**1. WAP for a menu driven calculator.**

**CODE:-**

// Program1

#include<iostream>

using namespace std;

int main(){

int choice,a,b;

char ch;

cout<<"1.WAP for a menu driven calculator\n";

do{

cout<<"enter the your choice"<<"\n"

<<"1:addition \n"

<<"2:subtrection"<<"\n"

<<"3:multiplcation"<<"\n"

<<"4:division \n";

cin >> choice ;

cout<<"enter the first number:\n";

cin >> a ;

cout<<"enter the second number:\n";

cin >> b ;

switch(choice){

case 1:cout<<a+b;break;

case 2:cout<<a-b;break;

case 3:cout<<a\*b;break;

case 4:if(b!=0){

cout<<a/b<<"\n";

}

else{

cout<<"conditinal divion";

}

default:

cout << "worng choice try agin";

}

cout<<"\n Do you wish to continue (Y/N)";

cin >> ch;

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

cout<<"\nending program";

return(0);

}

**2. WAP to find max. and min. of three numbers entered by the user ( USING FUNCTIONS)**

**CODE:-**

// program 2

#include<iostream>

using namespace std;

int max(int n1,int n2,int n3){

if(n1 >= n2 && n1 >= n3){

return(n1);

}

if(n2 >= n1 && n2 >= n3){

return(n2);

}

if(n3 >= n1 && n3 >= n2) {

return(n3);

}

}

int min(int n1,int n2,int n3){

if(n1 <= n2 && n1 <= n3){

return(n1);

}

if(n2 <= n1 && n2 <= n3){

return(n2);

}

if(n3 <= n1 && n3 <= n2) {

return(n3);

}

}

int main(){

int n1, n2, n3;

cout<<"WAP to find max. and min. of three numbers entered by the user ( USING FUNCTIONS)\n";

cout << "Enter three numbers: \n";

cin >> n1 >> n2 >> n3;

cout<<"max number: "<< max(n1,n2,n3)<<"\n";

cout<<"min number: "<<min(n1,n2,n3);

return (0);

}

**3. WAP to find the roots of a quadratic equation (USING FUNCTIONS )**

**CODE:-**

#include <iostream>

#include <cmath>

using namespace std;

int main() {

float a, b, c, x1, x2, discriminant, realPart, imaginaryPart;

cout<<"3.WAP to find the roots of a quadratic equation (USING FUNCTIONS )\n";

cout << "Enter coefficients a, b and c: ";

cin >> a >> b >> c;

discriminant = b\*b - 4\*a\*c;

if (discriminant > 0) {

x1 = (-b + sqrt(discriminant)) / (2\*a);

x2 = (-b - sqrt(discriminant)) / (2\*a);

cout << "Roots are real and different." << endl;

cout << "x1 = " << x1 << endl;

cout << "x2 = " << x2 << endl;

}

else if (discriminant == 0) {

cout << "Roots are real and same." << endl;

x1 = (-b + sqrt(discriminant)) / (2\*a);

cout << "x1 = x2 =" << x1 << endl;

}

else {

realPart = -b/(2\*a);

imaginaryPart =sqrt(-discriminant)/(2\*a);

cout << "Roots are complex and different." << endl;

cout << "x1 = " << realPart << "+" << imaginaryPart << "i" << endl;

cout << "x2 = " << realPart << "-" << imaginaryPart << "i" << endl;

}

return (0);

}

**4. WAP to generate a Fibonacci series till the count entered by the user**

**CODE:-**

// Fibonacci series

//PROGRAM 4

#include <iostream>

using namespace std;

int main ()

{

long number;

cout<<"4. WAP to generate a Fibonacci series till the count entered by the user\n";

cout << "Please type a number: ";

cin >> number;

cout<<"Fibonacci series = <";

int fib1=0,fib2=1,fib3;

for(int i=0;i<number;i++){

fib3=fib1+fib3;

cout<<fib3<<",";

fib1=fib2;

fib2=fib3;

}

cout<<">";

return (0);

}

**5. WAP to solve permutation and combination using recursion**

**CODE:-**

// program 5

#include <iostream>

using namespace std;

long factorial (long a)

{

if (a > 1)

return (a \* factorial (a-1));

else

return (1);

}

int main ()

{

long number,r;

cout<<"5. WAP to solve permutation and combination using recursion\n";

cout << "Please type a number: ";

cin >> number;

cin >> r;

cout<<"Pertmutetion = "<<factorial(number)/factorial(r)<<"\n";

cout<<"Combination = "<<factorial(number)/(factorial(number-r)\*factorial(r));

return (0);

