GOVERNMENT POLYTECHNIC, AHMEDABAD

COMPUTER ENGINEERING DEPARTMENT

Affiliated To

Gujarat Technological University, Ahmedabad

Micro project Report

D. E. Second Year (Semester– III)

Sub: Data Structure and Algorithms (4330704)

Government Polytechnic, Ahmedabad
Computer Engineering Department

CERTIFICATE

This is to certify that

Sr. No.	Enrollment No.	Name
1	Nagwadiya Tarun Pravinbhai	206170307221
2	Prakhar N. Parikh	216170307037
3	Chelaramani Pratham K.	216170307006

Of **Third** semester of Diploma in Computer Engineering of Government Polytechnic, Ahmedabad has completed the Micro-Project satisfactorily in Subject **Data Structure and Algorithms (4330704)** for the academic year **2022-2023** as prescribed in the curriculum.

Lecturer,
Computer Engg. Dept.,
Government Polytechnic, Ahmedabad

HOD
Computer Engg. Dept.,
Government Polytechnic,
Ahmedabad

RUBRICS FOR MICRO-PROJECT ASSESMENT

Parameters	Allocated Marks	High	Medium	Low
Problem Analysis and Solution(R1)	8	Problem is Properly Analysed and Solved	Problem is Properly Analysed but Partially Solved	Problem is Properly Analysed but not Solved.
		8 Marks	5 Marks	2 Marks
Viva Voce(R2)	2	Student Answered All The Viva Voce Questions	Student Answered Only A Few Viva Voce Questions	Student Did Not Answer Any Viva Voce Questions
		2 Marks	1 Marks	0 Marks

INDEX

1. Description of Problem
2. Solution of Problem in terms of Flowchart
3. Solution of Problem in terms of coding
4. Output (screenshots)
5. References if any

Enrollment Number	Student Name	Marks(R1)	Marks(R2)	Total Marks
206170307221	Nagwadiya Tarun Pravinbhai			
216170307006	Chelaramani Pratham K.			
216170307037	Prakhar N. Parikh			
Name and Sign of Faculty :				

1

Description of the problem

1) Arithmetic Expression Evaluator using Stack method (GUI) - Python

- This is a GUI (Graphical User Interface) software.
- It evaluates the given infix or postfix expressions and displays the answer.
- It uses the popular GUI framework 'Tkinter' also know as 'Tk/Tcl' which is a part of the 'Standard Python Library'.
- For sake of simplicity and code management, the code is divided into 2 parts : 1) Custom module for actual processing and calculation 2) Program for GUI

