PROBLEM SOLVING WITH C-UE19CS151

NAME-SATYAM RUDRADUTTA DWIVEDI

SRN-PES2UG19CS370

PRN-PES2201900061

SECTION H

**PROBLEM STATEMENT- Develop a mini employee leave management system; You need to have the employee details in a file, allow the employee to log in using his employee ID (which is unique); allow employees to avail a certain number of leaves per year (which is split into x casual leaves, y medical leaves and z earned leaves); if the employee is well within the allowed limits, automatically approve the leave; else reject and provide the reason.**

I have coded the program using a highly modular approach and I have tried my best to split the code into individual independent sections and by doing this I achieved a seamless integration between the different sections and code works flawlessly

I have used the concept of POPULATING **A STRUCTURE BY READING A FILE** and I have tried this Idea entirely by my own thinking and performed an extensive research on the topic.

The code has

**void main2();**//this containsthe main part of the code workings,its purp-ose is explained below just before the function

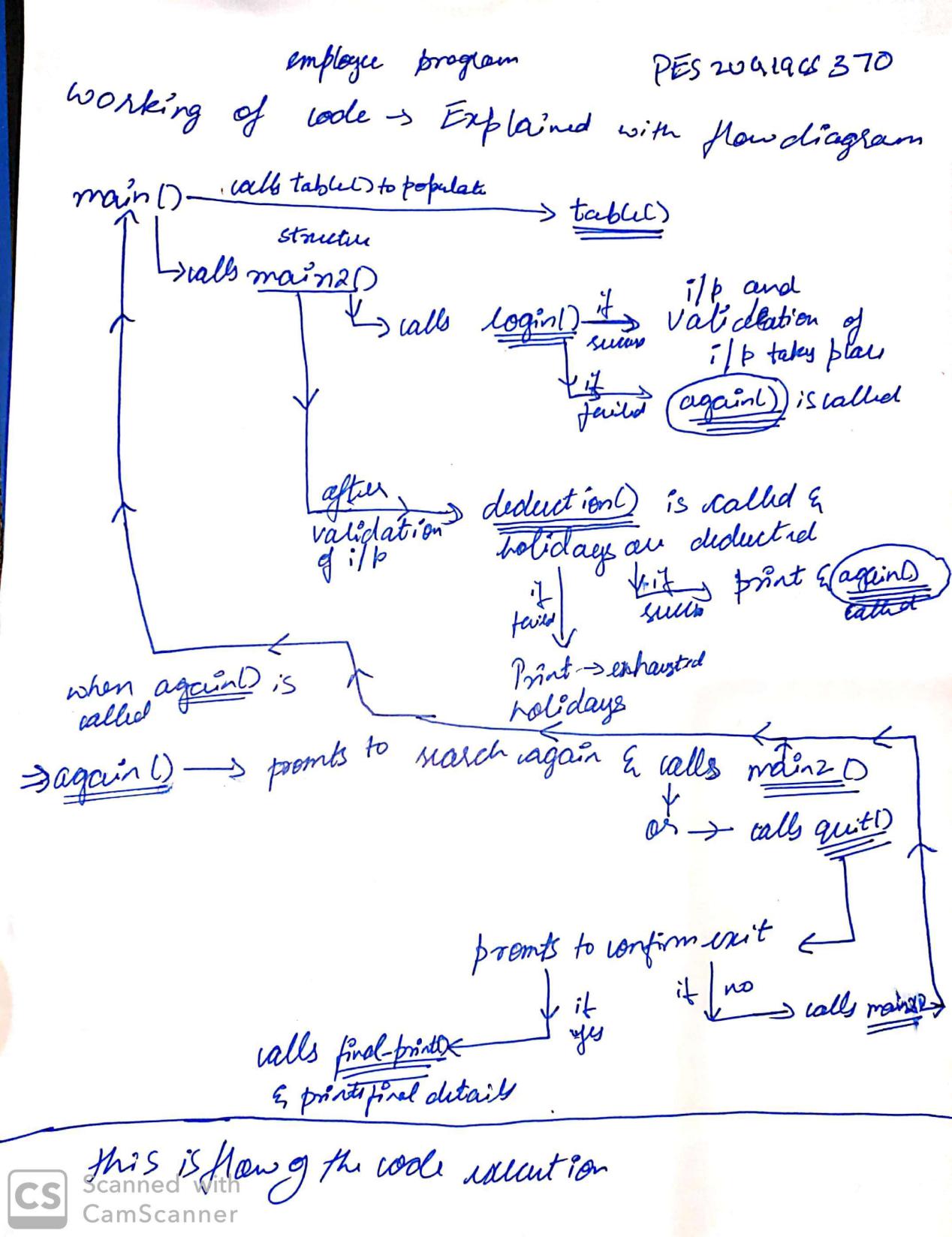
**void final\_print();**//prints the details of the user on quitting the wizard

