1) Write a program in c++ to display message “Welcome to c++ programming” in screen.

* **NAME: BIDHATA PANDEY**
* **ROLL NO: 13**
* **PROGRAM: BE COMPUTER**

#include <iostream>

using namespace std;

int main()

{

cout << "Welcome to C++ programming!" << endl;

return 0;

}



2) Write a program in c++ to read two numbers from user and display their product.

#include<iostream>

using namespace std;

int main()

{

int a,b;

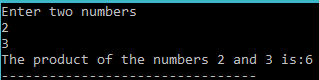
cout<<"Enter two numbers"<<endl;

cin>>a>>b;

cout<<"The product of the numbers "<<a<<" and "<<b<<" is:"<<a\*b;

return 0;

}



3) Write a program to read temperature in centigrade and convert into Fahrenheit.

#include<iostream>

using namespace std;

int main()

{

float C,F;

cout<<"Enter temprature in Centigrade:"<<endl;

cin>>C;

F=((C\*9)+32\*5)/5;

cout<<"The temprature in Fahrenheit is:"<<F;

return 0;

}



4) Write a program to read n numbers from user and find maximum and minimum number among them using concept of array.

#include<iostream>

using namespace std;

int main ()

{

int arr[10], n, i, max, min;

cout << "Enter the size of the array : ";

cin >> n;

cout << "Enter the elements of the array : ";

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

cin >> arr[i];

max = arr[0];

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

{

if (max < arr[i])

max = arr[i];

}

min = arr[0];

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

{

if (min > arr[i])

min = arr[i];

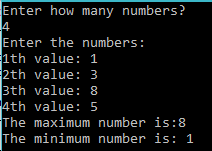
}

cout << "Largest element : " << max;

cout << "Smallest element : " << min;

return 0;

}



5) Write a program in c++ to allocate memory for n fractional numbers using new operator. Store n number in allocated memory by reading from user and display the numbers in ascending order.

#include <iostream>

using namespace std;

class Fraction

{

private:

int i,n,j;

float temp;

float \*p;

public:

void readData();

void displayData1();

void process();

void displayData2();

};

void Fraction::readData()

{

cout<<"How many fractional numbers are there:"<<endl;

cin>>n;

p= new float[n];

cout<<"Enter the fractional numbers: "<<endl;

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

{

cin>>\*(p+i);

}

}

void Fraction::displayData1()

{

cout<<"Values withhin array are:"<<endl;

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

{

cout<<\*(p+i)<<endl;

}

}

void Fraction::process()

{

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

{

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

{

if(\*(p+i)<\*(p+j))

{

temp=\*(p+i);

\*(p+i)=\*(p+j);

\*(p+j)=temp;

}

}

}

}

void Fraction::displayData2()

{

cout<<"Values within array in ascending order are:"<<endl;

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

{

cout<<\*(p+i)<<"\t";

}

}

int main()

{

Fraction fc;

fc.readData();

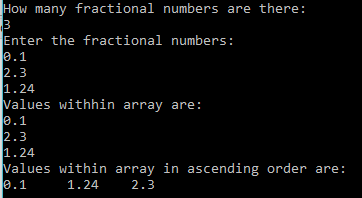
fc.displayData1();

fc.process();

fc.displayData2();

return 0;

}



6) Write a program as your choice to illustrate the use of manipulators for formatting output.

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

int num1=42354, num2=3423;

cout<<endl<<"num1="<<setw(20)<<setfill('\*')<<num1<<endl;

cout<<"num2="<<setfill('$')<<setw(8)<<num2;

return 0;

}



7) Write a program in c++ to define an enumerator color with various names of colors as its members.

#include<iostream>

using namespace std;

int main()

{

enum color

{

Red=1, Green=2, Yellow=3, Violet=4

};

color c;

int j;

cout<<"Please enter the integer(1 to 4)";

cin>>j;

switch(j)

{

case 1:

cout<<"color=red";

break;

case 2:

cout<<"color=green";

break;

case 3:

cout<<"color=yellow";

break;

case 4:

cout<<"color=violet";

break;

}

return 0;

}



8)Write a program to read your full name (i.e. string with blank space) and display in screen.

