#include<stdio.h>

typedef struct queue

{

char q[100];

int rear;

int front;

}QUEUE;

void init(QUEUE\* pq)

{

pq->rear = -1;

pq->front = 0;

}

int isempty(QUEUE\* pq)

{

return (pq->front > pq->rear);

}

int isfull(QUEUE\* pq)

{

return (pq->rear == 99);

}

void enqueue(QUEUE\* pq, int ele)

{

if(isfull(pq))

printf("overflow\n");

else

{

pq->rear++;

pq->q[pq->rear] = ele;

}

}

int dequeue(QUEUE\* pq)

{

int ele;

if(isempty(pq))

{ printf("underflow\n");

ele= -1;

}

else

{

ele = pq->q[pq->front];

pq->front++;

}

return ele;

}

int display(QUEUE\* pq)

{

int k=pq->front;

while(k <= pq->rear)

{

printf("%d\t", pq->q[k]);

k++;

}

}

int ToQ(QUEUE\* pq)

{

if (pq->front > pq->rear)

printf("Underflow\n");

else

return pq->q[pq->front];

}

int main()

{

QUEUE pq;

init(&pq);

int ch,ele;

while(1)

{ printf("enter the operation to be performed:\n");

printf("1:Enqueue\n2:Dequeue\n3:display\n4:isempty\n5:ToQ\n");

scanf("%d", &ch);

switch(ch)

{

case 1: printf("enter the element\n");

scanf("%d",&ele);

enqueue(&pq,ele);

break;

case 2: ele = dequeue(&pq);

printf("deleted element is %d",ele);

break;

case 3: printf("the elements are:\n");

display(&pq);

break;

case 4: if(isempty(&pq))

printf("Empty\n");

else

printf("Not empty\n");

break;

case 5: printf("%d\n",ToQ(&pq));

break;

default:

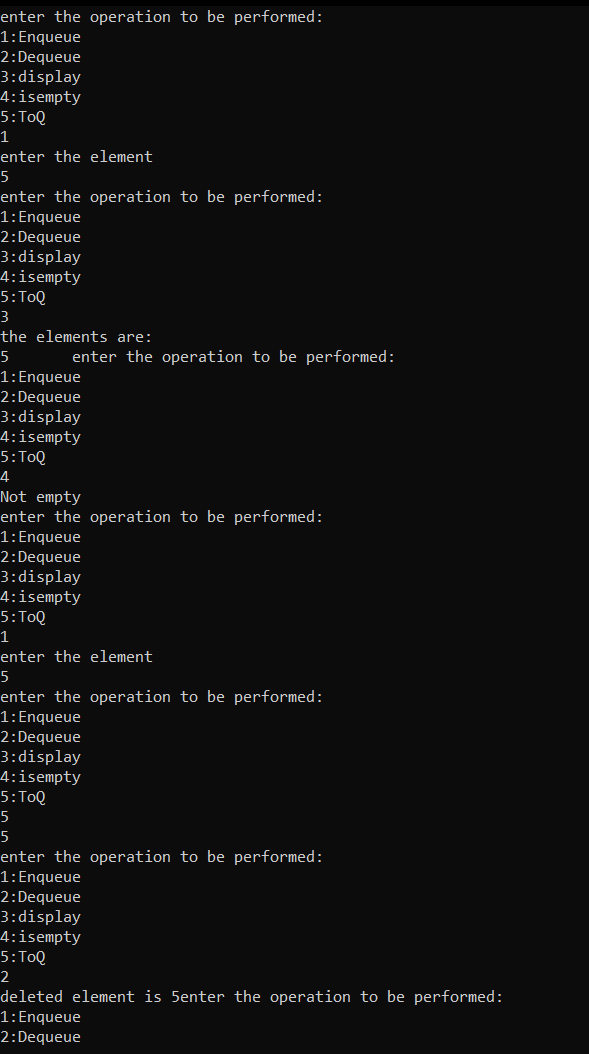
printf("Enter proper no;");