**void quit()**;//this is one function which handles the termination of code prompting the user to take another trail

**void again();**//this function handles the login if user wishes to take another leave

**void table(char file\_n[]);**//populating the structure from data elements from the file by using specific delimiter

**int login();**//this function verifies the password with the one present in the file database



I have taken the **name of file from which the data** is read from **COMMAND LINE ARGUMENT**

The code and its working is given below

In comments

**FEW IMPORTANT POINTS REGARDING THE GUIDELINES**

1)There are 2 validations happening in the code

2)I have minimized the global variable and kept it to 2 variable-structure and linematched

3)I have minimized use of variables which have vague names

4)I have taken only one input through command line arguement as it remains constant throughout the code and other inputs change based on the condition of the code and cannot be taken from command line

5)As I have kept the crucial data as global so I don’t pass argument as doing this would cause access problems in the functions and overwriting issues which happens unknowingly

6)main function has minimal involvement in the execution process

7)There are no errors or any warnings shown while execution in Windows subsystem for Linux(WSL),as shown in the screenshots

8)makefile utility is tested and used in my code and all the comments describe the functionality of the code

**I HAVE SPLIT THE CODE INTO 4 .C AND 1.H FILE AND THERE IS ONE .MK FILE AND .TXT FILE FROM WHICH DATA IS READ**

**THE 4 .C FILES ARE AS FOLLOWS**

1. **main\_function.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

#include"server.h"

struct Empdetials emp[5];  // definition of emp

int linematched;

void main(int argc,char \*argv[])//command line arguments takes the name of the file to be read

{

    char file\_name[30];//to copy the name of the file into a character array

    strcpy(file\_name,argv[1]);//copying the name here

    //printf("%s\n\n",file\_name);//tersting if the name is taken sucessfully

    table(file\_name);//calling the function to initiate the process of filling a structure with all the data from the files

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

    printf("----------------------WELCOME TO LEAVE MANEGMENT SYSTEM--------------\n");

    main2();//the use of this is explained in the next .c file

}

1. **main2\_table.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

#include"server.h"

/\*main function has a call to table() and invoking main() for re-login used to reset the structure and the deductions were lost,so I copied

everything of the main function into another function called main2() which is basically the amin function without he table() invoking so the

structure is not resetted after every call from again() and quit() during Re-login and structure keeps a track of the deductions that happened and

in this way I also removed the Welcome sentence to the same user,but he main reason is to avoid rewriting of the structure on every call

and losing the deduction data\*/

void main2()

{

    int casualm=10;//defining the maximum possible leaves here

    int medicalm=15;

    int earnedm=7;

    int leave\_days=0;//handles the number of leaves that a user will enter

    int choice=0;//type of leave is handled by this variable 1 is casual,2 is emergency,3 is earned

    int log=0;//CHECKS IF LOGIN WAS SUCCESFULL OR NOT

    int res=0;//CHECKS IF DEDUCTION WAS SUCCESFULL OR NOT

    //-------------------------------------------------------------------------------------------------------------------------------------------------

    //--------------------------------------------------------------------------------------------------------------------------------------------------

    log=login();//login is invoked and log stores the value which determines wheter the ogin was succesful or not

    if(log==1)//if log=1,implies that the login was succesful and the user was found

    {

        printf("PLEASE SELECT HE TYPE OF LEAVE YOU WANT TO TAKE\n");

        printf("1.CASUAL LEAVE\n");

        printf("2.MEDICAL LEAVE\n");

        printf("3.EARNED LEAVE\n");

        scanf("%d",&choice);

        printf("PLEASE ENTER THE NUMBER OF LEAVES YOU WANT TO APPLY FOR\n");

        scanf("%d",&leave\_days);

    }

    else

    {

        printf("SORRY THE LOGIN CREDENTIALS DON'T MATCH WITH ANY USER\n\n\n");//when the login fails this message is displayed

        again();//gives user another chance to take give the input

    }

    //first validation of the input values,this ompare the input with the largest value of holidays available

    int flag=1;//this controls wheter the variable entetred is within the maximum limit and that decuction() function can be called

     if(choice==1)

    {

        if(leave\_days>casualm)

        {

            printf("THE ENTERED NUMBER OF DAYS CROSSES THE MAXIMUM LIMIT \n");

            again();

            flag=0;