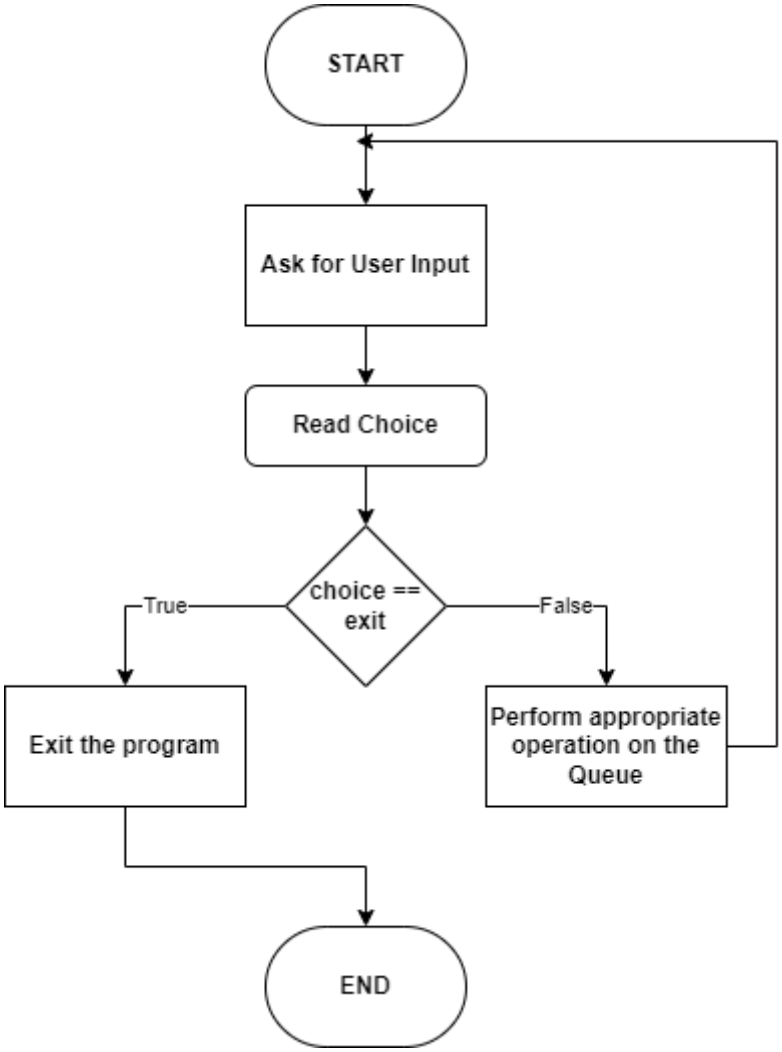
2) Person Queue Record Maintainer – C++

- This programs maintains a queue of persons using the 'Simple Queue' data structure.
- The user can add, remove or search a person in the queue.
- Such program can be used to keep the count of persons at ticket counters or food courts.

