**SAKSHI PANDEY**

**MCA I SEMESTER**

**Y23271024**

**Project title:** **ALUMNI FILE SYSTEM**

**Overview:** This project is a C program that creates and manages a file system for storing and manipulating the information of the alumni of a college or university.

**Objective:** The objective of this project is to demonstrate the use of structures, file handling, functions, strings and pointers in C programming, and to apply them to a real-world problem of managing alumni data.

**Project justification:** This project is justified by the need to maintain and update the records of the alumni of a college or university, and to provide easy and efficient access to their data. The project helps me to improve my skills and knowledge in C programming.

**Program Description**: This program is designed to manage the information of the alumni of a college or university. It uses a structure named alumni to store the details of each alumni, such as roll number, name, batch year, branch, email, and phone number. The program can perform the following functions:

* **add\_entry()**: This function allows the user to add a new entry to the file system, by entering the data of an alumni and appending it to the end of the file.
* **delete\_entry()**: This function allows the user to delete an existing entry from the file system, by entering the roll number of an alumni and removing it from the file.
* **search\_entry():** This function allows the user to search for an entry in the file system, by entering the roll number, name, batch year, or branch of an alumni and displaying the matching records.
* **modify\_entry():** This function allows the user to modify an existing entry in the file system, by entering the roll number of an alumni and updating the data of the corresponding record.
* **display\_entry()**: This function allows the user to display all the entries in the file system, by printing the data of each alumni in a tabular format.
* **group\_search()**: This function allows user to display all entries with given batch year and branch in the file system, by printing the data of each alumni in a tabular format.

The program uses a binary file to store the data of the alumni, and uses the roll number as the primary key to identify each record. The program also handles the possible errors and exceptions that may occur during the execution, such as invalid input, file not found, record not found, etc. The program is written in C language and uses the standard input/output and file handling libraries.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <stdbool.h>

#include <ctype.h>

struct alumni

{

char roll\_number[25];

char name[30];

char batch\_year[5];

char branch[5];

char email[30];

char phone\_number[25];

};

FILE \*fptr;

bool is\_valid\_roll\_number (char \*roll\_number)

{

for (int i = 0; i < strlen (roll\_number); i++)

{

if (isalnum (roll\_number[i]) == 0)

{

return false;

}

}

return true;

}

bool is\_valid\_name\_branch (char \*name\_branch)

{

for (int i = 0; i < strlen (name\_branch); i++)

{

if (isalpha (name\_branch[i]) == 0 && name\_branch[i] != ' ')

{

return false;

}

}

return true;

}

bool is\_valid\_phone\_batch (char \*phone\_batch)

{

for (int i = 0; i < strlen (phone\_batch); i++)

{

if (isdigit (phone\_batch[i]) == 0)

{

return false;

}

}

return true;

}

bool is\_valid\_email (char \*email)

{

int i, len = strlen(email);

int at\_count = 0, at\_index = -1, dot\_count = 0, dot\_index = -1;

for (i = 0; i < len; i++)

{

if (email[i] == '@')

{

at\_count++;

at\_index = i;

}

else if (email[i] == '.')

{

dot\_count++;

dot\_index = i;

}

}

if (at\_count != 1)

return false;

if (dot\_count < 1 || dot\_index < at\_index)

return false;

if (dot\_index == len - 1)

return false;

if (!isalpha(email[0]))

return false;

for (i = 0; i < len; i++)

{

if (!isalnum(email[i]) && email[i] != '@' && email[i] != '.' && email[i] != '\_' && email[i] != '-')

return false;

}

return true;

}

void add\_entry()

{

struct alumni temp;

struct alumni \*ptr = &temp;

fptr = fopen("alumni.txt", "a+");

if (fptr == NULL)

{

printf("Error opening file.\n");

exit(1);

}

char choice;

do

{

printf("Enter roll number: ");

scanf("%s", ptr->roll\_number);

if (!is\_valid\_roll\_number (ptr->roll\_number))

{

printf("Invalid roll number. Please enter a valid alphanumeric string.\n");

continue;

}

int found = 0;

struct alumni record;

rewind(fptr);

while (fread(&record, sizeof(struct alumni), 1, fptr) == 1)

{

if (strcmp(record.roll\_number, ptr->roll\_number) == 0)

{

found = 1;

break;