#include<iostream>

using namespace std;

int main()

{

char st[30];

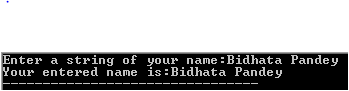
cout<<"Enter a string of your name:";

cin.get(st,20);

cout<<"Your entered name is:"<<st;

return 0;

}



9) Write a program to read a sentence from user and display it in uppercase letters.

#include<iostream>

#include<string.h>

using namespace std;

int main()

{

char sentence[50];

cout<<"Enter a sentence";

cin.get(sentence,50);

strupr(sentence);

cout<<"i am a human :"<<endl<<sentence;

return 0;

}



10) Write a c++ program to find the sum of the digits of a given number.

#include<iostream>

using namespace std;

int main()

{

int num, sum=0;

cout<<"Enter the number:";

cin>>num;

while(num!=0)

{

sum=sum+num%10;

num=num/10;

}

cout<<"The sum of the digits of is"<<sum;

return 0;

}



11) Write a program in c++ to read some number of lines of text until user enters **‘\*’** character and display the entered text in screen.

#include<iostream>

using namespace std;

int main()

{

char str[1000];

cout<<"Enter some text(press \* and enter to end input)"<<endl;

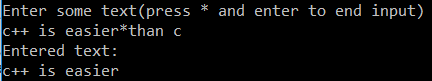
cin.getline(str,1000,'\*');

cout<<"Entered text:"<<endl;

cout<<str;

return 0;

}



12)Write a function Add() which receives three numbers of type float and finds sum of these numbers. Use this function in main() function to calculate sum three numbers read from user.

#include <iostream>

using namespace std;

float Add(float, float, float);

int main()

{

float x, y, z;

cout<<"Enter the three floating point numbers:"<<endl;

cin>>x>>y>>z;

cout<<"The sum of the numbers "<<x<<", "<<y<<" and "<<z<<" is : "<<Add(x,y,z);

return 0;

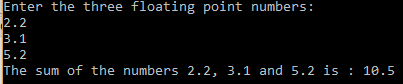
}

float Add(float x, float y, float z)

{

return x+y+z;

}



13)Write a program which has three user-defined functions named ReadData(), AddData( ) and DisplayData.

#include<iostream>

using namespace std;

class Number

{

private:

float a,b,c;

public:

float sum;

void readData();

void addData();

void displayData();

};

void Number :: readData()

{

cout<<"Enter three floating numbers"<< endl;

cin>>a>>b>>c;

}

void Number :: addData()

{

sum=a+b+c;

}

void Number :: displayData()

{

cout<<" The sum of the numbers "<<a<<" , "<<b<<" and "<<c<<" is :"<<sum;

}

int main()

{

Number n;

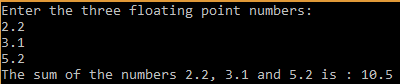
n.readData();

n.addData();

n.displayData();

return 0;

}



14)Write a function named factorial (x) which accepts an int x and returns long value. Write a program to calculate permutation of numbers i.e. nPr using above function factorial ( ) .

#include<iostream>

using namespace std;

long int factorial(int);

void test(int,int);

int main()

{

int n,r;

long int npr;

cout<<"Enter the values of n and r"<<endl;

cin>>n>>r;

void test(n,r);

npr=factorial(n)/factorial(n-r);

cout<<"The value of permutation where n="<<n<<" and r="<<r<<" is :"<<npr;

}

long int factorial( int x)

{

if (x==1)

{

return 1;

}

else

{

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

}

}

void test(int p, int q)

{

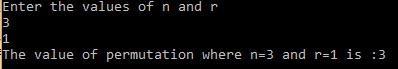
if(p<q)

{

cout<<"Invalid";

}

}



15)Write a function that will evaluate a floating point number to be raised to an integer power i.e. y=mn

#include<iostream>

#include<cmath>

using namespace std;

double calculate(float,int);

int main()

{

int n;

float m,y;

cout<<"Enter the floating number and integer to be raised to it"<<endl;

cin>>m>>n;

y=calculate(m,n);

cout<<"y=m^n="<<m<<"^"<<n<<"= "<<y;

return 0;

}

double calculate(float p, int q)

{

return(pow(p,q));

}

16)Create a function which accepts a character through argument by reference and convert into uppercase if it in lowercase and vice versa. Use this function to in main ( ) program.