}

**6. WAP to find sum and product of two numbers using recursion**

**CODE:-**

//program6

#include<iostream>

using namespace std;

int sum(int a, int b)

{

if(b==0)

return(a);

else

return(sum(a+1,b-1));

}

int pro(int a,int b)

{

if(b==0)

return(0);

else

return(a+(pro(a,b-1)));

}

int main()

{

int a,b,c,d;

cout<<"6. WAP to find sum and product of two numbers using recursion\n";

cout<<"\nEnter two numbers\n";

cin>>a>>b;

cout<<"\nSum of two numbers is : "<<sum(a,b);

cout<<"\n\nProduct of two numbers is :"<<pro(a,b);

cout<<"\n\nEnding program";

return(0);

}

**7. WAP to find reverse , sum , max. and min. of an array entered by the user ( USING FUNCTIONS)**

**CODE:-**

//program7

#include<iostream>

using namespace std;

void rev\_sum(int n,int \*x)

{

int sum=0;

cout<<"Array in original:\n";

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

cout<<x[i]<<" ";

cout<<"\nArray in reverse:\n";

for(int i=n;i>=1;i--)

{

cout<<x[i]<<" ";

sum+=x[i];

}

cout<<"\nThe sum of the array is: "<<sum;

}

void max\_min(int n,int \*x)

{

int max1=x[1],min1=x[1];

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

{

if(x[i]>max1)

max1=x[i];

if(x[i]<min1)

min1=x[i];

}

cout<<"\nThe maximum of the array is: "<<max1;

cout<<"\nThe minimum of the array is: "<<min1;

}

int main()

{

int n,x[100];

cout<<"7. WAP to find reverse , sum , max. and min. of an array entered by the user ( USING FUNCTIONS)\n";

cout<<"Enter the lenght of array ";

cin>>n;

cout<<"Enter elements in array: ";

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

cin>>x[i];

rev\_sum(n,x);

max\_min(n,x);

return(0);

}

**8. WAP to find reverse and sum of the digits of a number entered by the user**

**CODE:-**

//program8

#include<iostream>

using namespace std;

int main()

{

int num,dig,sum=0,s=0;

cout<<"8. WAP to find reverse and sum of the digits of a number entered by the user";

cout<<"\nEnter the number: ";

cin>>num;

while(num>0)

{

dig=num%10;

sum+=dig;

s=(s\*10)+dig;

num=num/10;

}

cout<<"\nNumber in reverse order is "<<s;

cout<<"\nSum of digit is "<<sum;

return (0);

}

**9. WAP to perform various matrix operations ( sum, difference, product, transpose and trace) matrices entered by the user (USING FUNCTIONS)**

**CODE:-**

//program9

#include<iostream>

using namespace std;

void mat\_read(int x[50][50], int r, int c)

{

int i,j;

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

{

for(j=0;j<c;j++)

cin>>x[i][j];

}

}

void mat\_write(int z[50][50], int r, int c)

{

int i,j;

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

{

for(j=0;j<c;j++)

cout<<z[i][j]<<"\t";

cout<<"\n";

}

}

int mat\_add(int x[50][50], int y[50][50], int z[50][50], int r,int c)

{

int i,j;

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

{

for(j=0;j<c;j++)

z[i][j]=x[i][j]+y[i][j];

}

return 0;

}

int mat\_sub(int x[50][50], int y[50][50], int z[50][50], int r,int c)

{

int i,j;

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

{

for(j=0;j<c;j++)

z[i][j]=x[i][j]-y[i][j];

}

return 0;

}

int mat\_mul(int x[50][50], int y[50][50], int z[50][50], int r1,int c2,int c1)

{

int i,j,k;

for(i=0;i<r1;i++)

for(j=0;j<c2;j++)

{

z[i][j]=0;

for(k=0;k<c1;k++)

z[i][j]=z[i][j]+(x[i][k]\*y[k][j]);

}

return 0;

}

int mat\_transpose(int a[50][50],int z[50][50],int r,int c)

{

int i,j;

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

for(j=0;j<c;j++)

{

z[j][i]=a[i][j];

}

return 0;

}

int mat\_trace(int x[50][50] ,int r, int c)

{ int i,j,trc=0;

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

for(j=0;j<c;j++)

