1

A-2 Write a Python program to store marks scored in subject “Fundamental of Data Structure” by N students in the class. Write functions to compute following:

1. The average score of class
2. Highest score and lowest score of class
3. Count of students who were absent for the test
4. Display mark with highest frequency

**marklist = [20,50, 40, 50, 90, 50, 90, 90, None, 89, None]**

**#n=int (imat "Enter no of students "))**

**# for i in range(n):**

**# mark = int(input (f"Enter marks (i+1) student :"))**

**#marklist.append(mark)**

**print (marklist)**

**total = 0**

**absent\_student = 0**

**freq = {}**

**max\_val = marklist [0]**

**min\_val = marklist [0]**

**for mark in marklist:**

**if mark == None:**

**absent\_student += 1**

**else:**

**total += mark**

**# calculate Max and min marks**

**if mark < min\_val:**

**min\_val = mark**

**if max\_val < mark:**

**max\_val = mark**

**if mark != None:**

**if freq.get(mark) == None:**

**freq[mark] = 1**

**else:**

**freq[mark] += 1**

**print ("a.Average Score of the class = ",total/len (marklist))**

**print ("b.Highest Score= ",max\_val ,"and Lowest Score =",min\_val)**

**print ("c. Number of absent student= ",absent\_student)**

**highest\_freq = 0**

**highest\_freq\_mark = 0**

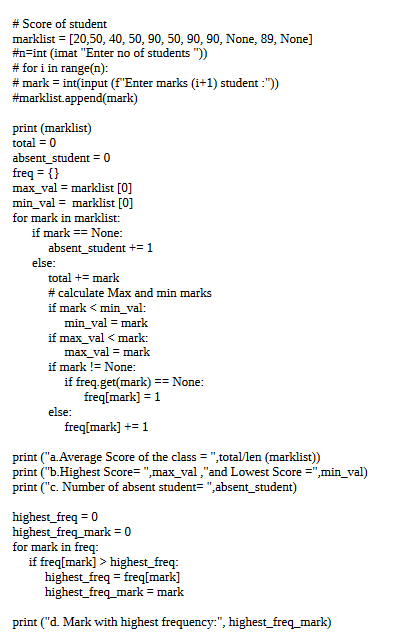
**for mark in freq:**

**if freq[mark] > highest\_freq:**

**highest\_freq = freq[mark]**

**highest\_freq\_mark = mark**

**print ("d. Mark with highest frequency:", highest\_freq\_mark)**



**2**

In second year computer engineering class, group A student’s play cricket, group B students play badminton and group C students play football.

Write a Python program using functions to compute following: -

List of students who play both cricket and badminton

List of students who play either cricket or badminton but not both

Number of students who play neither cricket nor badminton

Number of students who play cricket and football but not badminton.

def intersection(l1,l2):

res = []

for student in l1:

if student in l2:

res.append(student)

return res

def union (l1, l2):

res = l2.copy()

for student in l1:

if not student in l2:

res.append(student)

return res

def diff(l1, l2):

res = []

for student in l1:

if not student in l2:

res.append(student)

return res

a = [1,2,3,4,5,6,7]

b = [2,3,5,7,9,10]

c = [2,4,6,8,10,12]

print (f"A = {a} \nB = {b} \nC ={c} \n")

print (f"a. {intersection(a, b)}")

# print(union(a,b))

# print(diff(a,b))

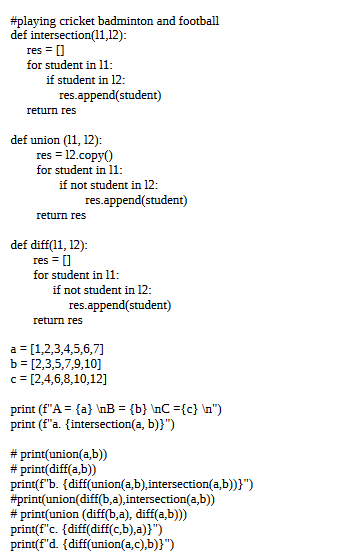
print(f"b. {diff(union(a,b),intersection(a,b))}")