}

}

if (found == 1)

{

printf("The roll number already exists. Please enter a different one.\n");

continue;

}

printf("Enter name: ");

scanf(" %[^\n]s", ptr->name);

if (!is\_valid\_name\_branch (ptr->name))

{

printf("Invalid name. Please enter a valid alphabetical string.\n");

continue;

}

printf("Enter batch year: ");

scanf("%s", ptr->batch\_year);

if (!is\_valid\_phone\_batch (ptr->batch\_year))

{

printf("Invalid batch year. Please enter a valid numeric string.\n");

continue;

}

printf("Enter branch: ");

scanf("%s", ptr->branch);

if (!is\_valid\_name\_branch (ptr->branch))

{

printf("Invalid branch. Please enter a valid alphabetical string.\n");

continue;

}

printf("Enter email: ");

scanf("%s", ptr->email);

if (!is\_valid\_email (ptr->email))

{

printf("Invalid email. Please enter a valid email string.\n");

continue;

}

printf("Enter phone number: ");

scanf("%s", ptr->phone\_number);

if (!is\_valid\_phone\_batch (ptr->phone\_number))

{

printf("Invalid phone number. Please enter a valid numeric string.\n");

continue;

}

fwrite(ptr, sizeof(struct alumni), 1, fptr);

printf("Entry added successfully.\n");

printf("Do you want to add another entry? (y/n): ");

scanf(" %c", &choice);

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

fclose(fptr);

}

void delete\_entry()

{

struct alumni temp;

struct alumni \*ptr = &temp;

int found = 0;

char roll[25];

printf("Enter roll number of the student to delete: ");

scanf("%s", roll);

fptr = fopen("alumni.txt", "r");

if (fptr == NULL)

{

printf("Error opening file.\n");

exit(1);

}

FILE \*ftemp;

ftemp = fopen("temp.txt", "w");

if (ftemp == NULL)

{

printf("Error creating temporary file.\n");

exit(1);

}

while (fread(ptr, sizeof(struct alumni), 1, fptr))

{

if (strcmp(ptr->roll\_number,roll)==0)

{

found = 1;

}

else

{

fwrite(ptr, sizeof(struct alumni), 1, ftemp);

}

}

fclose(fptr);

fclose(ftemp);

if (found)

{

remove("alumni.txt");

rename("temp.txt", "alumni.txt");

printf("Entry deleted successfully.\n");

}

else

{

remove("temp.txt");

printf("Entry not found.\n");

}

}

void search\_entry()

{

struct alumni temp;

struct alumni \*ptr = &temp;

int found = 0;

char roll[25];

printf("Enter roll number of the student to search: ");

scanf("%s", roll);

fptr = fopen("alumni.txt", "r");

if (fptr == NULL)

{

printf("Error opening file.\n");

exit(1);

}

while (fread(ptr, sizeof(struct alumni), 1, fptr))

{

if (strcmp(ptr->roll\_number,roll)==0)

{

found = 1;

printf("Entry found.\n");

printf("Roll number: %s\n", ptr->roll\_number);

printf("Student Name: %s\n", ptr->name);

printf("Batch Year: %s\n", ptr->batch\_year);

printf("Branch: %s\n", ptr->branch);

printf("Email: %s\n", ptr->email);

printf("Phone Number: %s\n", ptr->phone\_number);

break;

}

}

fclose(fptr);

if (!found)

{

printf("Entry not found.\n");

}

}

void modify\_entry()

{

struct alumni temp;

struct alumni \*ptr = &temp;

int found = 0,ch;

char roll[25];

printf("Enter roll number of the student to modify: ");

scanf("%s", roll);

fptr = fopen("alumni.txt", "r+");

if (fptr == NULL)

{

printf("Error opening file.\n");

exit(1);

}

while (fread(ptr, sizeof(struct alumni), 1, fptr))

{

if (strcmp(ptr->roll\_number,roll)==0){

found = 1;

printf("1.Name\t2.Email\t3.Phone Number\n");

printf("Enter choice you want to modify:\n");

scanf("%d",&ch);

switch(ch){

case 1:

printf("Enter name: ");

scanf(" %[^\n]s", ptr->name);

if (!is\_valid\_name\_branch (ptr->branch))

{

printf("Invalid branch. Please enter a valid alphabetical string.\n");

continue;