        }

    }

    if(choice==2)

    {

        if(leave\_days>medicalm)

        {

            printf("THE ENTERED NUMBER OF DAYS CROSSES THE MAXIMUM LIMIT \n");

            again();

            flag=0;

        }

    }

    if(choice==3)

    {

        if(leave\_days>earnedm)

        {

            printf("THE ENTERED NUMBER OF DAYS CROSSES THE MAXIMUM LIMIT \n");

            again();

            flag=0;

        }

    }

    if(choice>3||choice<0)

    {

        printf("sorry!! wrong input\n");

        again();

        flag=0;

    }

    if(flag)//if the validation is succcesful the value of the variable remains 1 and this loop is entered

    {

        res=deduction(linematched,leave\_days,choice);//if res==1 then thern the validation was succesful and then the deduction is called and parameter is passed

        if(res==1)//if deduction was succesful then the

        {

            printf("DEDUCTION  WAS SUCCESFUL\n\n");

            again();

        }

    }

}

//I am trying to convert the file to a structure and then use the structure to access the data and I'm also using files to store the information

extern void table(char file\_n[])//this function is responsible for populating the structure by reading the file

{

   FILE \*fp;//definig a file pointer

   char ch;

   char  line[256];//definig 256 here,any arbitary number can be used

   char \*token;//token pointer that points to the elements of a line

   int tokenposition=0;//token here means the particular text separated by comma

   int lineposition=0;//line here is the entire line which has entire details of the an emmployee

   fp=fopen(file\_n,"r");//opening the file in read mode as we are only accessing information and not rewrting it

   //-------------------------------------------------------------------------------------------------------------------------------------------------

   while(fgets(line,256,fp) !=NULL) //accessing the lines one by one and trying to recognize the end of a line

   {

       tokenposition=0;//when we tokenize the line we need the count to keep a track of where we are int he string

      //printf("%s \n",line);//used to check which line is not being read by the compiler

       token = strtok(line,",");

        //this seperates a line into token(entity) based on a delimiter,here the delimiter is a comma

       //strtok-tring tokenization method

       while(token != NULL)

       {

           switch(tokenposition)//acessing the required element through the tokenposition counter

           {

               case 0://NAME

                   strcpy(emp[lineposition].name,token);

                   break;

               case 1://ID

                   strcpy(emp[lineposition].id,token);

                   break;

               case 2://PASSWORD

                   strcpy(emp[lineposition].password,token);

                   break;

               case 3://CASUAL LEAVE

                   emp[lineposition].casual=atoi(token);

                   break;

               case 4://MEDICAL LEAVE

                   emp[lineposition].medical=atoi(token);

                   break;

               case 5://EARNED LEAVE

                   emp[lineposition].earned=atoi(token);

                   break;

           }

           token =strtok(NULL,",");

       tokenposition++;//to access the next token in the same line

       }

       lineposition++;//after all the tokens are put in a structure,wwe move to the next line and redo the entire process of reading a line

   }

}

1. **login\_deduction.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

#include"server.h"

int login()

{

    int i;//for the loop

    char userid[9];//user ID int he majn function,this will be passed to the the login function

    char userpassword[5];//user password in the main function

    printf("ENTER YOUR 8 CHARACTER USER ID\n");

    scanf("%s",userid);

    printf("ENTER THE 4 CHARACTER PASSWORD\n");

    scanf("%s",userpassword);

    linematched=0;

    for(i=0;i<5;i++)

    {

        if((strcmp(emp[i].id,userid)==0)&&(strcmp(emp[i].password,userpassword)==0))//checking if login credentils are corre3ct or wrong

        {

            linematched=i;

            printf("WELCOME %s TO THE LEAVE MANEGMENT SYSTEM\n",emp[linematched].name);

            return 1;//TO INDICATE THAT USER WAS FOUND AND LOGIN WAS SUCCESFUL

            break;

        }

    }

}

int deduction(int linematched,int days,int type\_of\_leave)//here all inputs are validated for the second time after the validation in main function

{

    switch(type\_of\_leave)

    {

        case 1://casual leave

        {

            if(days>emp[linematched].casual)//second validation happens here with respect to the specified user

            {

                printf("SORRY YOU HAVE EXHAUSTED YOUR LEAVE QUOTA\n\n");

                again();

                return 0;

                break;

            }

            else

            {

                emp[linematched].casual=emp[linematched].casual-days;

                return 1;

                break;

            }

            break;

        }

        case 2://medical leave

        {

            if(days>emp[linematched].medical)