#include<iostream>

using namespace std;

char change(char &p);

int main()

{

char m,x;

cout<<"Enter a character:";

cin>>m;

x=change(m);

cout<<"The final output is:"<<x;

}

char change(char &p)

{

if (p>='A' && p<='Z')

{

return(p+32);

}

else if (p>='a' && p<='a')

{

return(p-32);

}

}



17)Write a function named TempConversion ( ) with void return type which accepts temperature value in centigrade as reference variable ( i.e. call by reference) and change -this value into Fahrenheit value. Display converted value in main ( )function.

#include<iostream>

using namespace std;

float temp(float);

int main()

{

double c;

cout << "Enter the temprature in centigrade:"<<endl;

cin >> c;

cout << "The value of temprature in fahrenheit is:"<<temp(c);

}

float temp(float x)

{

return((9/5)\*x/32);

}



18)Write a program to find area of a triangle and square using function overloading.

#include<iostream>

using namespace std;

int Area(int);

float Area(float, float);

int main()

{

int l;

float b,h;

cout<<"Enter the length of square and base and height of triangle:"<<endl;

cin>>l>>b>>h;

cout<<"The area of the square is:"<<Area(l)<<endl;

cout<<"The area of the triangle is:"<<Area(b,h);

return 0;

}

int Area( int x )

{

return (x\*x);

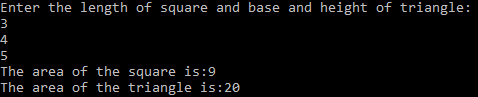
}

float Area( float x, float y)

{

return(x\*y);

}



19)Write a function Average ( ) which calculates average age of two students expressed in int value ( i.e. only year) and write another function having same name Average() which calculates average weight of two functions in main ( ) function.

#include<iostream>

using namespace std;

int Average(int,int);

float Average(float,float);

int main()

{

int a,b;

float c,d;

cout<<"Enter the birth year of the two students:"<<endl;

cin>>a>>b;

cout<<"Enter the weight of the two students:"<<endl;

cin>>c>>d;

cout<<"The average birth year of the students is:"<<Average(a,b)<<endl;

cout<<"The average weight of the two students is:"<<Average(c,d)<<endl;

return 0;

}

int Average(int x, int y)

{

return((x+y)/2);

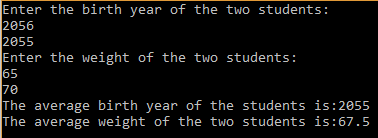
}

float Average(float x, float y)

{

return((x+y)/2);

}



20)Write a inline function to calculate area of a circle from its radius Read radius from user and calculate its area using above function.

#include<iostream>

using namespace std;

inline float Area(float r)

{

return(3.14\*r\*r);

}

int main()

{

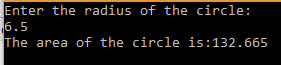
float r;

cout<<"Enter the radius of the circle:"<<endl;

cin>>r;

cout<<"The area of the circle is:"<<Area(r)<<endl;

}



21)Write a function named AreaOfCirc1e ( ) which receives two arguments: radius and PI. Make default argument to PI to use default value 3.14 if it is not supplied in function call otherwise use user-supplied value.

#include<iostream>

using namespace std;

float Area(float , float=3.14);

int main()

{

float r;

cout<<"Enter the radius of the circle:"<<endl;

cin>>r;

cout<<"The area of the circle is: "<<Area(r)<<endl;

return 0;

}

float Area(float r, float PI)

{

return(PI\*r\*r);

}



22)Write a function which displays all even numbers from nl to n2 where nl and n2 are its arguments. Use default value for argument n2 and call this function in main ( ) to display all even numbers from nl to n2.

