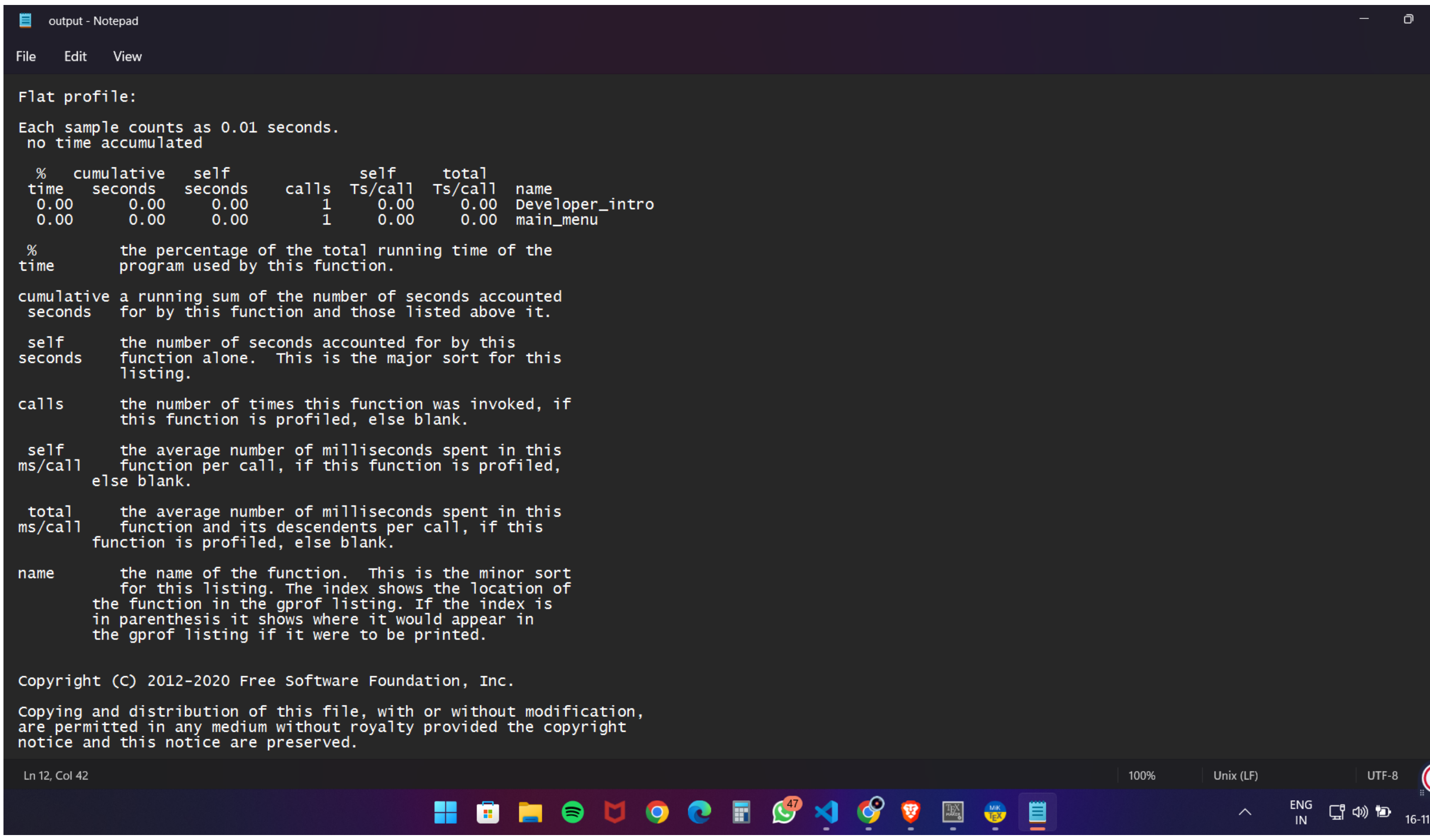


```
C:\Users\rg119\Desktop\a.exe
NAME     subject        Author          ADDRESS         EMAIL   PHONENUMBER
-----------------------------------------------------------------------------------------