            {

                printf("SORRY YOU HAVE EXHAUSTED YOUR LEAVE QUOTA\n\n");

                again();

                return 0;

                break;

            }

            else

            {

                emp[linematched].medical=emp[linematched].medical-days;

                return 1;

                break;

            }

            break;

        }

        case 3://earned leave

        {

            if(days>emp[linematched].earned)

            {

                printf("SORRY YOU HAVE EXHAUSTED YOUR LEAVE QUOTA\n\n");

                again();

                return 0;

                break;

            }

            else

            {

                emp[linematched].earned=emp[linematched].earned-days;

                return 1;

                break;

            }

            break;

        };

    };

}

1. **quit\_final\_final.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

#include"server.h"

void quit()

{

    char ans[5];

    printf("ARE YOU SURE YOU WANT TO QUIT THE WIZARD? Y OR N\n");

    scanf("%s", ans);

    if (tolower(ans[0]) == 'y')

    {

        final\_print();

        printf("\n\n");

        printf("---THANK YOU FOR USING OUR PORTAL--\n");

    }

    else

    {

        main2();

    }

}

void again()

{

    char ans1[5];

    printf("SEARCH AGAIN USING ID AND PASSWORD? Y OR N\n");

    scanf("%s", ans1);

    if (tolower(ans1[0]) == 'y')

    {

        main2();//FOR RE-LOGIN PROCESS

    }

    else

    {

        quit();

    }

    printf("\n\n\n\n\n\n");

}

void final\_print()

{

    printf("----------------FINAL STATEMENT OF LEAVE OF EACH EMPLOYEE------------------------\n\n");

    int i;

    for (i = 0; i < 5; i++)

    {

        printf("NAME : %s\n",emp[i].name);

        printf("USER ID : %s\n",emp[i].id);

        printf("CASUAL LEAVE LEFT : %d\n",emp[i].casual);

        printf("MEDICAL LEAVE LEFT : %d\n",emp[i].medical);

        printf("EARNED LEAVE LEFT : %d\n\n",emp[i].earned);

    }

    printf("---------THANK YOU FOR USING LEAVE MANEGMENT PORTAL-----\n");

}

**.H FILES**

1. **server.h**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

extern int linematched;//global variable which stores the variable

extern struct Empdetials

{

    char name[25];

    char id[9];//onr extra character to store /n

    char password[5];

    int casual;//casual,medical,earned leave as per the question

    int medical;

    int earned;

}emp[5];//declaring array of structures which will later be populated using table() function

//extern struct Empdetials emp[5]; // declaration of emp

extern void main2();//this containsthe main part of the code workings,its pupose is explained below just before the function

extern void final\_print();//prints the details of the user on quitting the wizard

extern void quit();//this is one function which handles the termination of code prompting the user to take another trail

extern void again();//this function handles the login if user wishes to take another leave

extern void table(char file\_n[]);//populating the structure from data elements from the file by using specific delimiter

extern int login();//this function verifies the password with the one present in the file database

extern int deduction(int linematched,int days,int type\_of\_leave);//this performs another validation and deducts the days user desires for taking the leave