{

if(i==j)

trc=trc+x[i][j];

}

return(trc);

}

int main()

{

int x[50][50], y[50][50],z[50][50],row1,col1,row2,col2,ch,s,s1;

char ch1;

cout<<"9.WAP to perform various matrix operations(sum, difference, product, transpose and trace)matrices entered by the user(USING FUNCTIONS)\n";

do

{

cout<<"Enter no of rows for first matrix:\n";

cin>>row1;

cout<<"\nEnter no of columns for first matrix:\n";

cin>>col1;

cout<<"Enter no of rows for second matrix:\n";

cin>>row2;

cout<<"\nEnter no of columns for second matrix:\n";

cin>>col2;

cout<<"\n Enter elements of first matrix:\n";

mat\_read(x,row1,col1);

cout<<"\n Enter elements of second matrix:\n";

mat\_read(y,row2,col2);

cout<<"\n First matrix entered by user is :\n";

mat\_write(x,row1,col1);

cout<<"\n Second matrix entered by user is :\n";

mat\_write(y,row2,col2);

cout<<"\n\n Now enter your choice of operations to be performed on matrices :\n 1) Addition\n 2)Subtraction \n 3) Multiplication\n";

cout<<" 4) Transpose\n 5) Trace \n";

cin>>ch;

switch(ch)

{

case 1: if(row1==row2 && col1==col2)

{

mat\_add(x,y,z,row1,col1);

cout<<"\nSum of two matrices is :\n";

mat\_write(z,row1,col1);

}

else

cout<<"\nmatrices are not compatible for addition";

break;

case 2: if(row1==row2 && col1==col2)

{

mat\_sub(x,y,z,row1,col1);

cout<<"\nDifference of two matrices is :\n";

mat\_write(z,row1,col1);

}

else

cout<<"\nmatrices are not compatible for subtraction";

break;

case 3: if(col1==row2)

{

mat\_mul(x,y,z,row1,col2,col1);

cout<<"\nMultiplication of two matrices is :\n";

mat\_write(z,row1,col2);

}

else

cout<<"\nmatrices are not compatible for multiplication";

break;

case 4: mat\_transpose(x,z,row1,col1);

cout<<"\nTranspose of first matrix is:\n";

mat\_write(z,col1,row1);

mat\_transpose(y,z,row2,col2);

cout<<"\n \n Transpose of second matrix is :\n";

mat\_write(z,col2,row2);

break;

case 5: if(row1==col1)

{

s=mat\_trace(x,row1,col1);

cout<<"\nTrace of first matrix is :"<<s;

}

else

cout<<"\nTrace of first matrix cant be calculated";

if(row2==col2)

{

s1=mat\_trace(y,row2,col2);

cout<<"\nTrace of second matrix is "<<s1;

}

else

cout<<"\nTrace of second matrix cant be calculated";

break;

default: cout<<"\nWrong choice entered";

break;

}

cout<<"\ndo you wish to continue (y/n)";

cin>>ch1;

}while(ch1=='Y' || ch1=='y');

return (0);

}

**10. WAP to sort an array using Selection sort and display the list after each iteration**

**CODE:-**

//program10

#include<iostream>

using namespace std;

void selection\_sort(int n,int \*a){

int temp;

for (int i=0;i<n-1;i++){

for (int j=i+1;j<n;j++){

if(a[i]>a[j])

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

for(int k=0;k<n;k++){

cout<<a[k]<<" ,";

}

cout<<endl;

}

}

int main(){

int n,a[50];

cout<<"10. WAP to sort an array using Selection sort and display the list after each iteration\n";

cout<<"enter array lengh\n";

cin>>n;

cout<<"\nenter element of array\n";

for(int i=0;i<n;i++){

cin>>a[i];

}

cout<<"\nelement of array after sorting are \n";

for(int i=0;i<n;i++){

cout<<a[i]<<" ,";

}

cout<<endl;

selection\_sort(n,a);

return(0);

}

**12. WAP to take input of 4 integers and check whether they are in AP or in GP , and find the sum of the series**

**CODE:-**

#include<iostream>

#include<cmath>

using namespace std;

class series

{

private:

int n,choice;

float a,b,c,d;

float s1,s2;

char ch;

public:

void input();

void determine();

void sum\_ap();

void sum\_gp();

};

void series::input()

{

cout<<"\nEnter 4 integers\n";

cin>>a>>b>>c>>d;

determine();

}

void series::determine()