}

break;

case 2:

printf("Enter new email: ");

scanf("%s", ptr->email);

if (!is\_valid\_email (ptr->email))

{

printf("Invalid email. Please enter a valid email string.\n");

continue;

}

break;

case 3:

printf("Enter new phone number: ");

scanf("%s", ptr->phone\_number);

if (!is\_valid\_phone\_batch (ptr->phone\_number))

{

printf("Invalid phone number. Please enter a valid numeric string.\n");

continue;

}

break;

}

fseek(fptr, -sizeof(struct alumni), SEEK\_CUR);

fwrite(ptr, sizeof(struct alumni), 1, fptr);

printf("Entry modified successfully.\n");

break;

}

}

fclose(fptr);

if (!found)

{

printf("Entry not found\n");

}

}

void display\_all()

{

struct alumni temp;

struct alumni \*ptr = &temp;

int found=0;

fptr = fopen("alumni.txt", "r");

if (fptr == NULL)

{

printf("Error opening file.\n");

exit(1);

}

printf("Roll Number\tStudent Name\tBatch\tBranch\tEmail Id\t\t\tPhone Number\n");

while (fread(ptr, sizeof(struct alumni), 1, fptr))

{

found++;

printf("%s\t%s\t%s\t%s\t%s\t%s\n", ptr->roll\_number,ptr->name,ptr->batch\_year,ptr->branch,ptr->email,ptr->phone\_number);

}

fclose(fptr);

if(!found)

{

printf("Entry not found\n");

}

}

void group\_search()

{

struct alumni temp;

struct alumni \*ptr = &temp;

int found=0;

char branch[5];

char batch[5];

printf("Enter batch year to search: ");

scanf("%s", batch);

printf("Enter branch to search: ");

scanf("%s", branch);

fptr = fopen("alumni.txt", "r");

if (fptr == NULL)

{

printf("Error opening file.\n");

exit(1);

}

printf("Roll Number\tStudent Name\tBatch\tBranch\tEmail\t\t\tPhone Number\n");

while (fread(ptr, sizeof(struct alumni), 1, fptr))

{

if ((strcmp(ptr->batch\_year,batch)==0)&&(strcmp(ptr->branch,branch)==0))

{

found++;

printf("%s\t%s\t%s\t%s\t%s\t%s\n", ptr->roll\_number,ptr->name,ptr->batch\_year,ptr->branch,ptr->email,ptr->phone\_number);

}

}

fclose(fptr);

if(!found)

{

printf("Entry not found\n");

}

}

int menu()

{

int choice;

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

printf("Alumni Database Menu\n");

printf("1. Add entry\n");

printf("2. Delete entry\n");

printf("3. Search entry\n");

printf("4. Modify entry\n");

printf("5. Display all entries\n");

printf("6. Search by batch and branch\n");

printf("7. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

return choice;

}

int main()

{

printf("Sakshi Pandey\tY23271024\n");

printf("ALUMNI FILE SYSTEM\n");

int choice;

do {

choice = menu();

switch (choice)

{

case 1:

add\_entry();

break;

case 2:

delete\_entry();

break;

case 3:

search\_entry();

break;

case 4:

modify\_entry();

break;

case 5:

display\_all();

break;

case 6:

group\_search();

break;

case 7:

printf("Exiting the program\n");

break;

default:

printf("Invalid choice\n");

break;

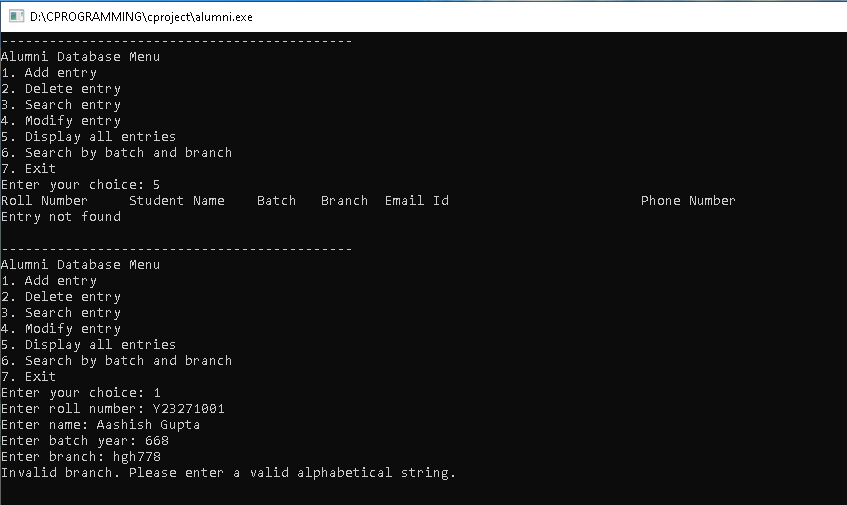
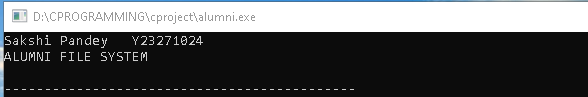
}