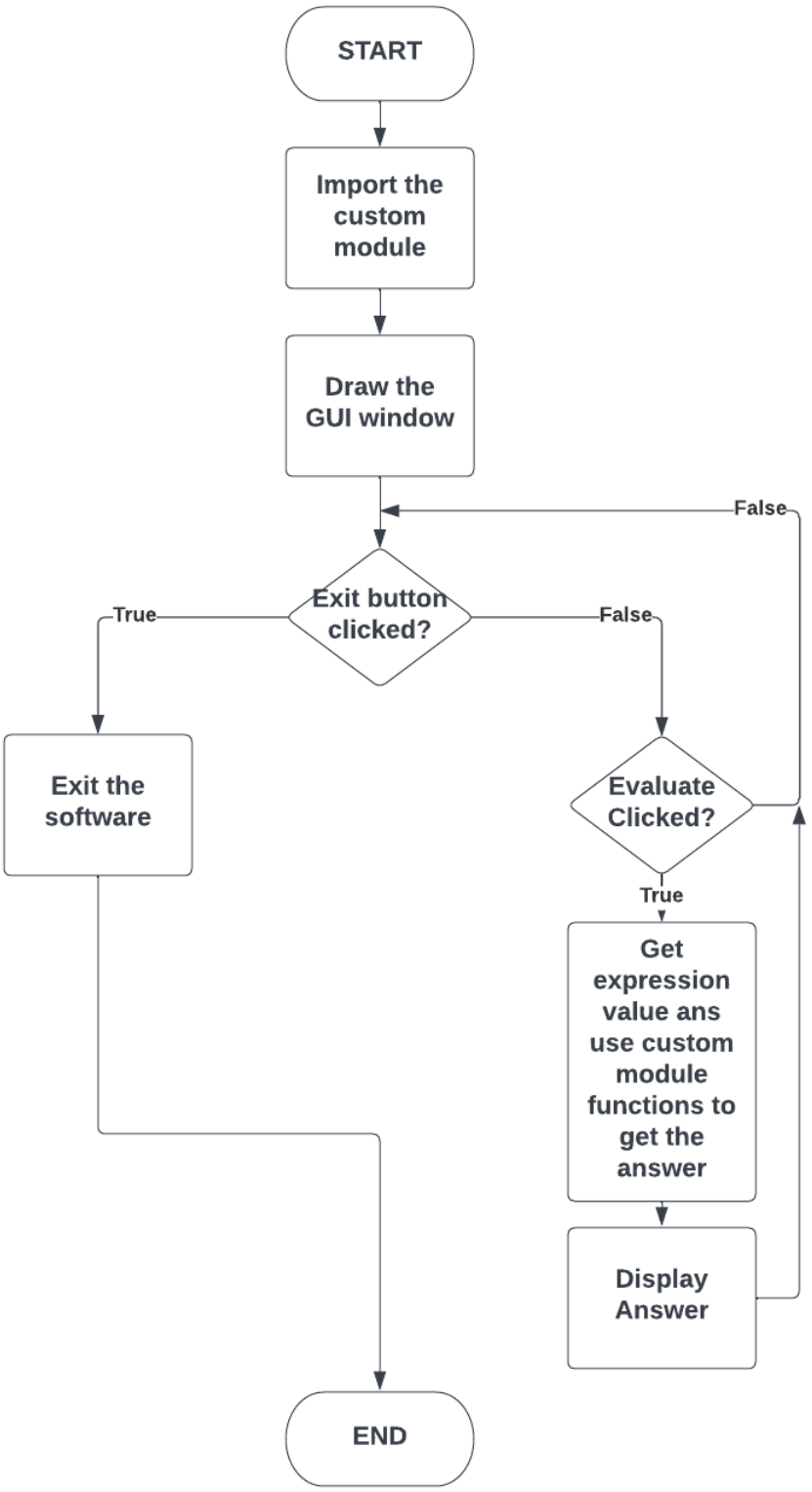
2

Solution of problem in terms of flowchart

1) Person Queue Record Maintainer



2) Arithmetic Expression Evaluator



3

Solution of problem in terms of coding

1) Arithmetic Expression Evaluator

i) Custom Evaluation Module

```
def evaluate_postfix(expression):
    stack = []

    expression = expression.split(" ")
    for character in expression:
        if character not in ['+', '-', '*', '/']:
            stack.append(character)
        else:
            operand1 = int(stack.pop(len(stack) - 1))
            operand2 = int(stack.pop(len(stack) - 1))
            result = 0
            if character == '+':
                result = operand2 + operand1
                stack.append(result)
            elif character == '-':
                result = operand2 - operand1
                stack.append(result)
            elif character == '*':
                result = operand2 * operand1
                stack.append(result)
            elif character == '/':
                result = operand2 // operand1
                stack.append(result)

    return stack[0]
```

```
def evaluate_prefix(expression):
    stack = []

    expression = reversed(expression.split(" "))
    for character in expression:
        if character not in ['+', '-', '*', '/']:
            stack.append(character)
        else:
            operand2 = int(stack.pop(len(stack) - 1))
            operand1 = int(stack.pop(len(stack) - 1))
            result = 0
            if character == '+':
                result = operand2 + operand1
                stack.append(result)
            elif character == '-':
                result = operand2 - operand1
                stack.append(result)
            elif character == '*':
                result = operand2 * operand1
                stack.append(result)
            elif character == '/':
                result = operand2 // operand1
                stack.append(result)

    return stack[0]
```

ii) Main Software

```
from tkinter import *
import arithmatic_evaluators

def evaluate_clicked(expression_entry, type_var, ans_label):
    if type_var.get() == 1:
        calculate_prefix(expression_entry.get(), ans_label)
    elif type_var.get() == 2:
        calculate_postfix(expression_entry.get(), ans_label)

def calculate_prefix(expression, ans_label):
    result = "Answer : " +
str(arithmatic_evaluators.evaluate_prefix(expression))
    ans_label.configure(text=result)
    ans_label.update()

def calculate_postfix(expression, ans_label):
```

```
    result = "Answer : " +
str(arithmatic_evaluators.evaluate_postfix(expression))
    ans_label.configure(text=result)
    ans_label.update()

if __name__ == '__main__':
    root = Tk()

    root.geometry("750x450")
    root.title("Arithmetic Expression Evaluator")
    root.configure(background="yellow")

    title_label = Label(root, text="Welcome to the arithmetic expression
evaluator", font=("Ariel", 20, "bold"))
    title_label.configure(background="yellow")
    title_label.configure(foreground="blue")
    title_label.place(x=50, y=20)

    info_label = Label(root, text="Please enter your expression",
font=("Ariel", 20, "bold"))
    info_label.configure(background="yellow")
    info_label.configure(foreground="red")
    info_label.place(x=170, y=100)

    note_label = Label(root, text="Note : Please enter the values space
seperated. For eg. 9 + 12 - 123", font=("Ariel", 12, "bold"))
    note_label.configure(background="yellow")
    note_label.configure(foreground="red")
    note_label.place(x=110, y=130)

    expression_entry = Entry(root, font=("Ariel", 15, "bold"))
    expression_entry.place(x=190, y=170, width=350)

    type_var = IntVar()
    prefix_button = Radiobutton(root, text="Prefix", variable=type_var,
value=1, font=("Ariel", 15, "bold"))
    prefix_button.configure(background="yellow")
    prefix_button.place(x=200, y=200)

    postfix_button = Radiobutton(root, text="Postfix", variable=type_var,
value=2, font=("Ariel", 15, "bold"))
    postfix_button.configure(background="yellow")
    postfix_button.place(x=400, y=200)

    evaluate_button = Button(root, text="Evaluate!", font=("Ariel", 15,
"bold"), command=lambda : evaluate_clicked(expression_entry, type_var,
ans_label))
    evaluate_button.configure(background="red")
```

```
    evaluate_button.configure(foreground="white")
    evaluate_button.place(x=300, y=250)

    ans_label = Label(root, text="", font=("Ariel", 20, "bold"),
anchor="center")
    ans_label.configure(background="yellow")
    ans_label.configure(foreground="red")
    ans_label.place(x=0, y=300, width=750)

    credits_label1 = Label(root, text="Developed by : Prakhar Parikh",
font=("Ariel", 14, "bold"))
    credits_label1.configure(background="yellow")
    credits_label1.configure(foreground="blue")
    credits_label1.place(x=200, y=350)
    credits_label2 = Label(root, text="Tarun Nagwadia", font=("Ariel", 14,
"bold"))
    credits_label2.configure(background="yellow")
    credits_label2.configure(foreground="blue")
    credits_label2.place(x=340, y=380)
    credits_label3 = Label(root, text="Pratham Chelaramani", font=("Ariel",
14, "bold"))
    credits_label3.configure(background="yellow")
    credits_label3.configure(foreground="blue")
    credits_label3.place(x=345, y=410)

    root.mainloop()
```