#print(union(diff(b,a),intersection(a,b))

# print(union (diff(b,a), diff(a,b)))

print(f"c. {diff(diff(c,b),a)}")

print(f"d. {diff(union(a,c),b)}")

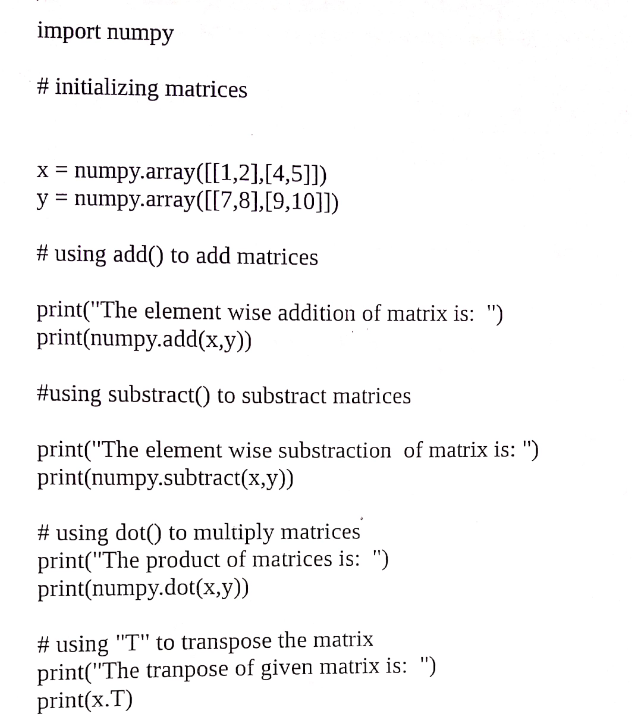


3

Write a Python program to compute following computation on matrix:

Addition of two matrices B) Subtraction of two matrices

c) Multiplication of two matrices d) Transpose of a matrix



4

Write a Python program to store first year percentage of students in array. Write function for sorting array of floating point numbers in ascending order using

Selection Sort and display top 5 scores

marks=[3,46,8,31,2,85,87]

def selectionsort(marks):

for i in range(len(marks)):

min\_idx = i

for j in range(i+1,len(marks)):

if marks[min\_idx]>marks[j]:

min\_idx =j

marks[i], marks[min\_idx] = marks[min\_idx],marks[i]

print("Marks of students after performing selection sort on the list: ")

for i in range(len(marks)):

print(marks[i])

selectionsort(marks)

5

Write a Python program to store first year percentage of students in array. Write function for sorting array of floating point numbers in ascending order using quick sort and display top five scores.

9

Department of Computer Engineering has student's club named 'Pinnacle Club'. Students of second, third and final year of department can be granted membership on request. Similarly one may cancel the membership of club. First node is reserved for president of club and last node is reserved for secretary of club. Write C++ program to maintain club member‘s information using singly linked list. Store student PRN and Name. Write functions to:

Add and delete the members as well as president or even secretary.

Compute total number of members of club

Display members

Two linked lists exists for two divisions. Concatenate two lists

#include<iostream>

Using namespace std;

class sll

{

struct node

{

int prn;

char name[10];

node \*next;

}\*start;

public:

sll()

{

start=NULL;

}

void create();

void display();

void insert\_beginning();

void insert\_end();

void insert\_mid();

void del\_beginning();

void del\_end();

void del\_mid();

int compute();

void concatenate(sll obj2);

};

void sll::create()

{

node \*temp;

node \*curr;

int prn;

char name;

int ans;

do{

temp=new node;

temp->next=NULL;

cout<<"Enter PRN number:"<<endl;

cin>>temp->prn;

cout<<"Enter name:"<<endl;

cin>>temp->name;

if(start==NULL)