------------------------------
somesh                           ■a              hostel          @gmail.com              3546346

 Press esc key to exit_
```

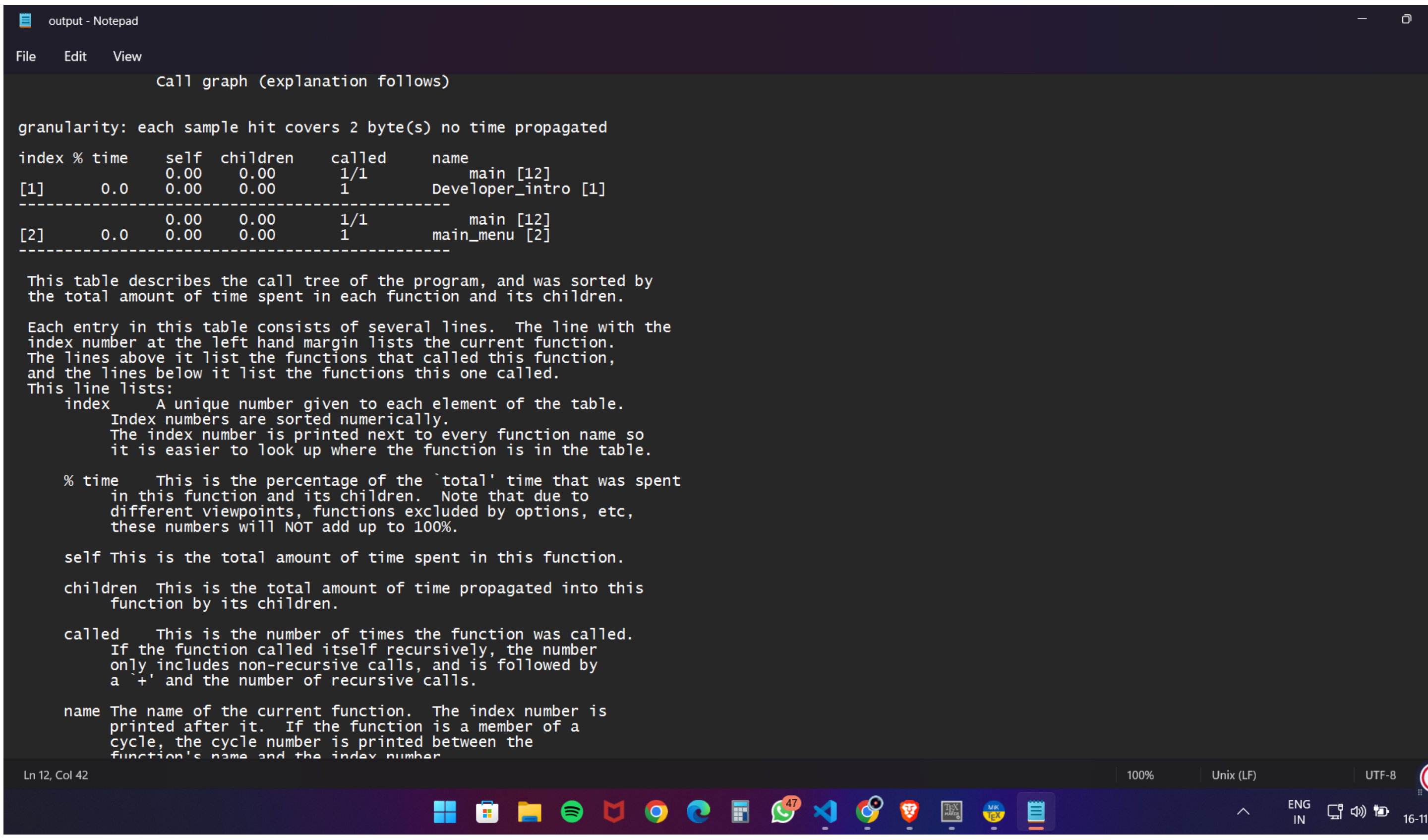
5 Profiling

profiling code: ...