2) Person Queue Record Maintainer

```
#include<iostream>
#include<conio.h>
#include<stdlib.h>
#include<string>

#define MAX 10

using namespace std;

void queue_insert();
void queue_delete();
void display();
void queue_search();
string queue_array[MAX];
int rear = - 1;
```

```
int front = - 1;

int main()
{
    int choice;
    while (1)
    {
        cout << "1.Add person to queue \n";
        cout << "2.Delete person from queue \n";
        cout << "3.Display all person in queue \n";
        cout << "4.Search for a person in queue \n";
        cout << "5.Quit \n\n";
        cout << "Enter your choice : ";
        cin >> choice;
        cout << "\n\n";
        switch (choice)
        {
            case 1:
                queue_insert();
                cout << "\n\nPress any key to continue...";
                getch();
                system("cls");
                break;
            case 2:
                queue_delete();
                cout << "\n\nPress any key to continue...";
                getch();
                system("cls");
                break;
            case 3:
                display();
                cout << "\n\nPress any key to continue...";
                getch();
                system("cls");
                break;
            case 4:
                queue_search();
                cout << "\n\nPress any key to continue...";
                getch();
                system("cls");
                break;
            case 5:
                exit(1);
            default:
                cout << "Wrong choice \n";
                cout << "\n\nPress any key to continue...";
                getch();
                system("cls");
```

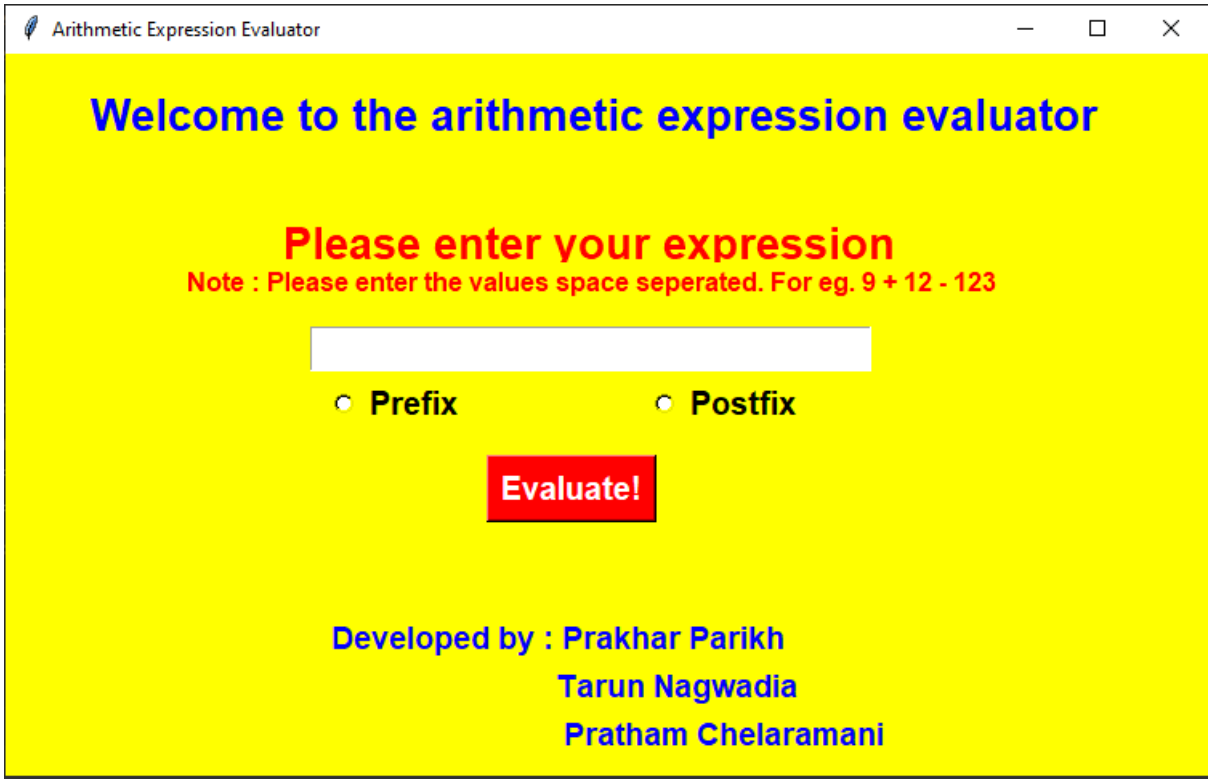
```
        }
    }
    return 0;
}

void queue_insert()
{
    string add_item;
    if (rear == MAX - 1)
    cout << "Queue Overflow \n";
    else
    {
        if (front == - 1)
        /*If queue is initially empty */
        front = 0;
        cout << "Insert the element in queue : ";
        cin >> add_item;
        rear = rear + 1;
        queue_array[rear] = add_item;
        cout << "\nElement inserted!";
    }
} /* End of queue_insert() */

void queue_delete()
{
    if (front == - 1 || front > rear)
    {
        cout << "Queue Underflow \n";
        return ;
    }
    else
    {
        cout << "Element deleted from queue is : " << queue_array[front] <<
endl;
        front = front + 1;
        if(front == rear){
            front = -1;
            rear = -1;
        }
    }
}

void display()
{
    cout << "The queue is : ";
    for (int i = front; i <= rear; i++)
        cout << queue_array[i] << " ";
    cout << "\n";
```