**MAKEFILE**

**makefile.mk**

a.out: main\_function.o quit\_again\_final.o login\_deduction.o main2\_table.o

    gcc main\_function.o quit\_again\_final.o login\_deduction.o main2\_table.o

quit\_again\_final.o: quit\_again\_final.c server.h

    gcc -c quit\_again\_final.c

login\_deduction.o: login\_deduction.c server.h

    gcc -c login\_deduction.c

main2\_table.o: main2\_table.c server.h

    gcc -c main2\_table.c

main\_function.o: main\_function.c server.h

    gcc -c main\_function.c

**employeeinfo.txt**

**Ramesh,QW120345,PO56,10,15,7**

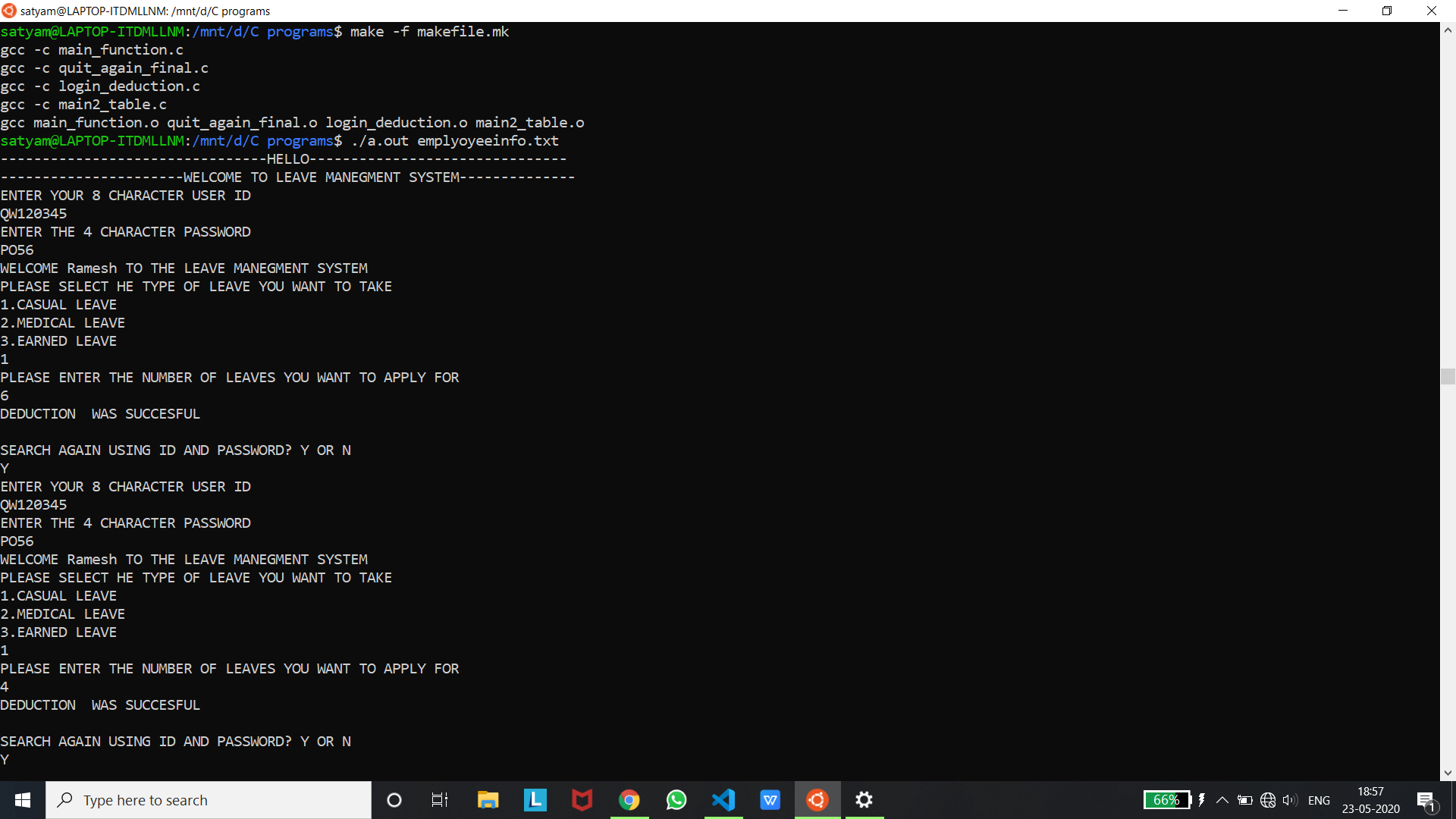
**Rajesh,QW120905,IO56,10,15,7**

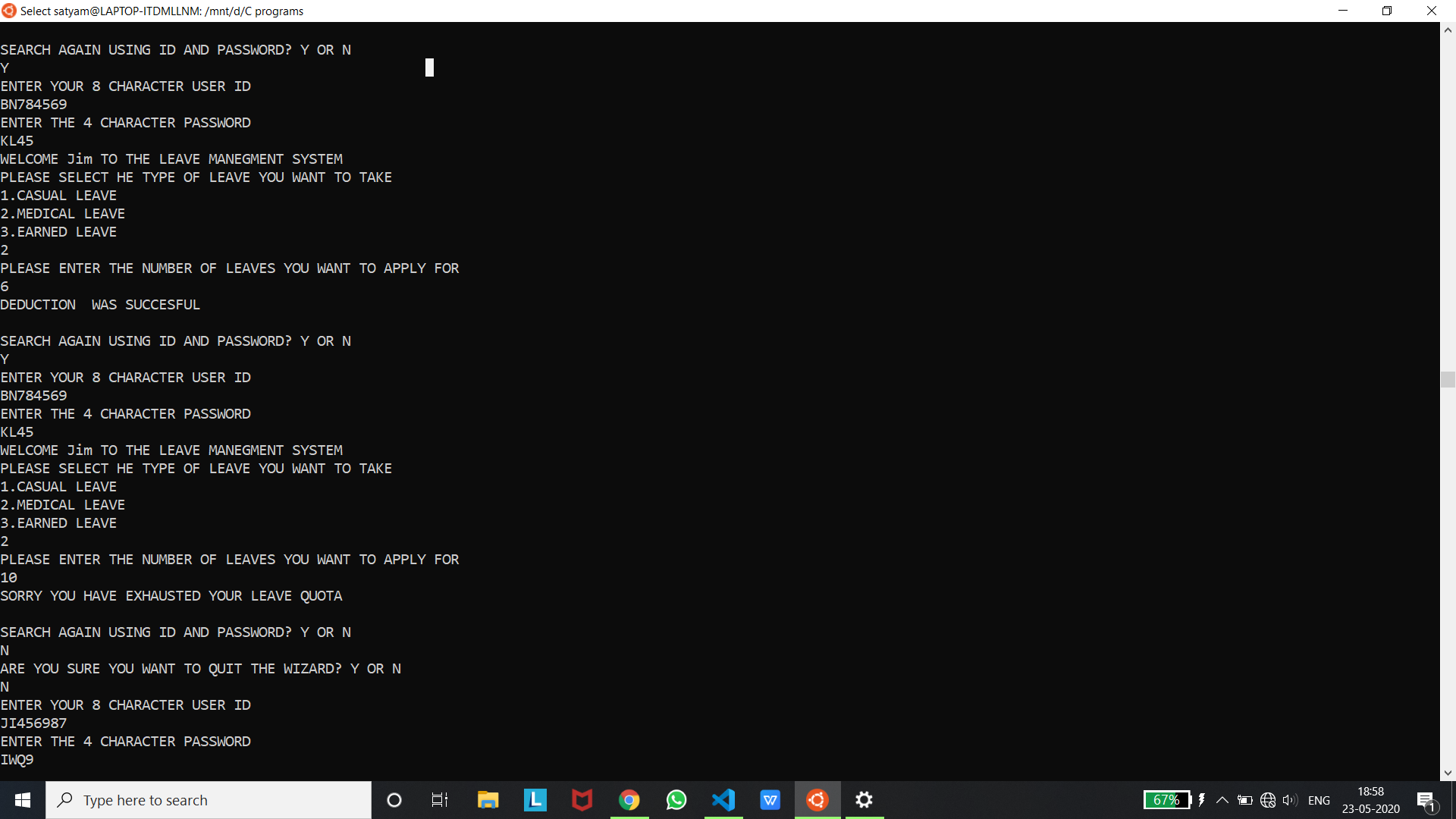
**Kajal,JI456987,IWQ9,10,15,7**