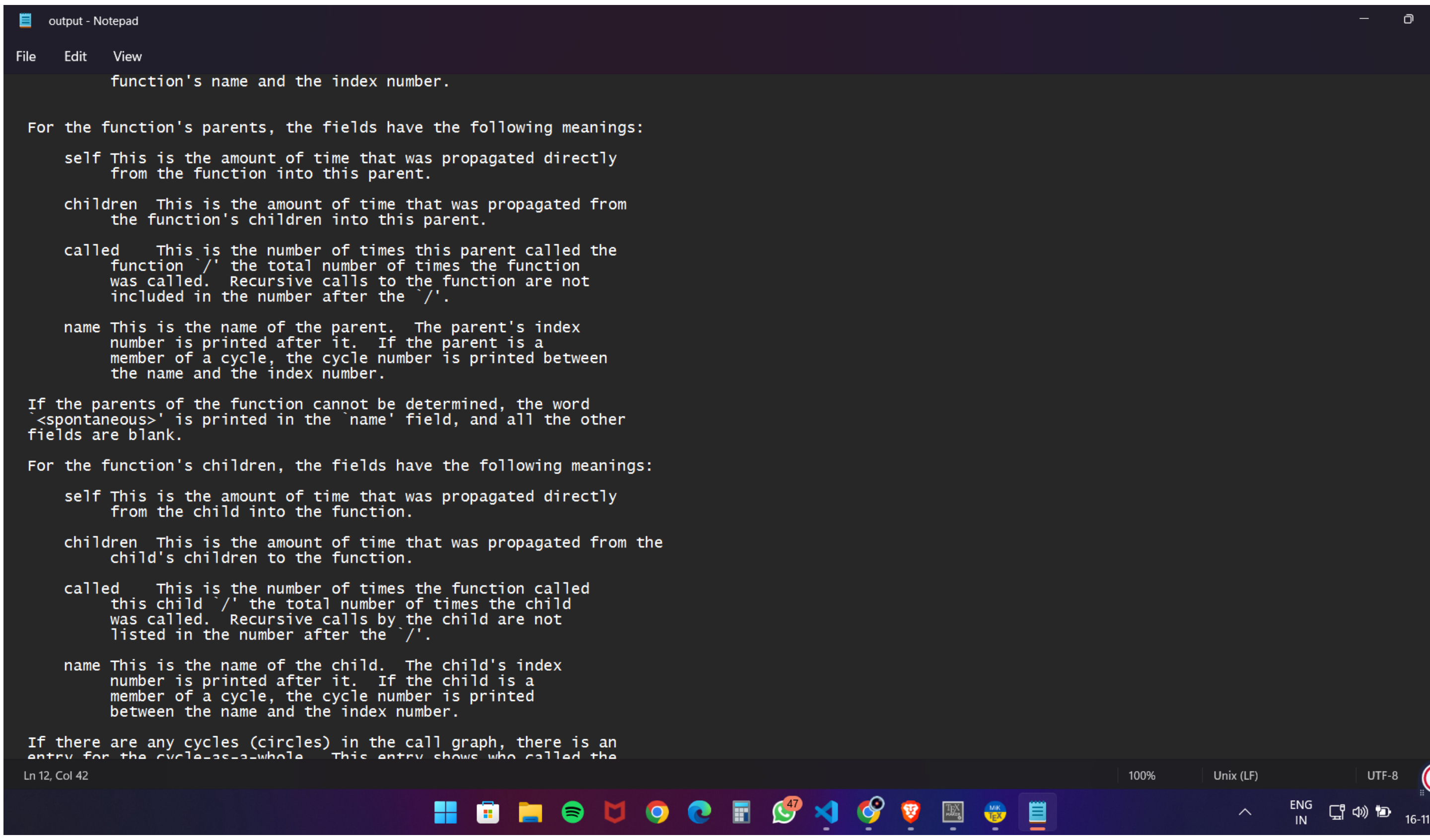
1. **gcc -Wall -pg thriftshop.c -o test**
2. **./test** ...will create gmon.out file
3. **gprof test