{

start=temp;

curr=temp;

}

else

{

curr->next=temp;

curr=temp;

}

cout<<"Do you want to add new node? 1 for yes"<<endl;

cin>>ans;

}while(ans==1);

}

void sll::display()

{

node \*t;

if(start==NULL)

{

cout<<"Club is emty"<<endl;

return;

}

else{

t=start;

while(t!=NULL)

{

cout<<t->prn<<" "<<t->name<<"->";

t=t->next;

}

cout<<"NULL";

}

}

void sll::insert\_beginning()

{

node \*temp;

temp =new node;

temp->next=NULL;cout<<"Enter PRN number:"<<endl;

cin>>temp->prn;

temp->next=NULL;cout<<"Enter name:"<<endl;

cin>>temp->name;

if(start==NULL)

{

start=temp;

}

else{

temp->next;

start=temp;

}

}

void sll::insert\_end()

{

node \*temp, \*last;

temp =new node;

temp->next=NULL;

cout<<"Enter PRN number:"<<endl;

cin>>temp->prn;

temp->next=NULL;

cout<<"Enter name:"<<endl;

cin>>temp->name;

if(start==NULL)

{

start=temp;

}

else{

last=start;

while(last->next!=NULL)

{

last=last->next;

}

last->next=temp;

}

}

void sll::insert\_mid()

{

node \*temp;

node \*curr;

int loc;

cout<<"\nENter location after which we want to insert:"<<endl;

cin>>loc;

temp=new node;

temp->next=NULL;

cout<<"Enter PRN number:"<<endl;

cin>>temp->prn;

cout<<"Enter name:"<<endl;

cin>>temp->name;

curr=start;

for(int i=1;i<loc;i++)

{

curr=curr->next;

}

temp->next=curr->next;

curr->next=temp;

}

void sll::del\_beginning()

{

node \*temp;

if(start==NULL)

{

cout<<"Club is empty"<<endl;

}

else{

temp=start;

start=start->next;

cout<<temp->prn<<"\tfirst element deleted"<<endl;

delete temp;

}

}

void sll::del\_end()

{

node \*temp,\*prev;

if(start==NULL)

{

cout<<"Club is empty"<<endl;

}

else{

temp=start;

while(temp->next==NULL)

{

prev=temp;

temp=temp->next;

}

cout<<temp->prn<<"\t last element deleted"<<endl;

delete temp;

prev->next=NULL;

}

}

void sll:: del\_mid()

{

node \*temp;

node \*curr;

int loc;

cout<<"Enter location of node which you want to delete"<<endl;

cin>>loc;

curr=start;

for(int i=1;i<loc;i++)

{

temp=curr;

curr=curr->next;

}

temp->next=curr->next;

cout<<curr->prn<<"\t has been deleted"<<endl;

delete curr;

}

int sll::compute()

{

node \*temp;

int count=0;

if(start==NULL)

{

cout<<"\nClub is empty"<<endl;

}

temp=start;

while(temp!=NULL)

{

count++;

temp=temp->next;

}

cout<<"Total number of members are \t"<<endl;

return 0;

}

void sll::concatenate(sll obj2)

{

node \*temp, \*last;

last=obj2.start;

if(last==NULL)

{

cout<<"\n List 2 is empty"<<endl;

return;

}

temp=start;

while(temp->next!=NULL)

{

temp=temp->next;

}

temp->next=last;

cout<<"\n After concatenation:";

}

int main()

{

sll obj;

int ch;

do{

cout<<"\n1. Create \n2.Insert at beginning \n3.Insert at end \n4.INsert after position \n5 Display list \n6 Delete first element \n7 Delete last element \n8 DElete Member \n9 Find total number of members \n10 concate lists \n0.Exit \n Enter your choice:";

cin>>ch;

switch(ch)