#include<iostream>

void Even(int,int);

using namespace std;

int main()

{

int n1;

cout<<"Enter the value for n1.";

cin>>n1;

Even(n1,50);

return 0;

}

void Even (int n1, int n2)

{

int i;

cout<<"Even Numbers:"<<endl;

for(i=n1;i<n2;i++)

{

if(i%2==0)

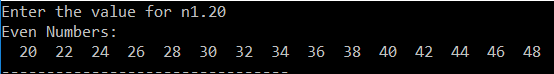
{

cout<<" "<<i;

}

}

}



23)Write a function with prototype int Increase ( int num, int n) to increase num by n. Make n optional and use 10 as default value for it. Use this function in main ( ) function to read num from user and increase it by n.

#include<iostream>

int Increase(int,int);

using namespace std;

int main()

{

int num,s;

cout<<"Enter the number from user:";

cin>>num;

s=Increase(num,10);

cout<<"The number after increase by 10 is:"<<s;

return 0;

}

int Increase(int num,int n)

{

return(num+n);

}



24)Write a program to calculate the area of circle, rectangle and triangle with function named Area ( ) .

#include<iostream>

using namespace std;

int area(int,int);

float area(float);

float area(float,float);

int main()

{

int l,b;

float r,bs,ht;

cout<<"Enter length and breadth of rectangle:";

cin>>l>>b;

cout<<"Enter radius of circle:";

cin>>r;

cout<<"Enter base and height of triangle:";

cin>>bs>>ht;

cout<<"Area of rectangle is"<<" "<<area(l,b)<<endl;

cout<<"Area of circle is"<<" "<<area(r)<<endl;

cout<<"Area of triangle is"<<" "<<area(bs,ht);

}

int area(int l,int b)

{

return(l\*b);

}

float area(float r)

{

return(3.14\*r\*r);

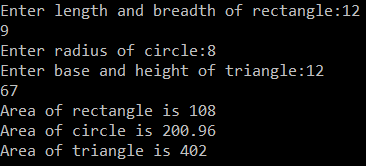
}

float area(float bs,float ht)

{

return((bs\*ht)/2);

}



25)Define a class Rectangle with length and breadth as private members and appropriate member functions to read data members, calculate and display area. Write a main program to read length and breadth of a rectangle and calculate area using object of the class.

#include<iostream>

using namespace std;

class Rectangle

{

private:

float l,b;

public:

float Area;

void readData();

void calculate();

void display();

};

void Rectangle::readData()

{

cout<<"Enter the value of length and breadth of the rectangle: "<<endl;

cin>>l>>b;

}

void Rectangle::calculate()

{

Area=l\*b;

}

void Rectangle::display()

{

cout<<"The area of the rectangle is: "<<Area;

}

int main()

{

Rectangle ob;

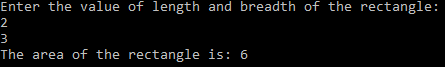
ob.readData();

ob.calculate();

ob.display();

return 0;

}



26)Modify above program to define member function outside of the class.

#include<iostream>

using namespace std;

class Rectangle

{

private:

float l,b;

public:

float Area;

void readData()

{

cout<<"Enter the value of length and breadth of the rectangle: "<<endl;

cin>>l>>b;

}

void calculate()

{

Area=l\*b;

}

void display()

{

cout<<"The area of the rectangle is: "<<Area;

}

};

int main()

{

Rectangle ob;

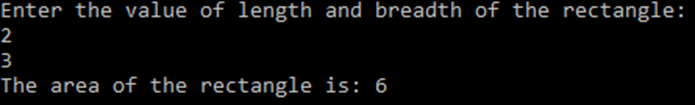
ob.readData();

ob.calculate();

ob.display();

return 0;

}



27)What is the output of the following programs.