gmon.out** > **output** ...this will convert gmon.out to readable format



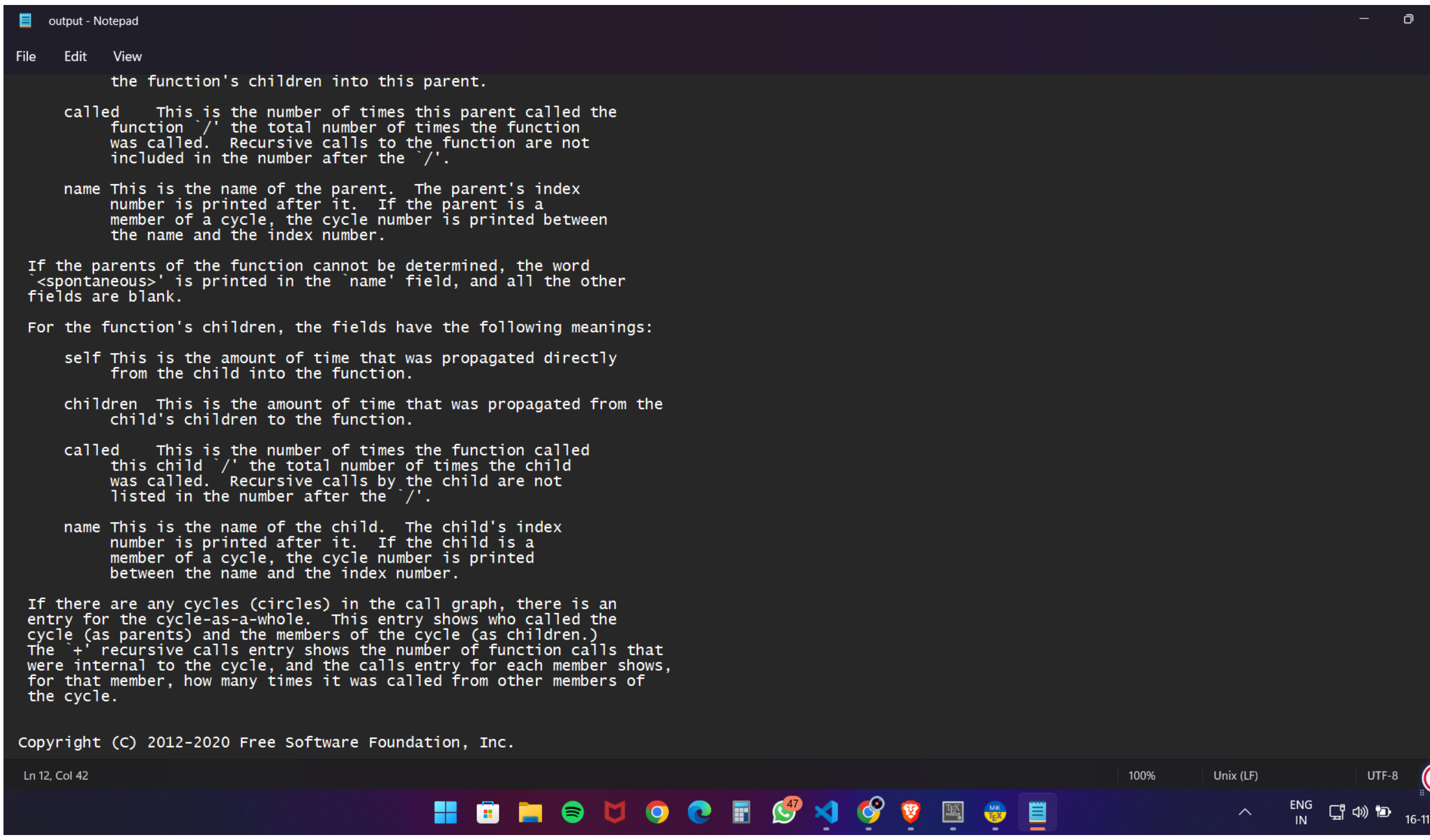
```
Flat profile:

Each sample counts as 0.01 seconds.