{

case 1:

obj.create();

obj.display();

break;

case 2:

obj.insert\_beginning();

obj.display();

break;

case 3:

obj.insert\_end();

obj.display();

break;

case 4:

obj.insert\_mid();

obj.display();

break;

case 5:

obj.display();

break;

case 6:

obj.del\_beginning();

obj.display();

break;

case 7:

obj.del\_end();

obj.display();

break;

case 8:

obj.del\_mid();

obj.display();

break;

case 9:

obj.compute();

obj.display();

break;

case 10:

{

sll obj2,obj3;

cout<<"\n List1:"<<endl;

obj2.create();

cout<<"\n List2:"<<endl;

obj3.create();

obj2.concatenate(obj3);

obj.display();

break;

}

}

}while(ch!=0);

cout<<"\n End of the program"<<endl;

return 0;

}

OR

In any language program mostly syntax error occurs due to unbalancing delimiter such as (),{},[]. Write C++ program using stack to check whether given expression is well parenthesized or not.

7][ln

10

The ticket booking system of Cinemax theater has to be implemented using C++ program. There are 10 rows and 7 seats in each row. Doubly circular linked list has to be maintained to keep track of free seats at rows. Assume some random booking to start with. Use array to store pointers (Head pointer) to each row. On demand

The list of available seats is to be displayed

The seats are to be booked

The booking can be cancelled

#include <iostream>

#include<stdlib.h>

using namespace std;

class node

{ public:

node\* next;

node\* prev;

int seat;

string id;

int status;

};

class cinemax

{

public:

node\* head,\* tail ,\* temp;

cinemax()

{

head=NULL;

}

void create\_list();

void display();

void book();

void cancel();

void avail();

};

void cinemax::create\_list()

{

int i=1;

temp=new node;

temp->seat=1;

temp->status=0;

temp->id="null";

tail=head=temp;

for(int i=2;i<=70;i++)

{

node \*p;

p= new node;

p->seat=i;

p->status=0;

p->id="null";

tail->next=p;

p->prev=tail;

tail=p;

tail->next=head;

head->prev=tail;

}

}

void cinemax::display()

{

{ int r=1;

node\* temp;

temp=head;

int count=0;

cout<<"\n------------------------------------------------------------------------------------\n";

cout<<" Screen this way \n";

cout<<"------------------------------------------------------------------------------------\n";

while(temp->next!=head)

{

if(temp->seat/10==0)

cout<<"S0"<<temp->seat<<" :";

else

cout<<"S"<<temp->seat<<" :";

if(temp->status==0)

cout<<"|\_| ";

else

cout<<"|B| ";

count++;

if(count%7==0)

{

cout<<endl;

r++;

}

temp=temp->next;

}

cout<<"S"<<temp->seat<<" :";

if(temp->status==0)

cout<<"|\_| ";

else

cout<<"|B| ";

}

}

void cinemax::book()

{ int x;

string y;

label:

cout<<"\n\n\nEnter seat number to be booked\n";

cin>>x;

cout<<"Enter your id number\n";

cin>>y;

if(x<1||x>70)

{

cout<<"Enter correct seat number to book (1-70)\n";

goto label;

}

node \*temp;

//temp=new node;

temp=head;

while(temp->seat!=x)

{

temp=temp->next;

}

if(temp->status==1)

cout<<"seat already booked!\n";

else{

temp->status=1;

temp->id=y;

cout<<"seat "<<x<<" booked!\n";

}

}

void cinemax::cancel()

{

int x;

string y;

label1:

cout<<"enter seat number to cancel booking\n";

cin>>x;

cout<<"enter you id\n";

cin>>y;

if(x<1||x>70)

{

cout<<"Enter correct seat number to cancel (1-70)\n";

goto label1;

}

node \*temp;

//temp=new node;

temp=head;

while(temp->seat!=x)

{

temp=temp->next;

}

if(temp->status==0)