#include<iostream>

using namespace std;

class ABC

{

static int a;

public:

void increase()

{

a++;

}

void display()

{

cout<<endl<<"a="<<a;

}

};

int ABC::a;

int main()

{

ABC ob1,ob2,ob3;

ob1.increase();

ob2.increase();

ob3.increase();

ob1.display();

ob2.display();

ob3.display();

return 0;

}



#include<iostream>

using namespace std;

class ABC

{

int a;

public:

void increase()

{

a++;

}

void display()

{

cout<<endl<<"a="<<a;

}

};

int main()

{

ABC ob1,ob2,ob3;

ob1.increase();

ob2.increase();

ob3.increase();

ob1.display();

ob2.display();

ob3.display();

return 0;

}



28)Write a program in C+ which has class Book with data members: book\_name, ISBN, author & price and appropriate function members to read and display data members. Use this class to read record§ often books and display in tabular form.

#include<iostream>

#include<iomanip>

using namespace std;

class Book

{

private:

long int ISBN;

float price;

char bookname[30], author[30];

public:

void readData()

{

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

cin>>bookname;

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

cin>>author;

cout<<"Enter the ISBN code of the book"<<endl;

cin>>ISBN;

cout<<"Enter the price of the book"<<endl;

cin>>price;

}

void displayData();

};

void Book:: displayData()

{

cout<<setw(20)<<bookname<<setw(20)<<author<<setw(20)<<ISBN<<setw(20)<<price<<endl;

}

int main()

{

Book b[10];

cout<<"ENTER BOOK DETAILS:"<<endl;

for( int i=0; i<2; i++)

{

cout<<"Book:"<<"\*"<<i+1<<endl;

b[i].readData();

}

cout<<"\*\*\*\*\*\*\*\*In Tabular form:\*\*\*\*\*\*\*"<<endl;

cout<<"S.N"<<setw(20)<<"Book name"<<setw(20)<<"Author name"<<setw(20)<<"ISBN"<<setw(20)<<"Price"<<endl;

for(int i=0; i<2; i++)

{

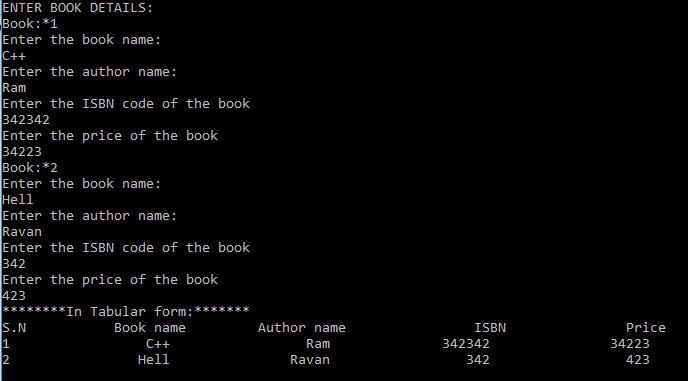
cout<<i+1;

b[i].displayData();

}

return 0;

}



29)Define a class named Distance with meter and cm as private data members and appropriate function members. Use this class to read two objects of the Distance class, add them by passing these two objects to a function member and finally display result object in main() function.

#include<iostream>

using namespace std;

class Distance

{

private:

float m,cm;

public:

float dis;

void readData()

{

cout<<"Enter the value of meter and centimeter"<<endl;

cin>>m>>cm;

}

void calculate()

{

dis=(m+cm/100);

}

void displayData()

{

cout<<"The total distance is: "<<dis<<"m"<<endl;

}

};

int main()

{

Distance d1,d2;

d1.readData();

d1.calculate();

d1.displayData();

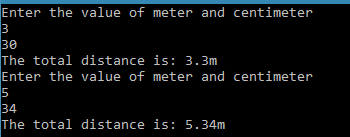
d2.readData();

d2.calculate();

d2.displayData();

return 0;

}



30)Define two classes named ABC and XYZ. Include 'a' as private data member in ABC and 'b' as private member in XYZ. Define an independent(i.e, non-memberfunction) function to calculate mean value of 'a ' and 'b'.