{

if((b-a)==(c-b)&&(c-b)==(d-c))

{

cout<<"\nSeries is in AP\n";

cout<<"\nSum of how many integers do you want?\t";

cin>>n;

sum\_ap();

}

else if(b/a==c/b&&c/b==d/c)

{

cout<<"\nSeries is in GP\n";

sum\_gp();

}

else

{

cout<<"\nInvalid series\n";

cout<<"Do you want to input again?\n";

cin>>ch;

if(ch=='y'||ch=='Y')

input();

}

}

void series::sum\_ap()

{

s1=(2\*a+(n-1)\*(b-a))\*n/2;

cout<<"\nSum of the AP series is\n"<<s1;

cout<<"\nDo you want to input again?\n";

cin>>ch;

if(ch=='y'||ch=='Y')

input();

}

void series::sum\_gp()

{

float r;

r=b/a;

if(r<1)

{

cout<<"You want to get the sum of finite series or infinite?\n";

cout<<"\nEnter 1. for finite\n";

cout<<"\nEnter 2. for infinite\n";

cin>>choice;

switch(choice)

{

case 1:

cout<<"\nSum of how many integers do you want?\t";

cin>>n;

s1= a\*((1-pow(r,n))/(1-r));

break;

case 2:

s1= a/(1-r);

break;

}

}

else

{

cout<<"\nGP series is finite";

cout<<"\nSum of how many integers do you want?\t";

cin>>n;

s1= a\*((1-pow(r,n))/(1-r));

}

cout<<"\nSum of the GP series is\t"<<s1;

cout<<"\nDo you want to input again?\n";

cin>>ch;

if(ch=='y'||ch=='Y')

input();

}

int main()

{

cout<<"12. WAP to take input of 4 integers and check whether they are in AP or in GP , and find the sum of the series ";

series sr;

sr.input();

cout<<"Ending Program";

}

**13. WAP to calculate the EOQ and the other values according to the model , also ask to user which model is to be used ( out of first 4 inventory models)**

**CODE:-**

//program13

#include<iostream>

#include<cmath>

using namespace std;

class inventory

{

private:

float dr,oc,ic,uc,pr,shc,base;//dr=demand rate,oc=ordering cost,ic=inventory charge,pr=production rate,shc=shortage cost

public:

void input()

{

cin>>dr;

cin>>oc;

cin>>ic;

cin>>uc;

base=(2\*oc\*dr)/(ic\*uc);

}

float model1()

{

return sqrt(base);

}

float model2()

{

cout<<"\nEnter the production rate of the items. \n(should be greater than the demand rate)"<<endl;

cin>>pr;

return sqrt((base\*pr)/(pr-dr));

}

float model3()

{

cout<<"\nEnter the shortage cost of the items."<<endl;

cin>>shc;

return sqrt(base\*((shc+(ic\*uc))/shc));

}

float model4()

{

cout<<"\nEnter the production rate of the items. \n(should be greater than the demand rate)"<<endl;

cin>>pr;

cout<<"\nEnter the shortage cost of the items."<<endl;

cin>>shc;

return sqrt(base\*(pr/(pr-dr))\*((shc+(ic\*uc))/shc));

}

};

int main()

{

inventory obj;

char ch;

float eoq;

int choice,eoq1;

do

{

cout<<"13. WAP to calculate the EOQ and the other values according to the model , also ask to user which model is to be used ( out of first 4 inventory models)\n";

cout<<"\nEnter the demand rate, ordering cost, inventory carrying charge and unit cost of the items respectively: "<<endl;

obj.input();

cout<<"\n\t\tMENU\t\t";

cout<<"\n1.Production is infinite and shortages are not allowed";

cout<<"\n2.Production is finite and shortages are not allowed";

cout<<"\n3.Production is infinite and shortages are allowed";

cout<<"\n4.Production is finite and shortages are allowed";

cout<<"\n\nFor which model from the above do you wish to compute the economic order quantity?: ";

cin>>choice;

switch(choice)