{

cout<<"seat not booked yet!!\n";

}

else

{

if(temp->id==y)

{

temp->status=0;

cout<<"seat cancelled!\n";

}

else

cout<<"wrong user id ,seat cannot be cancelled\n";

}

}

void cinemax::avail()

{

int r=1;

node\* temp;

temp=head;

int count=0;

cout<<"\n\n\n\n";

cout<<"---------------------------------------------------------------------------------------------\n";

cout<<" Screen this way \n";

cout<<"---------------------------------------------------------------------------------------------\n";

while(temp->next!=head)

{

{

if(temp->seat/10==0)

cout<<"S0"<<temp->seat<<" :";

else

cout<<"S"<<temp->seat<<" :";

if(temp->status==0)

cout<<"|\_| ";

else if(temp->status==1)

cout<<" ";

count++;

if(count%7==0)

{

cout<<endl;

}

}

temp=temp->next;

}

if(temp->status==0)

{

cout<<"S"<<temp->seat<<" :";

if(temp->status==0)

cout<<"|\_| ";

}

}

int main()

{ cinemax obj;

obj.create\_list();

int ch;

char c='y';

while(c=='y')

{ obj.display();

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout<<" CINEMAX MOVIE THEATRE \n";

cout<<" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout<<"\nEnter choice\n1.Current seat status\n2.Book seat \n3.Available seat\n4.Cancel seat\n";

cin>>ch;

switch(ch)

{

case 1:obj.display();

break;

case 2: obj.book();

break;

case 3:obj.avail();

break;

case 4: obj.cancel();

break;

default: cout<<"Wrong choice input\n";

}

cout<<"\nWant operation again (y/n)\n";

cin>>c;

system("clear");

}

rreturn 0;

}

OR

Write a C++ program to check the given string is palindrome or not.

#include<iostream>

#include<string.h>

#define max 50

using namespace std;

class STACK

{

private:

char a[max];

int top;

public:

STACK()

{

top=-1;

}

void push(char);

void reverse();

void convert(char[]);

void palindrome();

};

void STACK::push(char c)

{

top++;

a[top] = c;

a[top+1]='\0';

cout<<endl<<c<<" is pushed on stack ...";

}

void STACK::reverse()

{

char str[max];

cout<<"\n\nReverse string is : ";

for(int i=top,j=0; i>=0; i--,j++)

{

cout<<a[i];

str[j]=a[i];

}

cout<<endl;

}

void STACK::convert(char str[])

{

int j,k,len = strlen(str);

for(j=0, k=0; j<len; j++)

{

if( ( (int)str[j] >= 97 && (int)str[j] <=122 ) || ( (int)str[j] >= 65 && (int)str[j] <=90 ))

{

if( (int)str[j] <=90 )

{

str[k] = (char)( (int)str[j] + 32 );

}else

{

str[k] = str[j];

}

k++;

}

}

str[k]='\0';

cout<<endl<<"Converted String : "<<str<<"\n";

}

void STACK::palindrome()

{

char str[max];

int i,j;

for(i=top,j=0; i>=0; i--,j++)

{

str[j]=a[i];

}

str[j]='\0';

if(strcmp(str,a) == 0)

cout<<"\n\nString is palindrome...";

else

cout<<"\n\nString is not palindrome...";

}

int main()

{

STACK stack;

char str[max];

int i=0;

cout<<"\nEnter string to be reversed and check is it palindrome or not : \n\n";

cin.getline(str , 50);

stack.convert(str);

while(str[i] != '\0')

{

stack.push(str[i]);

i++;

}

stack.palindrome();

stack.reverse();

}

11

In any language program mostly syntax error occurs due to unbalancing delimiter such as (),{},[]. Write C++ program using stack to check whether given expression is well parenthesized or not.