#include<iostream>

using namespace std;

class XYZ;

class ABC

{

private:

int a=10;

friend float Mean(ABC, XYZ);

};

class XYZ

{

private:

int b=20;

friend float Mean(ABC, XYZ);

};

float Mean(ABC x, XYZ y)

{

float mean;

mean=(x.a+y.b)/2;

return mean;

}

int main()

{

ABC p;

XYZ q;

float m;

m=Mean(p,q);

cout<<"Mean="<<m;

return 0;

}



31)Write a program which has two classes as your choice such thatfunctionmembers of a class use private members of another class.

#include<iostream>

using namespace std;

class B;

class A

{

private:

int x=10;

public:

float Mean(B);

};

class B

{

private:

int y=20;

friend class A;

};

float A::Mean (B m)

{

float mean;

mean=x+m.y;

return mean;

}

int main()

{

A a;

B b;

float ans;

ans=a.Mean(b);

cout<<"Mean="<<ans;

return 0;

}



32)Define a class Rectangle with private data members: length and breadth initialize its data members with some fixed values (i.e. say 100 and 200 for length and breadth respectively) using constructor. Write a program to use an object of the class to calculate area of a rectangle.

#include<iostream>

using namespace std;

class Rectangle

{

private:

float l, b;

public:

Rectangle()

{

l=1.1;

b=2.2;

}

float getArea()

{

return l\*b;

}

};

int main()

{

float area;

Rectangle rec ;

area=rec.getArea();

cout<<"The area of the rectangle is:"<<area;

return 0;

}



33)Modify above program to read length and breadth of a rectangle in main() function and supply them in parameterized constructor to initialize its data members.

#include<iostream>

using namespace std;

class Rectangle

{

private:

float length, breadth;

public:

Rectangle(float len, float bre)

{

length=len;

breadth=bre;

}

float getArea()

{

return length\*breadth;

}

};

int main()

{

float area;

Rectangle rec(2.2,1.5);

area=rec.getArea();

cout<<"The area of the rectangle is:"<<area;

return 0;

}



34)Define a class Shape with dim1 and dim2 as private members. Create two constructors of the class, one with one argument and other with two arguments. Write a main() program to define object rectangle with two dimensions and another object square with only one dimension using appropriate constructor and calculate their area.

#include<iostream>

using namespace std;

class Shape

{

private:

float dim1,dim2;

public:

Shape(float V)

{

dim1=dim2=V;

}

Shape(float V1, float V2)

{

dim1=V1;

dim2=V2;

}

float Area()

{

return dim1\*dim2;

}

};

int main()

{

float area;

Shape rectangle(2.2,2.4), square(2.6);

area=rectangle.Area();

cout<<"The area of rectangle is :"<<area<<endl;

area=square.Area();

cout<<"The area of square is :"<<area;

return 0;

}



35)Write a program to illustrate the use of destructor in program for destroying variables created dynamically using new operator.

#include<iostream>

using namespace std;

class Test

{

private:

float \*a;

public:

Test()

{

a = new float(3);

}

void ReadData()

{

cout<<"Enter value of a, b and c:"<<endl;

cin>>\*a>> \*(a+1) >>\*(a+2);

}

float Multiply()

{

return((\*a) \* (\*(a+1)) \* (\*(a+2)));

}

~Test()

{

delete a;

cout<<"Array deleted><"<<endl;

}

};

int main()

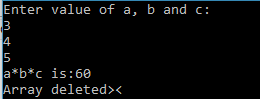
{

Test t;

t.ReadData();

cout<<"a\*b\*c is:"<<t.Multiply()<<endl;

}



36)Modify above program(#2) such that the name of data members must be same as that of arguments (i.e. parameters) in constructor.

#include<iostream>

using namespace std;

class Rectangle

{

private:

float length, breadth;

public:

Rectangle(float length, float breadth)

{

this->length=length;

this->breadth=breadth;

}

float getArea()

{

return length\*breadth;

}

};

int main()

{

float area;

Rectangle rec(2.2,1.5);

area=rec.getArea();

cout<<"The area of the rectangle is:"<<area;

return 0;

}



37)Write a program as your choice to use **‘this’** pointer to return an object of a class from a member function of the class.