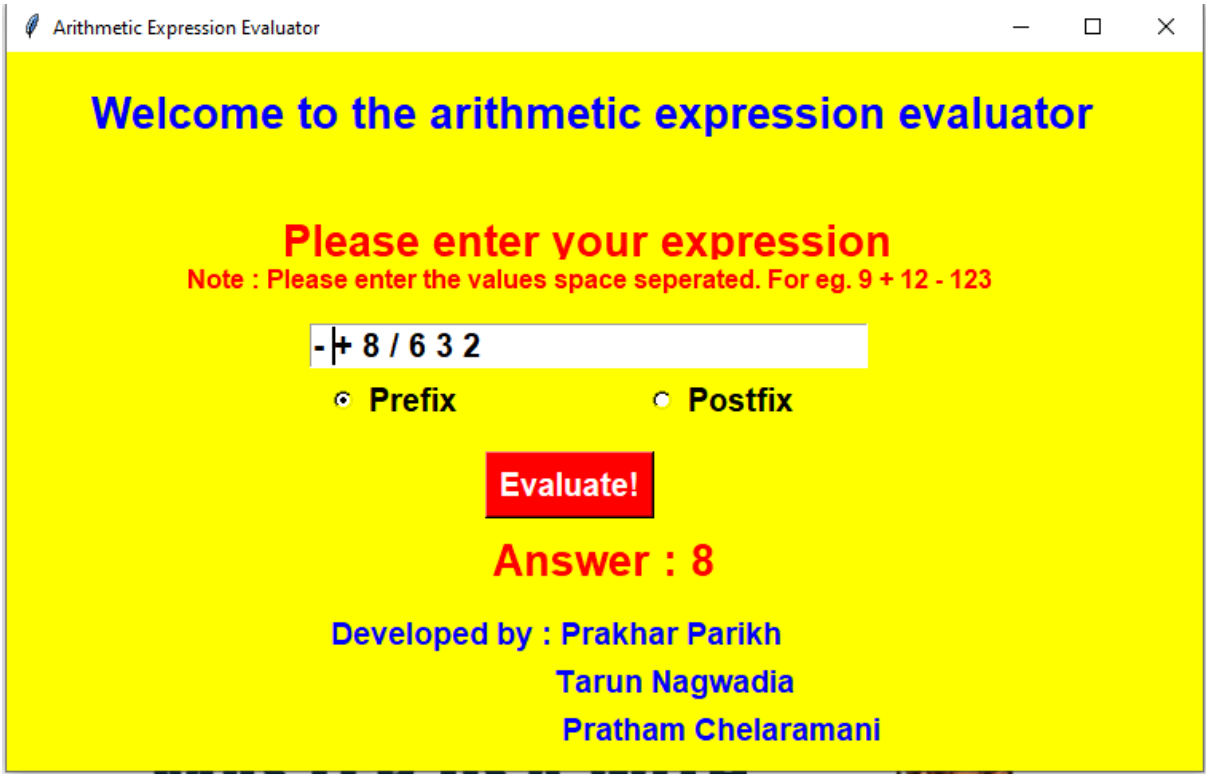
```
}

void queue_search()
{
    cout << "Enter name of person you want to search : ";
    string person;
    cin >> person;
    cout << endl;
    for (int i = front; i <= rear; i++){
        if(queue_array[i] == person){
            cout << "Person is waiting at position : " << i+1 << endl;
            goto finish;
        }
    }
    cout << "Person is not in the queue" << endl;
    finish:
    cout << "";
}
```