#include <iostream>

using namespace std;

#define size 10

class stackexp

{

int top;

char stk[size];

public:

stackexp()

{

top=-1;

}

void push(char);

char pop();

int isfull();

int isempty();

};

void stackexp::push(char x)

{

top=top+1;

stk[top]=x;

}

char stackexp::pop()

{

char s;

s=stk[top];

top=top-1;

return s;

}

int stackexp::isfull()

{

if(top==size)

return 1;

else

return 0;

}

int stackexp::isempty()

{

if(top==-1)

return 1;

else

return 0;

}

int main()

{

stackexp s1;

char exp[20],ch;

int i=0;

cout << "\n\t!! Parenthesis Checker..!!!!" << endl; // prints !!!Hello World!!!

cout<<"\nEnter the expression to check whether it is in well form or not : ";

cin>>exp;

if((exp[0]==')')||(exp[0]==']')||(exp[0]=='}'))

{

cout<<"\n Invalid Expression.....\n";

return 0;

}

else

{

while(exp[i]!='\0')

{

ch=exp[i];

switch(ch)

{

case '(':s1.push(ch);break;

case '[':s1.push(ch);break;

case '{':s1.push(ch);break;

case ')':s1.pop();break;

case ']':s1.pop();break;

case '}':s1.pop();break;

}

i=i+1;

}

}

if(s1.isempty())

{cout<<"\nExpression is well parenthesised...\n";}

else

{cout<<"\nSorry !!! Invalid Expression or not in well parenthesized....\n";}

return 0;

}

12

Write C++ program for simulating job queue. Write functions to add job and delete job from queue.

#include <iostream>

#define MAX 10

using namespace std;

struct queue

{

int data[MAX];

int front,rear;

};

class Queue

{

struct queue q;

public:

Queue(){q.front=q.rear=-1;}

int isempty();

int isfull();

void enqueue(int);

int delqueue();

void display();

};

int Queue::isempty()

{

return(q.front==q.rear)?1:0;

}

int Queue::isfull()

{

return(q.rear==MAX-1)?1:0;

}

void Queue::enqueue(int x)

{

q.data[++q.rear]=x;

}

int Queue::delqueue()

{

return q.data[++q.front];

}

void Queue::display()

{

int i;

cout<<"\n";

for(i=q.front+1;i<=q.rear;i++)

{

cout<<q.data[i]<<" ";

}

}

int main()

{

Queue obj;

int ch,x;

do

{

cout<<"\nWelcome\nPlease enter your choice:\n1.Insert Job\n2.Delete Job\n3.Display\n4.Exit ";

cin>>ch;

switch(ch)

{

case 1:

{

if (!obj.isfull())

{

cout<<"\nEnter data: ";

cin>>x;

obj.enqueue(x);

cout<<endl;

}

else

{

cout<<"Queue Overflow\n";

}

break;

}

case 2:

{

if(!obj.isempty())

{

cout<<"\nDeleted Element= "<<obj.delqueue()<<endl;

}

else

{

cout<<"\nQueue Underflow\n";

}

cout<<"\nRemaining Jobs: \n";

obj.display();

break;

}

case 3:

{

if (!obj.isempty())

{

cout<<"\nQueue contains: \n";

obj.display();

}

else

{

cout<<"\nQueue is empty\n";

}

break;

}

case 4:

{

exit(0);

}

default:

{

cout<<"Invalid choice";

}

}

}

while(true);

return 0;

}

13

Write C++ program to simulate deque with functions to add and delete elements from either end of the deque.