#include<iostream>

using namespace std;

class Student

{

private:

int no;

public:

void getno(int no)

{

this->no=no;

}

void Display()

{

cout<<" No"<<no<<endl;

}

};

int main()

{

Student male, female;

male.getno(10);

female.getno(15);

male.Display();

female.Display();

return 0;

}



38) Define a class named Currency with Rs and Paisa as private data members. Overload ‘++’ operator to use it in object of Currency class.

#include<iostream>

using namespace std;

class Currency

{

private:

int Rs,Paisa;

public:

Currency()

{

Rs=Paisa=0;

}

Currency(int Rupee,int Paiso)

{

Rs=Rupee;

Paisa=Paiso;

}

void Display()

{

cout<<"Rupees:"<<Rs<<"\tPaisa:"<<Paisa;

}

Currency operator + (Currency c)

{

Currency temp;

temp.Rs=Rs+c.Rs;

temp.Paisa=Paisa+c.Paisa;

return (temp);

}

};

int main()

{

Currency c1(2,11),c2(3,45),c3;

cout<<"Displaying c1:\n";

c1.Display();

cout<<"\nDisplaying c2:\n";

c2.Display();

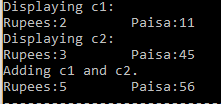
cout<<"\nAdding c1 and c2.\n";

c3=c1+c2;

c3.Display();

return 0;

}



39)Write a program as your choice to overload logical not (i.e. !) operator.

#include<iostream>

using namespace std;

class Number

{

private:

int n;

public:

Number(int x)

{

n=x;

}

void displaynum()

{

cout<<"The value of n is:"<<n<<endl;

}

Number operator ! ()

{

n=!n;

}

};

int main()

{

Number Num(10);

cout<<"Before calling ! operator"<<endl;

Num.displaynum();

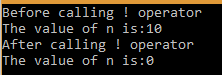
cout<<"After calling ! operator"<<endl;

!Num;

Num.displaynum();

return 0;

}



40) Define a class Distance with data members: m and cm and appropriate function members. Overload binary plus (i.e. ‘+’) operator to use for addition of two objects of the class.

#include<iostream>

using namespace std;

class Distance

{

private:

int m,cm;

public:

Distance()

{

m=cm=0;

}

Distance(int meter,int centi)

{

m=meter;

cm=centi;

}

void DispData()

{

cout<<"Meter:"<<m<<"\tCentimeter:"<<cm<<endl;

}

Distance operator + (Distance d)

{

Distance tp;

tp.m=m+d.m;

tp.cm=cm+d.cm;

if(tp.cm>100)

{

tp.m++;

tp.cm=tp.cm-100;

}

return tp;

}

};

int main()

{

Distance dis1(2,34), dis2(3,90),dis3;

cout<<"First data:\n";

dis1.DispData();

cout<<"Second data:\n";

dis2.DispData();

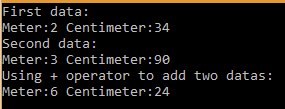
cout<<"Using + operator to add two datas:\n";

dis3=dis1+dis2;

dis3.DispData();

return 0;

}



41)Modify above program #3 to overload less than (i.e. <) operator to check whether one object is less than second or not.

#include<iostream>

using namespace std;

enum Boolean{

f,

t

};

class Distance

{

private:

float m,cm;

public:

Distance()

{

m=cm=0;

}

Distance(int meter,int centi)

{

m=meter;

cm=centi;

}

void DispData()

{

cout<<"Meter:"<<m<<"\tCentimeter:"<<cm<<endl;

}

Boolean operator < (Distance d)

{

float f1=m+(cm/100);

float f2=d.m+(d.cm/100);

return((f1<f2)?t:f);

}

};

int main()

{

Distance d1(1,92), d2(3,90);

cout<<"First data:\n";

d1.DispData();

cout<<"Second data:\n";

d2.DispData();

cout<<"Using < operator to relate two datas:\n";

if(d1<d2)

{

cout<<"d1<d2";

}

else

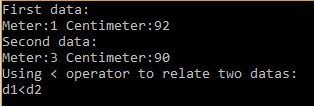
{

cout<<"d1>d2";

}

return 0;

}



42) Write a program to overload equality operator (i.e. ==) operator to check two strings for equality.