{

case 1:

eoq=obj.model1();

eoq1=eoq;

cout<<endl<<"Q\* = "<<eoq;

if(fabs(eoq-eoq1)>=0.5)

eoq=eoq1+1;

else

eoq=eoq1;

cout<<"\nThe Economic order quantity for the chosen model is: "<<eoq;

break;

case 2:

eoq=obj.model2();

eoq1=eoq;

cout<<endl<<"Q\* = "<<eoq;

if(fabs(eoq-eoq1)>=0.5)

eoq=eoq1+1;

else

eoq=eoq1;

cout<<"\nThe Economic order quantity for the chosen model is: "<<eoq;

break;

case 3:

eoq=obj.model3();

eoq1=eoq;

cout<<endl<<"Q\* = "<<eoq;

if(fabs(eoq-eoq1)>=0.5)

eoq=eoq1+1;

else

eoq=eoq1;

cout<<"\nThe Economic order quantity for the chosen model is: "<<eoq;

break;

case 4:

eoq=obj.model4();

eoq1=eoq;

cout<<endl<<"Q\* = "<<eoq;

if(eoq-eoq1>=0.5)

eoq=eoq1+1;

else

eoq=eoq1;

cout<<"\nThe Economic order quantity for the chosen model is: "<<eoq;

break;

default:

cout<<"\nWrong choice entered. ";

break;

}

cout<<"\nDo you wish to continue with another model(y/n)? ";

cin>>ch;

}

while (ch=='y'||ch=='Y');

return (0);

}

**14. WAP to solve the three queueing models ( M/M/1 ; M/M/C ; M/M/infinity )**

**CODE:-**

//program14

#include<iostream>

#include<cmath>

using namespace std;

class queue

{

private:

float lambda,mu,rho,rho1,pO,pn,sum,l,lq,n,w,wq,busy;

public:

int c;

int fun(int a) //finding factorial

{

int fact=1;

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

fact\*=i;

return fact;

}

void Server1() //function to implement single server

{

cout<<"\nM|M|1 is the queueing system. Assumptions are :";

cout<<"\n- Number of arrivals follows poisson probability distribution.";

cout<<"\n- Service time follows exponential probability distribution.";

cout<<"\n- There is a single server.";

cout<<"\n- Queue discipline is FcFs.";

cout<<"\n- System capacity is infinite.";

cout<<"\n- The average service rate (mu) is more than the average arrival rate (lambda).";

cout<<"\n\nWhere lambda and mu are constants, dt is incremental element, O(dt)is quantity which is negligible as compared to dt";

cout<<"\n\nEnter the number of arrivals in the system by time t ( n = ) ";

cin>>n;

cout<<"\nEnter the average arrival rate ( lambda = ) ";

cin>>lambda;

cout<<"\nEnter the average service rate ( mu = ) ";

cin>>mu;

// calculating traffic intensity and expected number of busy servers

rho=lambda/mu;

rho1=lambda/mu;

if(rho>=1)

cout<<"\nThe model cannot be computed.";

else

{

/\*calculating expected number of units and expected waiting time of the units

in the system and in the queue\*/

l=(rho)/(1-(rho));

lq=pow(rho,2)/(1-rho);

w=l/lambda;

wq=lq/lambda;

// calculating the expected number of busy servers

busy=floor(rho1);

//calculating the number of customers at time t (n>=0)

pn=pow(rho,n)\*(1-rho);

cout<<"\nOffered load of the system = "<<rho;

cout<<"\nCarried load of the system = "<<rho1;

cout<<"\nTraffic intensity = "<<rho;

cout<<"\nUtilisation factor of the system = "<<rho;

cout<<"\nProbability of n arrivals till time t = "<<pn;

cout<<"\nExpected number of customers waiting in the system = "<<l;

cout<<"\nExpected number of customers waiting in the queue = "<<lq;

cout<<"\nExpected waiting time in the system = "<<w;

cout<<"\nExpected waiting time in the queue = "<<wq;

}

}

void ServerC() //function to implement multiple (c) servers

{

cout<<"\nM|M|C is the queueing system. Assumptions are :";

cout<<"\n- Number of arrivals follows poisson probability distribution.";

cout<<"\n- Service time follows exponential probability distribution.";

cout<<"\n- Number of servers is finite & equal to C.";

cout<<"\n- Queue discipline is FcFs.";

cout<<"\n- System capacity is infinite.";

cout<<"\n- The average service rate (mu) is more than the average arrival rate (lambda).";

cout<<"\n\nWhere lambda and mu are constants, dt is incremental element, O(dt)is quantity which is negligible as compared to dt";

cout<<"\n\nEnter the number of arrivals in the system by time t ( n = ) ";

cin>>n;

cout<<"\nEnter the average arrival rate (number of arrivals per unit time) ( lambda = ) ";

cin>>lambda;

cout<<"\nEnter the average service rate (number of services per unit time) ( mu = ) ";

cin>>mu;