 no time accumulated

  %   cumulative   self              self     total
 time   seconds   seconds    calls  Ts/call  Ts/call  name
  0.00      0.00     0.00        1     0.00     0.00  Developer_intro
  0.00      0.00     0.00        1     0.00     0.00  main_menu

 %         the percentage of the total running time of the
time       program used by this function.

cumulative a running sum of the number of seconds accounted
 seconds   for by this function and those listed above it.

 self      the number of seconds accounted for by this
seconds    function alone.  This is the major sort for this
           listing.

calls      the number of times this function was invoked, if
           this function is profiled, else blank.

 self      the average number of milliseconds spent in this
ms/call    function per call, if this function is profiled,
	   else blank.

 total     the average number of milliseconds spent in this
ms/call    function and its descendents per call, if this
	   function is profiled, else blank.

name       the name of the function.  This is the minor sort
           for this listing. The index shows the location of
	   the function in the gprof listing. If the index is
	   in parenthesis it shows where it would appear in
	   the gprof listing if it were to be printed.


Copyright (C) 2012-2020 Free Software Foundation, Inc.

Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.
```



```
                Call graph (explanation follows)


granularity: each sample hit covers 2 byte(s) no time propagated

index % time    self  children    called     name
                0.00    0.00       1/1           main [12]
[1]      0.0    0.00    0.00       1         Developer_intro [1]
-----------------------------------------------
                0.00    0.00       1/1           main [12]
[2]      0.0    0.00    0.00       1         main_menu [2]
-----------------------------------------------

 This table describes the call tree of the program, and was sorted by
 the total amount of time spent in each function and its children.

 Each entry in this table consists of several lines.  The line with the
 index number at the left hand margin lists the current function.
 The lines above it list the functions that called this function,
 and the lines below it list the functions this one called.
 This line lists:
     index      A unique number given to each element of the table.
                Index numbers are sorted numerically.
                The index number is printed next to every function name so
                it is easier to look up where the function is in the table.

     % time     This is the percentage of the `total' time that was spent
                in this function and its children.  Note that due to
                different viewpoints, functions excluded by options, etc,
                these numbers will NOT add up to 100%.

     self       This is the total amount of time spent in this function.

     children   This is the total amount of time propagated into this
                function by its children.

     called     This is the number of times the function was called.
                If the function called itself recursively, the number
                only includes non-recursive calls, and is followed by
                a `+' and the number of recursive calls.

     name       The name of the current function.  The index number is
                printed after it.  If the function is a member of a
                cycle, the cycle number is printed between the
```



```
                function's name and the index number.


 For the function's parents, the fields have the following meanings:

     self       This is the amount of time that was propagated directly
                from the function into this parent.

     children   This is the amount of time that was propagated from
                the function's children into this parent.

     called     This is the number of times this parent called the
                function `/' the total number of times the function
                was called.  Recursive calls to the function are not
                included in the number after the `/'.

     name       This is the name of the parent.  The parent's index
                number is printed after it.  If the parent is a
                member of a cycle, the cycle number is printed between
                the name and the index number.

 If the parents of the function cannot be determined, the word
 `<spontaneous>' is printed in the `name' field, and all the other
 fields are blank.

 For the function's children, the fields have the following meanings:

     self       This is the amount of time that was propagated directly
                from the child into the function.

     children   This is the amount of time that was propagated from the
                child's children to the function.

     called     This is the number of times the function called
                this child `/' the total number of times the child
                was called.  Recursive calls by the child are not
                listed in the number after the `/'.