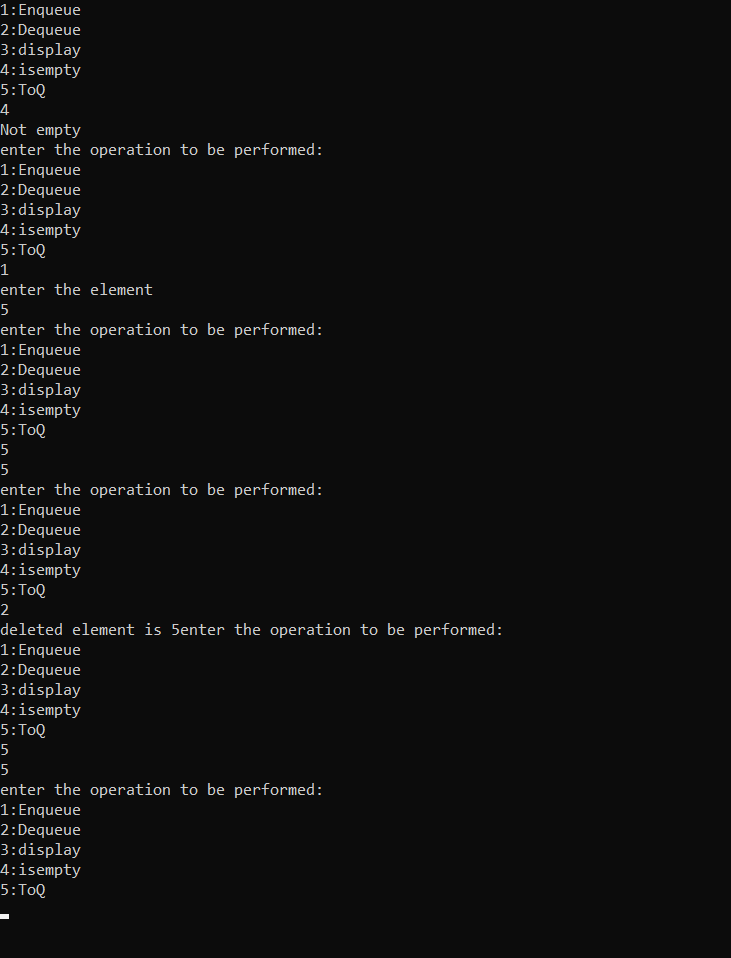
}

}

return 0;

}





Program 2

#include<stdio.h>

#include<stdlib.h>

#include<limits.h>

#include<string.h>

#define MAX 8

typedef struct appointment{

char patient\_name[25];

char date[6];

char slot[11];

}appoint;

appoint queue[MAX];

unsigned int size = 0;

unsigned int rear = MAX - 1;

unsigned int front = 0;

char \*slots[11] = {"09AM - 10AM", "10AM - 11AM","11AM - 12PM", "12PM - 01PM","01PM - 02PM","02PM - 03PM","03PM-04PM", "04PM - 05PM"};

char DATE[6];

char TDATE[6];

int enqueue();

int dequeue();

int isFull();

int isEmpty();

int main()

{

int ch;

printf("Enter today's date in the form DDMMYY:");

scanf("%s",TDATE);

/\* Run indefinitely until user manually terminates \*/

while (1)

{

printf("Choose an option.\n");

printf("1. Make an appointment\n");

printf("2. Attend an appointment. \n");

printf("3. Exit\n");

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

printf("Select an option: ");

scanf("%d", &ch);

switch (ch)

{

case 1:

if (enqueue())

printf("Element added to queue.");

else

printf("Queue is full.");

break;

case 2:

if (dequeue())

printf("Thank you.");

else

printf("No appointments have been scheduled for today.");

break;

case 3:

exit(0);

default:

printf("Invalid choice, please input number between (0-5).");

break;

}

printf("\n");

}

}

int enqueue()

{

if (isFull())

{

return 0;

}

rear = (rear + 1) % MAX;

char temp\_name[25];

printf("Enter your name: \n");

scanf("%s",temp\_name);

strcpy(queue[rear].patient\_name, temp\_name);

strcpy(queue[rear].slot, slots[size]);

printf("Enter date of appointment in the form DDMMYY:");

scanf("%s",DATE);

if(strcmp(DATE,TDATE)>=0)

{

strcpy(queue[rear].date, DATE);

printf("Patient %s ,scheduled appointment slot %s ", queue[rear].patient\_name,slots[rear]);

}

else

printf("%s\n","Invalid Date");

size++;

return 1;

}

int dequeue()

{

int data = INT\_MIN;

if (isEmpty())

{

return 0;

}

printf("Patient %s , you may now enter for your scheduled appointment slot %s", queue[front].patient\_name,slots[front]);

front = (front + 1) % MAX;

return 1;

}

int isFull()

{

return (size == MAX);

}

int isEmpty()

{

return (size == 0);

}