4

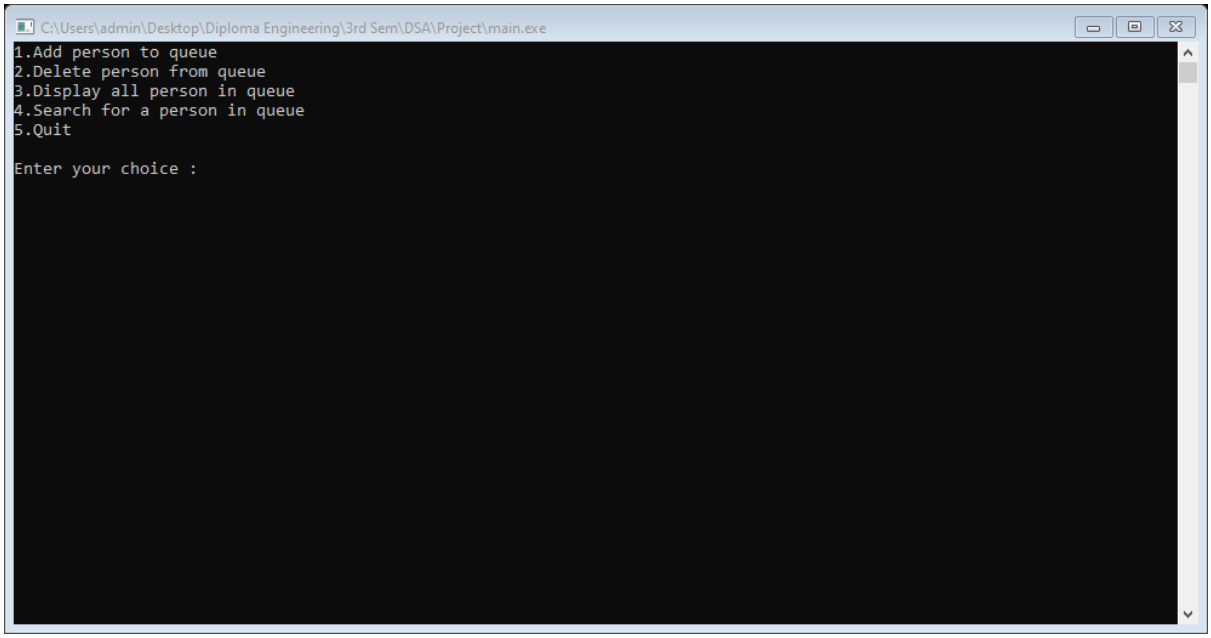
Screenshots

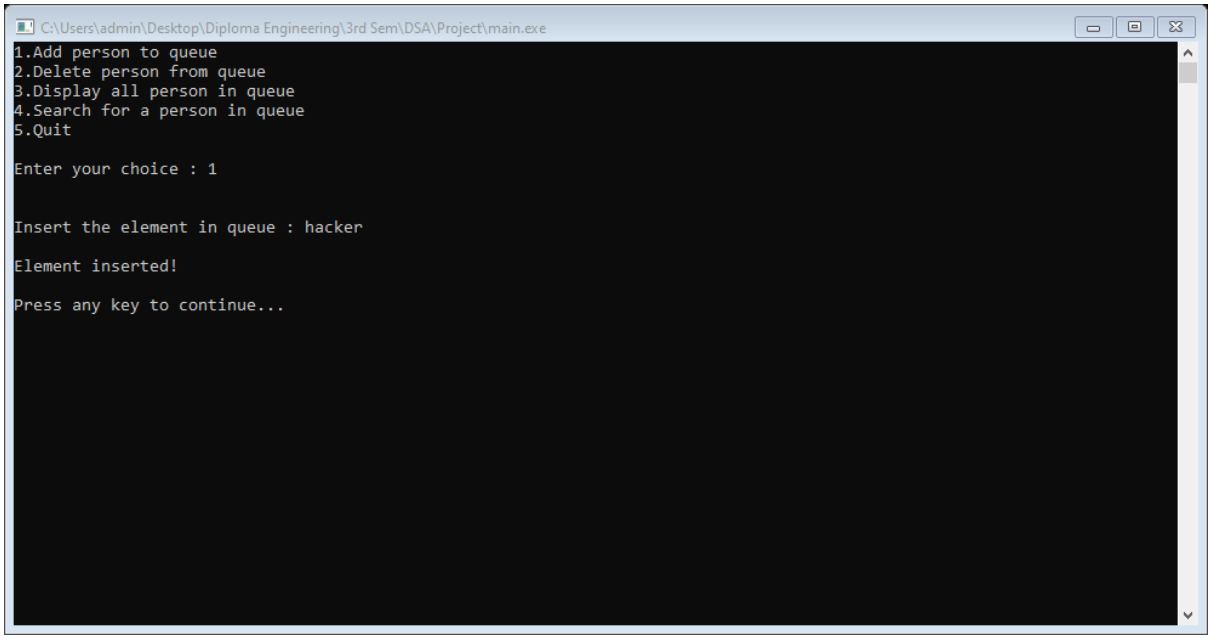
1) Arithmetic Expression Evaluator

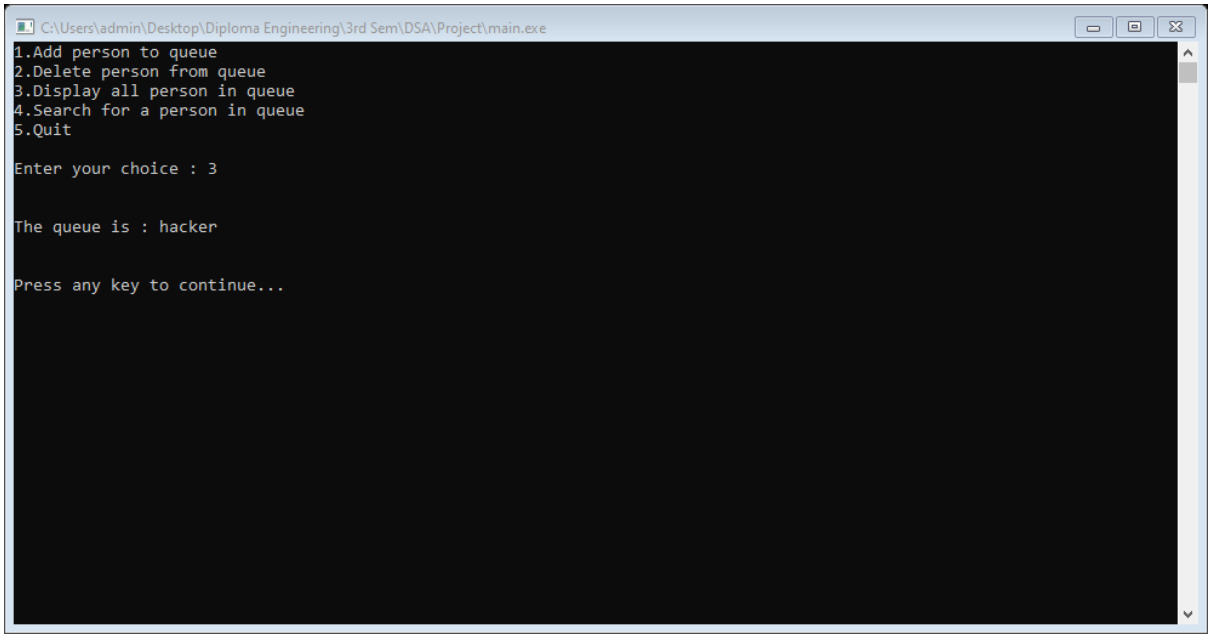


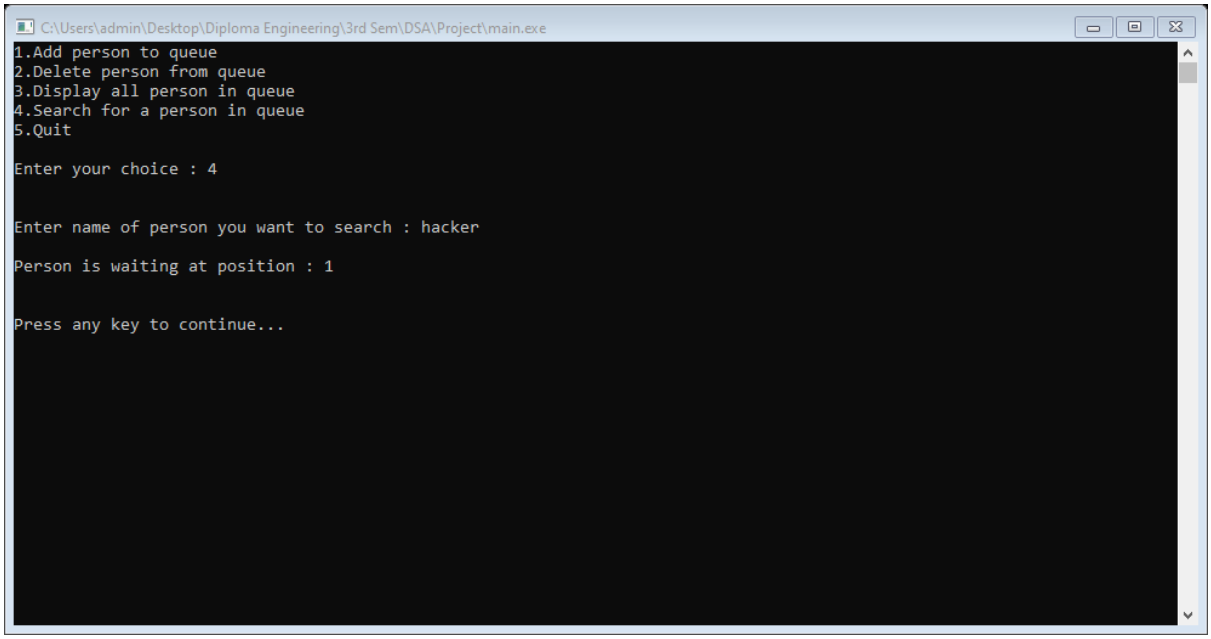


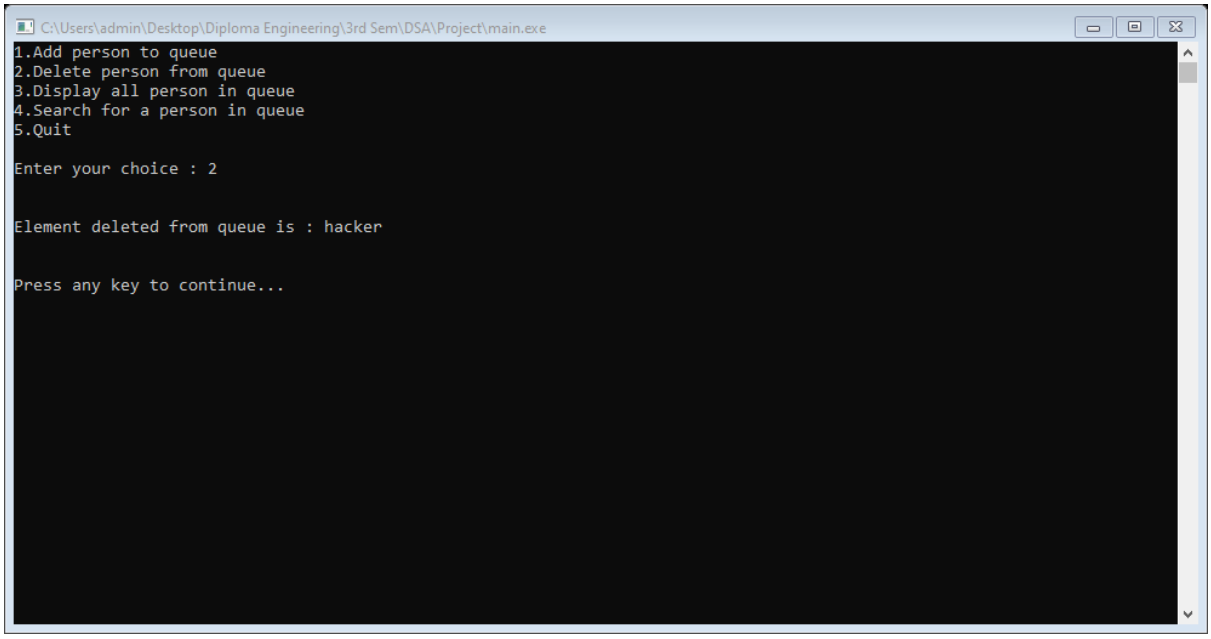
2) Person Queue Record Maintainer




C:\Users\admin\Desktop\Diploma Engineering\3rd Sem\DSA\Project\main.exe
1.Add person to queue
2.Delete person from queue
3.Display all person in queue
4.Search for a person in queue
5.Quit
Enter your choice : 1
Insert the element in queue : hacker
Element inserted!
Press any key to continue...


C:\Users\admin\Desktop\Diploma Engineering\3rd Sem\DSA\Project\main.exe
1.Add person to queue
2.Delete person from queue
3.Display all person in queue
4.Search for a person in queue
5.Quit
Enter your choice : 3
The queue is : hacker
Press any key to continue...


C:\Users\admin\Desktop\Diploma Engineering\3rd Sem\DSA\Project\main.exe
1.Add person to queue
2.Delete person from queue
3.Display all person in queue
4.Search for a person in queue
5.Quit
Enter your choice : 4
Enter name of person you want to search : hacker
Person is waiting at position : 1
Press any key to continue...


C:\Users\admin\Desktop\Diploma Engineering\3rd Sem\DSA\Project\main.exe
1.Add person to queue
2.Delete person from queue
3.Display all person in queue
4.Search for a person in queue
5.Quit
Enter your choice : 2
Element deleted from queue is : hacker
Press any key to continue...