} while (choice != 7);

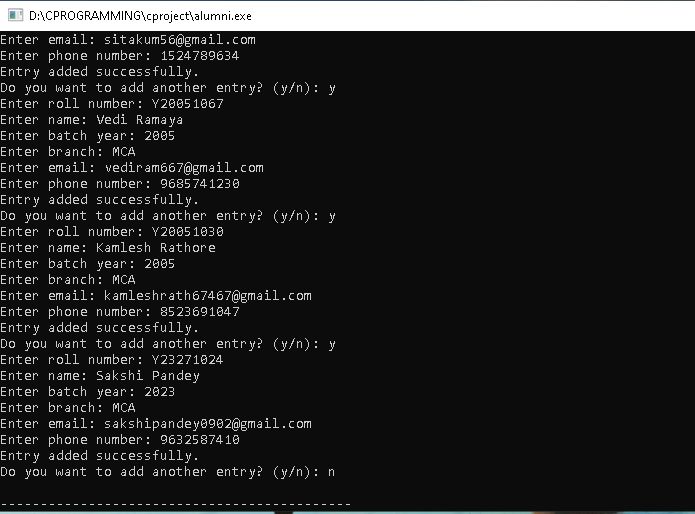
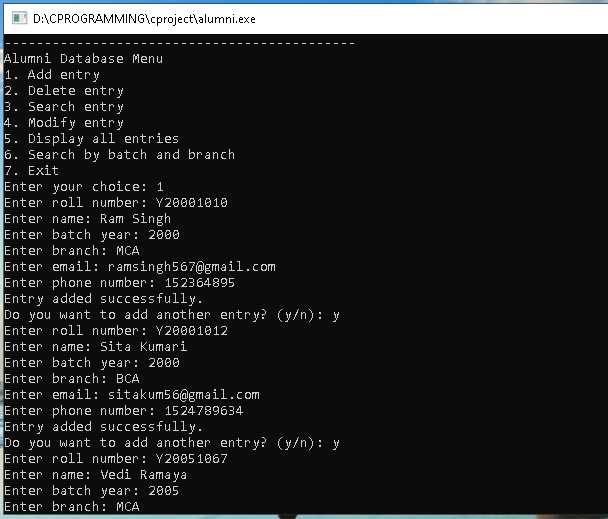
return 0;

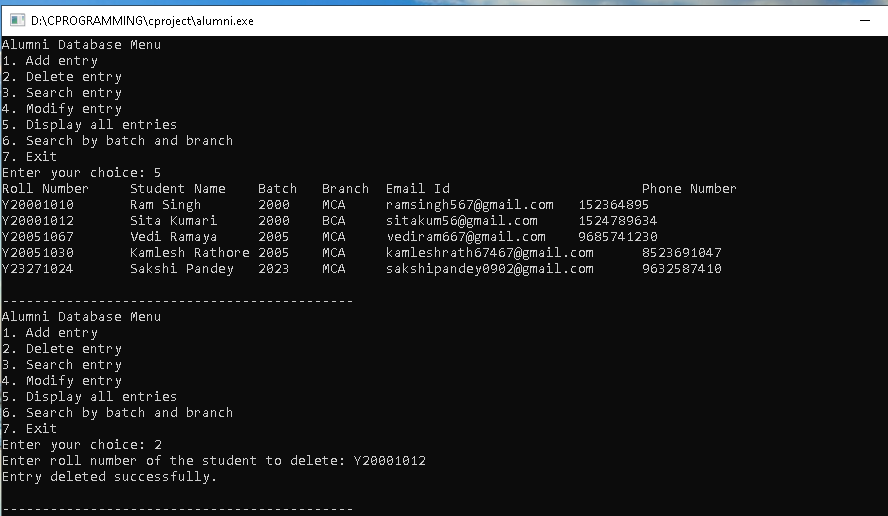
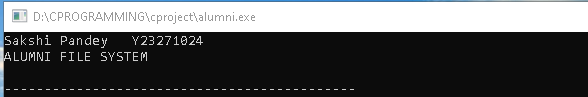
}