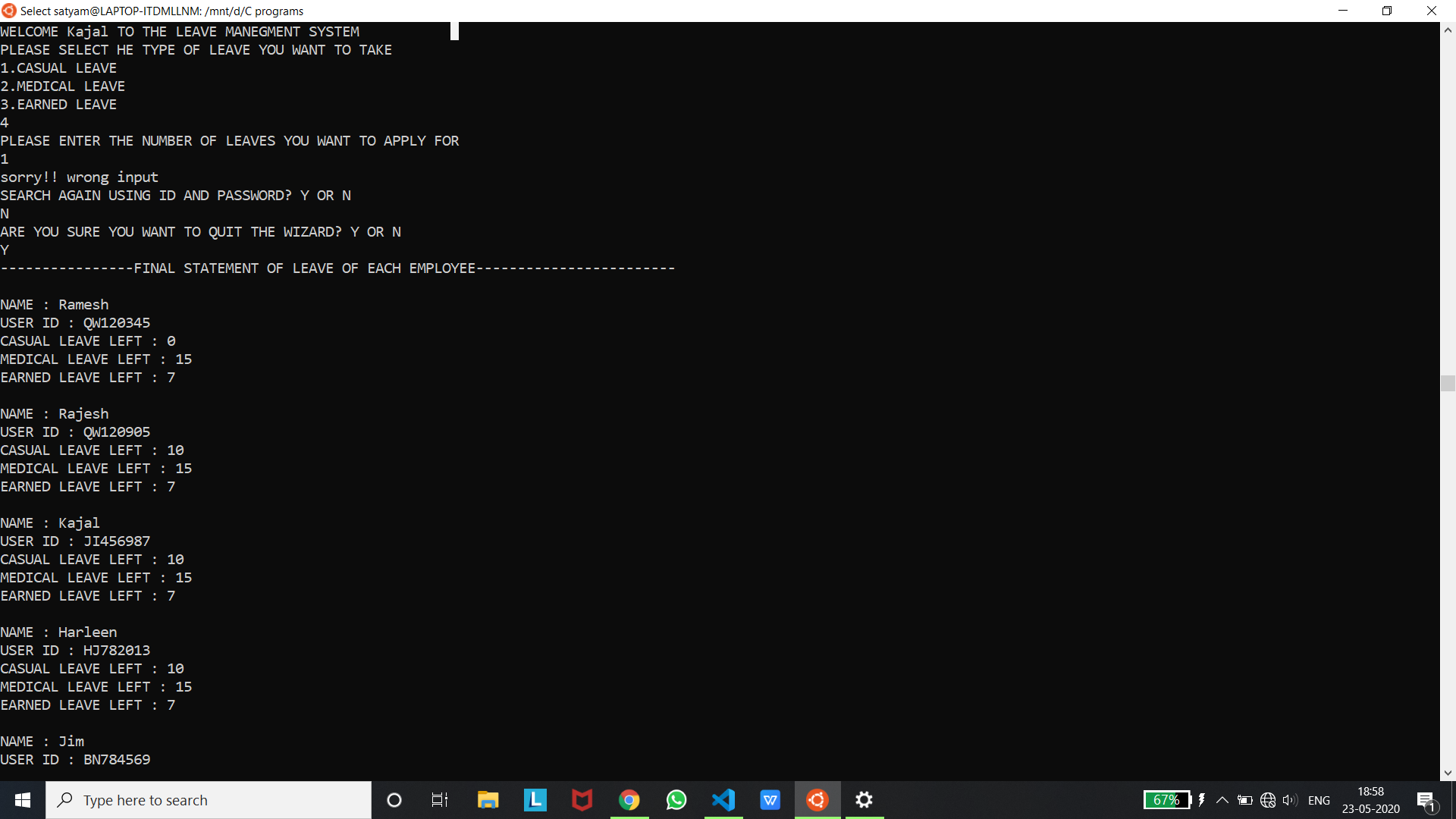
**Harleen,HJ782013,ZM12,10,15,7**

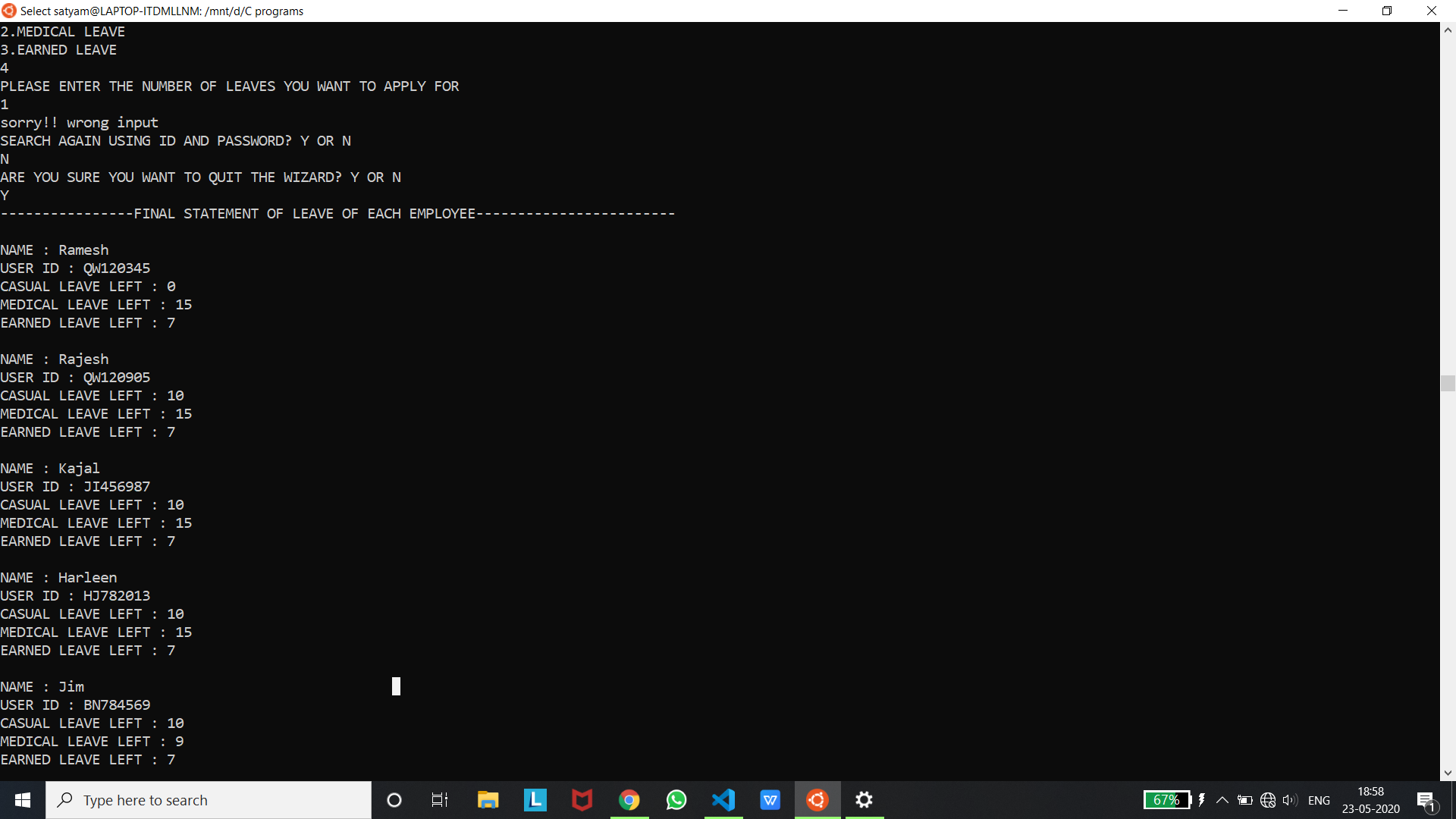
**Jim,BN784569,KL45,10,15,7**

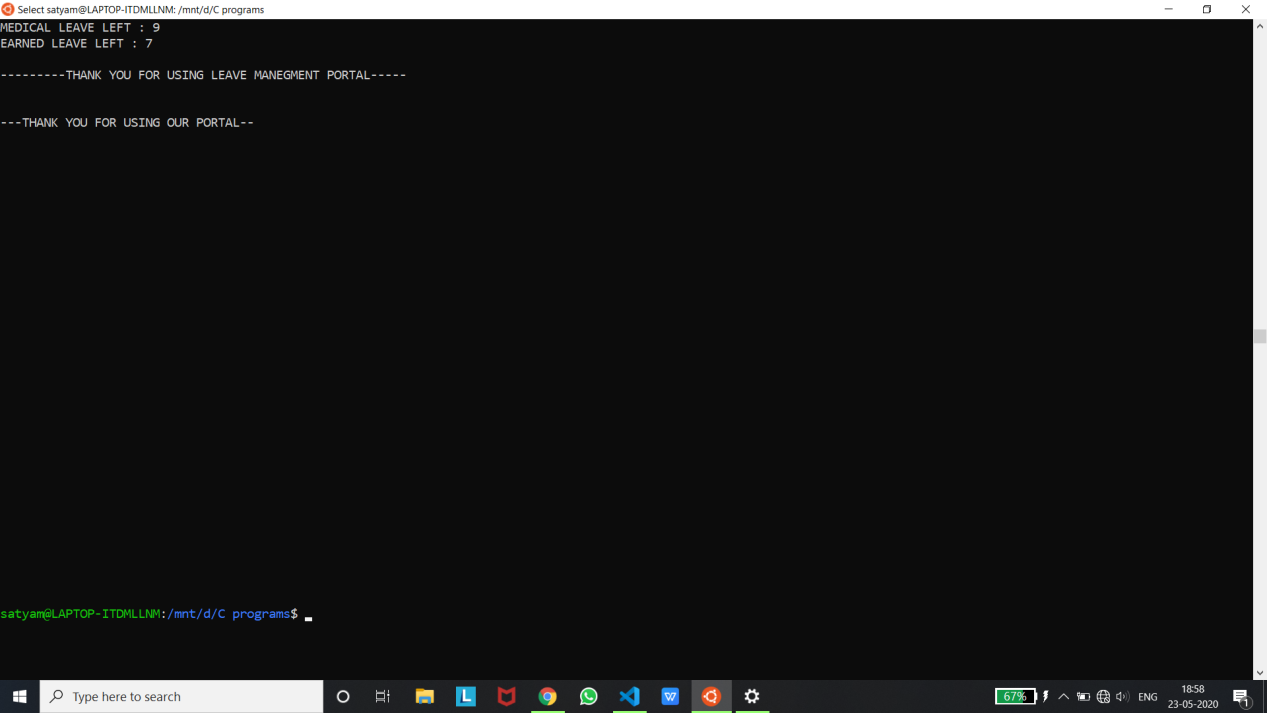
**OUTPUT SCREENSHOTS**

****

****

****

****

****