// calculating traffic intensity and expected number of busy servers

rho =lambda/(c\*mu);

rho1=lambda/mu;

if(rho>=1)

cout<<"\nThe model cannot be computed.";

else

{

busy=floor(rho1);

//calculating p(0 arrivals in the system by time t)

for (int j=0;j<c-1;j++)

sum+=pow(rho,j)/fun(j);

pO=1/(sum+(pow(rho\*c,c)))/((1-rho)\*fun(c));

//calculating p(n arrivals in the system by time t)

//for n<c and for n>=c

if (n<c)

pn=(pow(rho1,n)\*pO)/fun(n);

else

pn=(pow(rho1,n)\*pO)/(fun(c)\*pow(c,n-c));

/\*calculating expected number of units and expected waiting time of the units

in the system and in the queue\*/

lq=(pow(rho1,c))\*(lambda\*mu\*pO)/(fun(c-1)\*(c\*mu-lambda)\*(c\*mu-lambda));

wq=lq/lambda;

w=wq+(1/mu);

l=lambda\*w;

cout<<"\nOffered load of the system = "<<rho;

cout<<"\nCarried load of the system = "<<rho1;

cout<<"\nTraffic intensity = "<<rho;

cout<<"\nUtilisation factor of the system = "<<rho;

cout<<"\nProbability of zero arrivals till time t = "<<pO;

cout<<"\nProbability of n arrivals till time t = "<<pn;

cout<<"\nExpected number of customers waiting in the system = "<<l;

cout<<"\nExpected number of customers waiting in the queue = "<<lq;

cout<<"\nExpected waiting time in the system = "<<w;

cout<<"\nExpected waiting time in the queue = "<<wq;

}

}

void Serverinfi() //function to implement infinite servers

{

cout<<"\nM|M|~ is the queueing system. Assumptions are :";

cout<<"\n- Number of arrivals follows poisson probability distribution.";

cout<<"\n- Service time follows exponential probability distribution.";

cout<<"\n- Number of servers is infinite.";

cout<<"\n- Queue discipline is FcFs.";

cout<<"\n- System capacity is infinite.";

cout<<"\n- The average service rate (mu) is more than the average arrival rate (lambda).";

cout<<"\n\nWhere lambda and mu are constants, dt is incremental element, O(dt)is quantity which is negligible as compared to dt";

cout<<"\n\nEnter the number of arrivals in the system by time t ( n = ) ";

cin>>n;

cout<<"\nEnter the average arrival rate (number of arrivals per unit time) ( lambda = ) ";

cin>>lambda;

cout<<"\nEnter the average service rate (number of services per unit time) ( mu = ) ";

cin>>mu;

// calculating traffic intensity and expected number of busy servers

rho=lambda/mu;

rho1=lambda/mu;

/\*calculating expected number of units and expected waiting time of the units

in the system and in the queue\*/

l=rho;

lq=0;

w=l/lambda;

wq=0;

// calculating the expected number of busy servers

busy=floor(rho1);

//calculating the number of customers at time t (n>=0)

pn=(pow(rho,n)\*exp(-rho))/fun(n);

cout<<"\nOffered load of the system = "<<rho;

cout<<"\nCarried load of the system = "<<rho1;

cout<<"\nTraffic intensity = "<<rho;

cout<<"\nUtilisation factor of the system = "<<rho;

cout<<"\nProbability of n arrivals till time t = "<<pn;

cout<<"\nExpected number of customers waiting in the system = "<<l;

cout<<"\nExpected number of customers waiting in the queue = "<<lq;

cout<<"\nExpected waiting time in the system = "<<w;

cout<<"\nExpected waiting time in the system = "<<wq;

}

};

int main()

{

queue a;

char ch;

do

{

cout<<"\n\nEnter the number of servers ( C = ) ";

cin>>a.c;

if(a.c==1)

a.Server1();

else if ((a.c>1)&&(a.c<15))

a.ServerC();

else if ((a.c)>=15)

a.Serverinfi();

cout<<"\nDo you want to continue?(y/n) ";

cin>>ch;

}

while((ch=='y')||(ch=='Y'));

return 0;

}

**11. WAP to sort the list of strings and to search a string and to find the maximum and minimum of the strings**

**CODE:-**

#include<iostream.h>

#include<string.h>

#include<stdio.h>

void input(char a[10][20],int m)