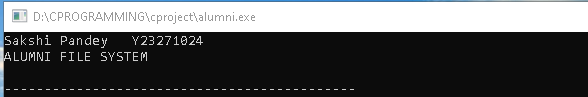
**OUTPUT**

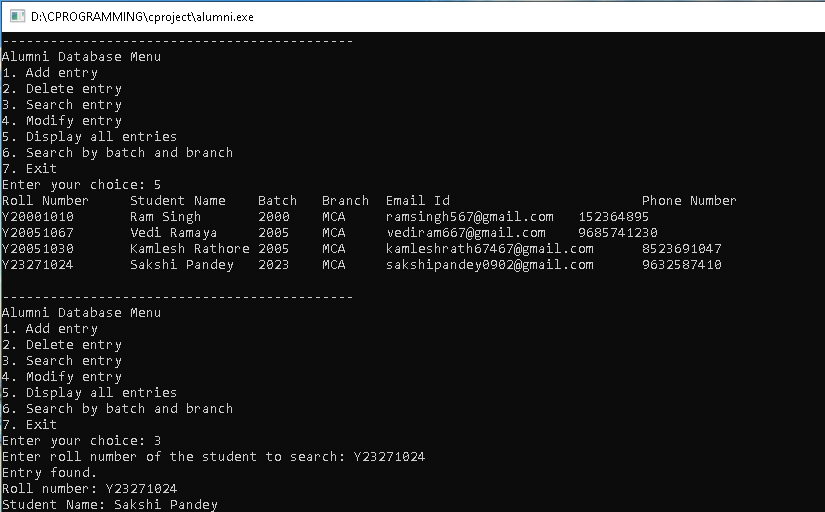
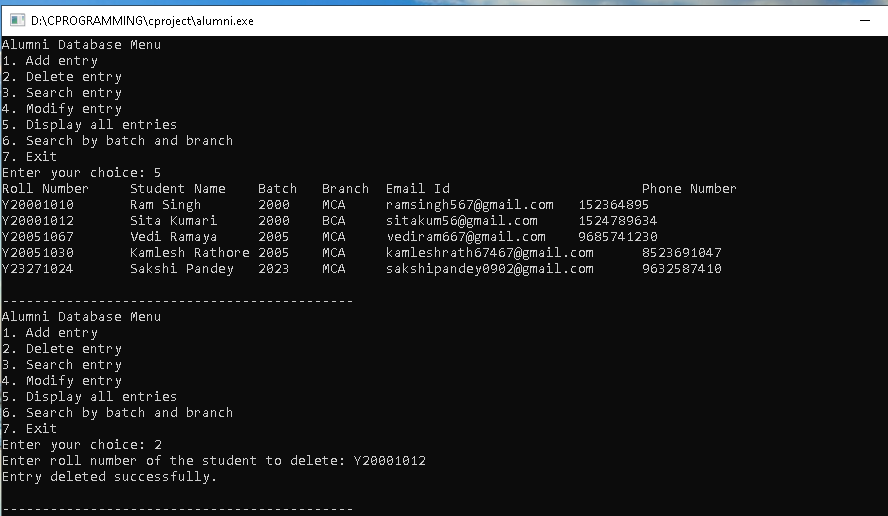