     name       This is the name of the child.  The child's index
                number is printed after it.  If the child is a
                member of a cycle, the cycle number is printed
                between the name and the index number.

 If there are any cycles (circles) in the call graph, there is an
```

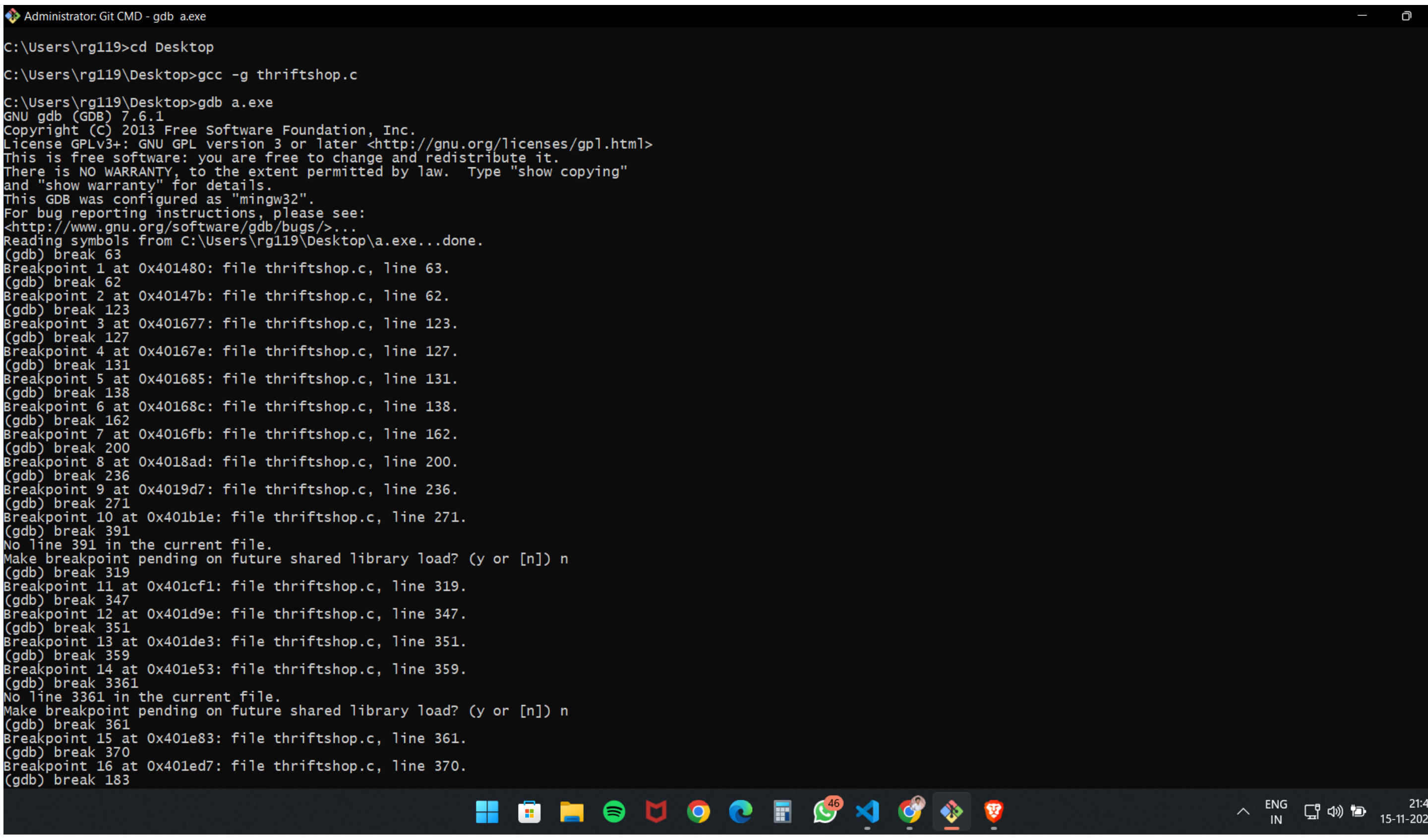


```
         the function's children into this parent.