{

int i;

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

{

cout<<"\n The "<<" "<<i+1<<" element is ";

gets(a[i]);

}

cout<<"The strings you entered are:\n";

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

{

cout<<a[i]<<"\n";

}

}

void sort(char a[10][20],int m)

{

char temp[20];

int i,min;

for(i=0;i<m-1;i++)

{

min=i;

for(int j=i+1;j<m;j++)

{

if((strcmp(a[i],a[j]))>0)

min=j;

}

strcpy(temp,a[i]);

strcpy(a[i],a[min]);

strcpy(a[min],temp);

}

cout<<"\n\t the alphabetical order \n";

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

cout<<"\n"<<a[i];

}

void search(char a[10][20],int m)

{

char b[20];

cout<<"\n enter the string to be found\n";

cin>>b;

for(int i=0;i<m-1;i++)

{

if((strcmp(a[i],b))==0)

{

cout<<"\n required string found\n ";

break;

}

}

}

void min(char a[10][20],int m)

{

char c[20];

char min1[10];

strcpy(min1,a[0]);

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

{

if(strcmp(min1,a[i])>0)

{ strcpy(c,a[i]);

strcpy(min1,a[i]);

}

}

cout<<"\n the minimum string is "<< min1;

}

void max(char a[10][20],int m)

{

char c[20];

char max1[10];

strcpy(max1, a[0]);

for(int i=0;i<m;i++)

{

if(strcmp(max1,a[i])<0)

{

strcpy(c,a[i]);

strcpy(max1,a[i]);

}

}

cout<<"\n the maximum string is "<< max1;

}

int main()

{

int choice ,m,i;

char a[10][20];

char ch;

do

{

cout<<" \n enter the no. of strings \n";

cin>>m;

input(a,m);

cout<<"\n choose the choice\n";

cout<<"\n 1: sorting of strings\n";

cout<<"\n 2: searching of strings\n";

cout<<"\n 3: to find minimum string \n";

cout<<"\n 4: to find maximum string\n";

cout<<"\n enter the choice of operation to be performed on the string\n";

cin>>choice;

switch(choice)

{

case 1:

sort(a,m);

br

case 2:

// input(a,m);

search(a,m);

break;

case 3:

// input(a,m);

min(a,m);

break;

case 4:

// input(a,m);

max(a,m);

break;

default:

cout<<"Invalid option";

}

cout<<"\n do you want to continue ";

cin>>ch;

}

while((ch=='y')||(ch=='Y'));

return 0;

}

**15. WAP to implement inheritance by defining classes**

**CODE:-**

#include<iostream.h>

#include<string.h>

#include<stdio.h>

class Media

{

protected:

char title[50], publication[30];

public:

virtual void read()

{

}

virtual void show()

{

}

};

class Book : public Media

{

private:

public:

int iPages;

Book()

{

iPages = 0;

}

void read()

{

fflush(stdin);

cout<<"\nEnter the title of the book: ";

cin>>title;

fflush(stdin);

cout<<"\nEnter the name of the publication: ";

cin>>publication;

while(1)

{

cout<<"\nEnter the number of pages: ";

cin>>iPages;

if(iPages<=0)

cout<<"\nWrong entry! Re-enter.\n";

else

break;

}

cout<<"\nDone.\n";

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

}

void show()

{

cout<<"\nThe title of the book: '"<<title<<"'"<<endl;

cout<<"\nThe name of the publication: "<<publication<<endl;

cout<<"\nNumber of pages: "<<iPages<<endl;

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

}

};

class Tape : public Media

{

private:

double dTime;

public:

Tape()

{

dTime = 0.0;

}

void read()

{

fflush(stdin);

cout<<"\nEnter the title of the tape: ";

cin>>title;

fflush(stdin);

cout<<"\nEnter the name of the publication: ";

cin>>publication;

while(1)

{

cout<<"\nEnter the time span(minutes): ";

cin>>dTime;

if(dTime<=0)

cout<<"\nWrong entry! Re-enter.\n";

else

break;

}

cout<<"\nDone.\n";

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

}

void show()

{

cout<<"\nThe title of the tape: '"<<title<<"'"<<endl;

cout<<"\nThe name of the publication: "<<publication<<endl;

cout<<"\nTime span: "<<dTime<<" mins"<<endl;

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

}

};

int main()

{

cout<<"\n\t BOOK/VIDEO SHOP\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

Book ob;