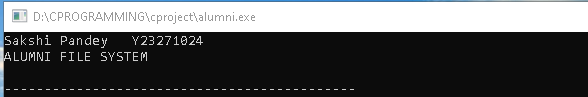
**ADD ENTRY**

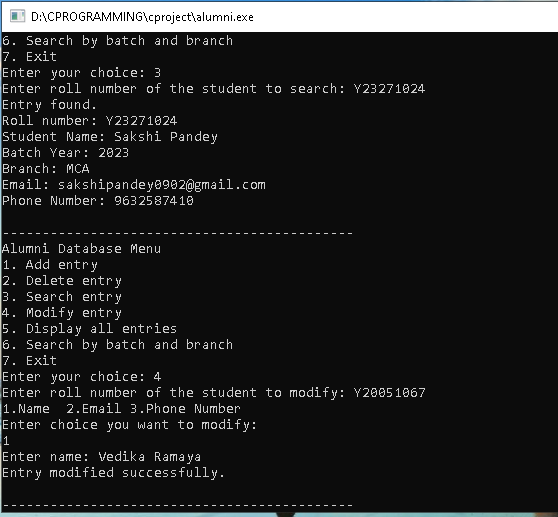
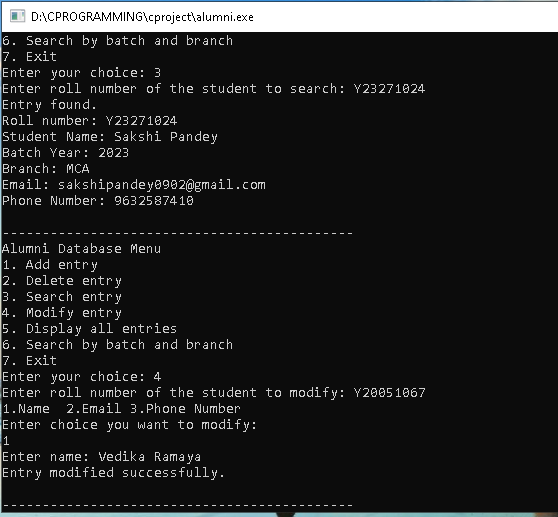


**DISPLAY ENTRY**

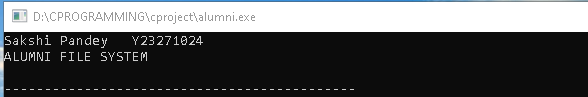
**DELETE ENTRY**

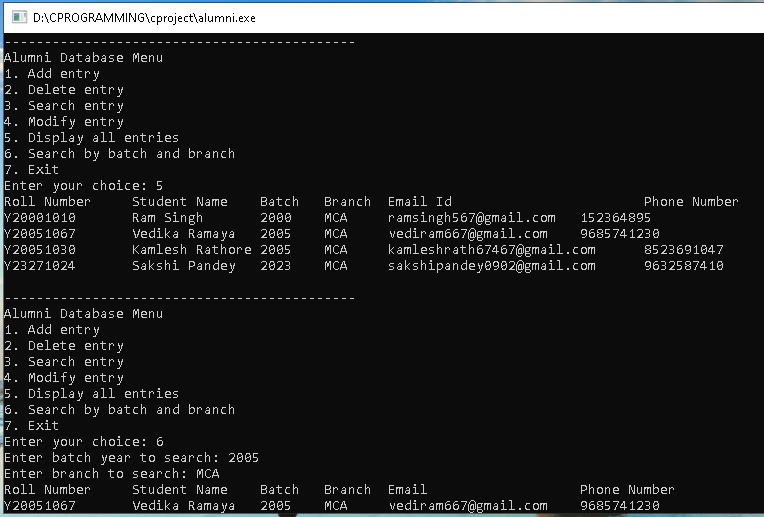
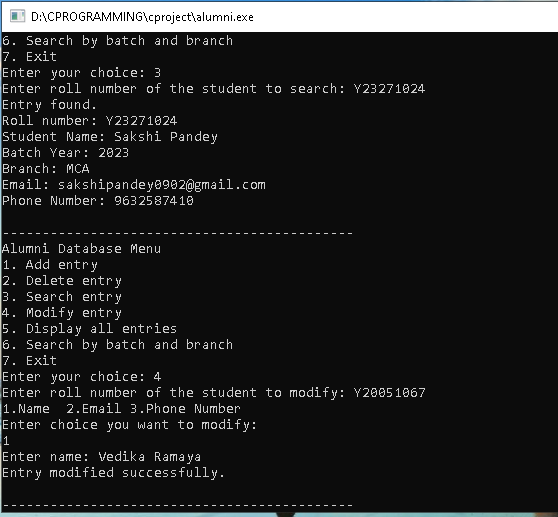


**SEARCH ENTRY**



**MODIFY ENTRY**





**SEARCH BY BRANCH AND BATCH YEAR**