    called    This is the number of times this parent called the
         function `/' the total number of times the function
         was called.  Recursive calls to the function are not
         included in the number after the `/'.

    name This is the name of the parent.  The parent's index
         number is printed after it.  If the parent is a
         member of a cycle, the cycle number is printed between
         the name and the index number.

 If the parents of the function cannot be determined, the word
 `<spontaneous>' is printed in the `name' field, and all the other
 fields are blank.

 For the function's children, the fields have the following meanings:

    self This is the amount of time that was propagated directly
         from the child into the function.

    children  This is the amount of time that was propagated from the
         child's children to the function.

    called    This is the number of times the function called
         this child `/' the total number of times the child
         was called.  Recursive calls by the child are not
         listed in the number after the `/'.

    name This is the name of the child.  The child's index
         number is printed after it.  If the child is a
         member of a cycle, the cycle number is printed
         between the name and the index number.

 If there are any cycles (circles) in the call graph, there is an
 entry for the cycle-as-a-whole.  This entry shows who called the
 cycle (as parents) and the members of the cycle (as children.)
 The `+' recursive calls entry shows the number of function calls that
 were internal to the cycle, and the calls entry for each member shows,
 for that member, how many times it was called from other members of
 the cycle.


Copyright (C) 2012-2020 Free Software Foundation, Inc.
```

6 Debugging

Debugging code: . . .

1. **gcc -g thriftshop.c**
2. **gdb a.exe** . . . gdb will start
3. **break linenumber**
4. **run**
5. **q or quit**



```
Administrator: Git CMD - gdb  a.exe

C:\Users\rg119>cd Desktop

C:\Users\rg119\Desktop>gcc -g thriftshop.c

C:\Users\rg119\Desktop>gdb a.exe
GNU gdb (GDB) 7.6.1
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "mingw32".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from C:\Users\rg119\Desktop\a.exe...done.
(gdb) break 63
Breakpoint 1 at 0x401480: file thriftshop.c, line 63.
(gdb) break 62
Breakpoint 2 at 0x40147b: file thriftshop.c, line 62.
(gdb) break 123
Breakpoint 3 at 0x401677: file thriftshop.c, line 123.
(gdb) break 127
Breakpoint 4 at 0x40167e: file thriftshop.c, line 127.
(gdb) break 131
Breakpoint 5 at 0x401685: file thriftshop.c, line 131.
(gdb) break 138
Breakpoint 6 at 0x40168c: file thriftshop.c, line 138.
(gdb) break 162
Breakpoint 7 at 0x4016fb: file thriftshop.c, line 162.
(gdb) break 200
Breakpoint 8 at 0x4018ad: file thriftshop.c, line 200.
(gdb) break 236
Breakpoint 9 at 0x4019d7: file thriftshop.c, line 236.
(gdb) break 271
Breakpoint 10 at 0x401b1e: file thriftshop.c, line 271.
(gdb) break 391
No line 391 in the current file.
Make breakpoint pending on future shared library load? (y or [n]) n
(gdb) break 319
Breakpoint 11 at 0x401cf1: file thriftshop.c, line 319.
(gdb) break 347
Breakpoint 12 at 0x401d9e: file thriftshop.c, line 347.
(gdb) break 351
Breakpoint 13 at 0x401de3: file thriftshop.c, line 351.
(gdb) break 359
Breakpoint 14 at 0x401e53: file thriftshop.c, line 359.
(gdb) break 3361
No line 3361 in the current file.
Make breakpoint pending on future shared library load? (y or [n]) n
(gdb) break 361
Breakpoint 15 at 0x401e83: file thriftshop.c, line 361.
(gdb) break 370
Breakpoint 16 at 0x401ed7: file thriftshop.c, line 370.
(gdb) break 183
```

. . . The End. . .