Tape obj;

int flag=1,ch,s=0;

while(flag)

{

cout<<"\nPRESS 1: NEW ENTRY.\nPRESS 2: SHOW LAST ADDED ENTRY.\nPRESS 0: EXIT.\n\nPRESS: ";

cin>>ch;

if(ch==1)

{

cout<<"\nPRESS 1: ENTER FOR BOOK.\nPRESS 2: ENTER FOR TAPE.\n\nPRESS: ";

cin>>ch;

if(ch==1)

{

s=1;

ob.read();

}

else if(ch==2)

{

s=2;

obj.read();

}

else

cout<<"\nWrong entry!!\n";

}

else if(ch==2)

{

if(s==0)

cout<<"\nNo entries yet.\n";

else if(s==1)

{

cout<<"\nDisplaying last added entry\n........\n";

ob.show();

}

else

{

cout<<"\nDisplaying last added entry\n........\n";

obj.show();

}

}

else if(ch==0)

flag=0;

else

cout<<"\nWrong entry!! Try again.\n";

}

cout<<"\n \*\*\* THANK YOU \*\*\*";

return 0;

}

**16. WAP to implement operator overloading**

**CODE:-**

#include<iostream.h>

class complex

{

float freal;

float fimg;

public:

complex()

{

freal=0.0;

fimg=0.0;

}

complex(float fr, float fi)

{

freal = fr;

fimg = fi;

}

void GetData();

void SetData();

complex operator+(complex &para)

{

complex temp;

temp.freal = freal + para.freal;

temp.fimg = fimg + para.fimg;

return temp;

}

complex operator-(complex &para)

{

complex temp;

temp.freal = freal - para.freal;

temp.fimg = fimg - para.fimg;

return temp;

}

complex operator\*(complex &para)

{

complex temp;

temp.freal = (freal \* para.freal)-(fimg \* para.fimg);

temp.fimg = (freal \* para.fimg)+(fimg \* para.freal);

return temp;

}

complex operator/(complex &para)

{

complex temp;

temp.freal = ((freal \* para.freal) + (fimg \* para.fimg)) / (pow(para.freal,2) + pow(para.fimg,2));

temp.fimg = ((para.freal \* fimg) - (freal \* para.fimg)) / (pow(para.freal,2) + pow(para.fimg,2));

return temp;

}

~complex()

{}

};

void complex :: SetData()

{

float fr;

float fi;

cout<<"Please enter the real part of the complex number : ";

cin>>fr;

cout<<"Please enter the imaginary part of the complex number : ";

cin>>fi;

cout<<endl;

freal=fr;

fimg=fi;

}

void complex :: GetData()

{

if(fimg!=0)

{

if(fimg>0)

cout<<freal<<"+"<<fimg<<"i\n";

else

cout<<freal<<fimg<<"i\n";

}

else

cout<<freal<<endl;

}

int main()

{

char choice;

complex c1,c2,chck,cOutput;

cout<<"\nEnter first complex number\n";

c1.SetData();

cout<<"\nEnter second complex number\n";

c2.SetData();

cout<<"The two entered complex numbers are as follows... \n\n";

c1.GetData();

cout<<endl;

c2.GetData();

while(1)

{

cout<<"\n\nPRESS ' + ' : ADD THE 2 COMPLEX NUMBERS.\n\nPRESS ' - ' : SUBTRACT THE 2 COMPLEX NUMBERS.\n\nPRESS ' \* ' : MULTIPLY THE 2 COMPLEX NUMBERS.\n\nPRESS ' / ' : DIVIDE THE 1ST COMPLEX NUMBER by THE 2ND.\n\nPRESS ' N ' : QUIT PROGRAM.\n";

cout<<"\n... Enter your choice :: ";

cin>>choice;

cout<<endl;

if(choice=='+'||choice=='-'||choice=='\*'||choice=='/')

{

if(choice=='+')

cOutput = c1 + c2;

else if(choice=='-')

cOutput = c1 - c2;

else if(choice=='\*')

cOutput = c1 \* c2;

else if(choice=='/')

cOutput = c1 / c2;

cout<<"\nAfter performing the ' "<<choice<<" ' operation, the resultant complex number = ";

cOutput.GetData();

}

else if(choice=='N' || choice=='n')

break;

else

cout<<"\n## WRONG ENTRY ##\nTry again.\n";

}

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\* THANK YOU \*\*\*\*\*\*\*\*\*\*\*\*";

return 0;

}