#include <iostream>

#define MAX 10

using namespace std;

struct queue

{

int data[MAX];

int front,rear;

};

class Queue

{

struct queue q;

public:

Queue(){q.front=q.rear=-1;}

int isempty();

int isfull();

void enqueue(int);

int delqueue();

void display();

};

int Queue::isempty()

{

return(q.front==q.rear)?1:0;

}

int Queue::isfull()

{

return(q.rear==MAX-1)?1:0;

}

void Queue::enqueue(int x)

{

q.data[++q.rear]=x;

}

int Queue::delqueue()

{

return q.data[++q.front];

}

void Queue::display()

{

int i;

cout<<"\n";

for(i=q.front+1;i<=q.rear;i++)

{

cout<<q.data[i]<<" ";

}

}

int main()

{

Queue obj;

int ch,x;

do

{

cout<<"\nWelcome\nPlease enter your choice:\n1.Insert Job\n2.Delete Job\n3.Display\n4.Exit ";

cin>>ch;

switch(ch)

{

case 1:

{

if (!obj.isfull())

{

cout<<"\nEnter data: ";

cin>>x;

obj.enqueue(x);

cout<<endl;

}

else

{

cout<<"Queue Overflow\n";

}

break;

}

case 2:

{

if(!obj.isempty())

{

cout<<"\nDeleted Element= "<<obj.delqueue()<<endl;

}

else

{

cout<<"\nQueue Underflow\n";

}

cout<<"\nRemaining Jobs: \n";

obj.display();

break;

}

case 3:

{

if (!obj.isempty())

{

cout<<"\nQueue contains: \n";

obj.display();

}

else

{

cout<<"\nQueue is empty\n";

}

break;

}

case 4:

{

exit(0);

}

default:

{

cout<<"Invalid choice";

}

}

}

while(true);

return 0;

}

14 Pizza parlor accepting maximum M orders. Orders are served in first come first served basis. Order once placed cannot be cancelled. Write C++ program to simulate the system using circular queue using array.

#include<iostream>

#include<cstdlib>

using namespace std;

class pizza

{

 int front,rear,q[5];

 public:

 pizza()

 {

 front=-1;

 rear=-1;

 }

 int isfull()

 {

if((front==0&&rear==4)||front==rear+1)

 {

 return 1;

 }

else

{

 return 0;

 }

 }

 int isempty()

 {

 if(front==-1&&rear==-1)

 {

 return 1;

 }

 else

{

 return 0;

 }

 }

 void add()

 {

 if(isfull()==0)

 {

 cout<<"\n Enter the Pizza ID: ";

 if(front==-1&&rear==-1)

 {

 front=0;

 rear=0;

 cin>>q[rear];

 }

 else

 {

 rear=(rear+1)%5;

 cin>>q[rear];

 }

 char c;

 cout<<" Do you want to add another order ? ";

 cin>>c;

 if(c=='y'||c=='Y')

 add();

 }

 else

 {

 cout<<"\n Orders are full ";

 }

 }

 void serve()

 {

 if(isempty()==0)

 {

 if(front==rear)

 {

 cout<<"\n Order served is : "<<q[front];

 front=-1;

 rear=-1;

 }

 else

 {

 cout<<"\n Order served is : "<<q[front];

 front=(front+1)%5;

 }

 }

 else

 {

 cout<<"\n Orders are empty ";

 }

 }

 void display()

 {

 if(isempty()==0)

 {

 for(int i=front;i!=rear;i=(i+1)%5)

 {

 cout<<q[i]<<"<- ";

 }

 cout<<q[rear];

 }

 else

 {

 cout<<"\n Orders are empty";

 }

 }

 void check()

 {

 int ch;

cout<<"\n\n \* \* \* \* PIZZA PARLOUR \* \* \* \* \n\n";

 cout<<"\n 1. Add a Pizza \n 2. Display the Orders \n 3. Serve a pizza \n 4. Exit \n Enter your choice : ";

 cin>>ch;

switch(ch)

{

 case 1:

 add();

break;

 case 2:

display();

 break;

 case 3:

serve();

 break;

 case 4:

exit(0);

 default:

 cout<<"Invalid choice ";

check();

 }

char ch1;

cout<<"\n Do you want to continue? ";

 cin>>ch1;

 if(ch1=='y'||ch1=='Y')

check();

 }

};

int main()

{

 pizza p1;

 p1.check();

 return 0;

}