#include<iostream>

#include<string.h>

using namespace std;

class String

{

private:

char s[10];

public:

void read()

{

cout<<"Enter a string:";

cin>>s;

}

int operator == (String st)

{

if (strcmp(s,st.s)==0)

{

return 1;

}

else

return 0;

}

};

int main()

{

String st1,st2;

st1.read();

st2.read();

if (st1==st2)

{

cout<<"Strings are equal.";

}

else

cout<<"Strings are not equal.";

}



43) Define a class Box with l, b and h as private data members and appropriate member functions for reading data members and calculate and display volume of a box. Use decrement operator (i.e. --) to decrease object of Box class by one (i.e. decrease each data member by one).

#include<iostream>

using namespace std;

class Box

{

private:

float l,b,h;

public:

void readData()

{

cout<<"Enter value of l,b,h:"<<endl;

cin>>l>>b>>h;

}

float calculate()

{

return(l\*b\*h);

}

void displayData()

{

cout<<"Length:"<<l<<endl;

cout<<"Breadth:"<<b<<endl;

cout<<"Height:"<<h<<endl;

cout<<"The volume is:"<<calculate()<<endl;

}

Box operator --()

{

Box d;

d.l=--l;

d.b=--b;

d.h=--h;

return d;

}

};

int main()

{

Box b1,b2;

b1.readData();

b1.calculate();

b1.displayData();

cout<<"Using -- operator to decrease value of b1"<<endl;

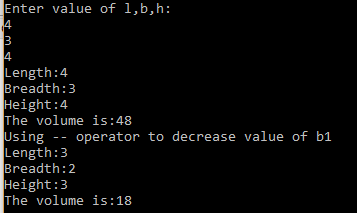
b2=--b1;

b2.calculate();

b2.displayData();

return 0;

}



44) Overload multiplication operator (i.e. \*) using friend function to multiply two matrices.

#include<iostream>

#include<iomanip>

using namespace std;

class matrix

{

public:

int a[3] [3];

matrix()

{

for(int i=0; i<3; i++)

{

for(int j=0; j<3; j++)

{

a[i][j]=0;

}

}

}

void read()

{

cout<<"Enter the value for the matrix:"<<endl;

for(int i=0; i<3; i++)

{

for(int j=0; j<3; j++)

{

cout<<"i"<<i<<j<<"\t";

cin>>a[i][j];

}

}

}

void show()

{

cout<<"The matrix is :"<<endl;

for(int i=0; i<3; i++)

{

for(int j=0; j<3; j++)

{

cout<<setw(4)<<a[i][j];

}

cout<<endl;

}

}

friend matrix operator \* (matrix x, matrix y);

};

matrix operator \* (matrix x, matrix y)

{

matrix c;

for(int i=0; i<3; i++)

{

for(int j=0; j<3; j++)

{

c.a[i][j]=0;

for(int k=0; k<3; k++)

{

c.a[i][j] = c.a[i][j] + x.a[i][k] \* y.a[k][j];

}

}

}

return(c);

}

int main()

{

matrix a,b,c;

a.read();

b.read();

c=a\*b;

a.show();

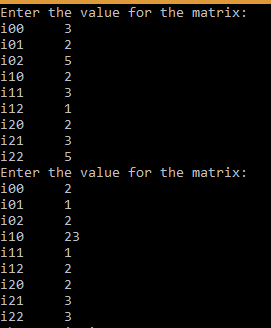
b.show();

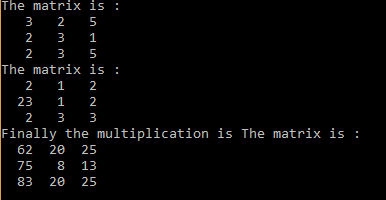
cout<<"Finally the multiplication is ";

c.show();

return 0;

}





45)Overload '-='operator to evaluate the following expression:

A-=B where A and B are objects of a class.

#include <iostream>

using namespace std;

class Example

{

private:

int m,cm;

public:

Example(int meter,int centimeter)

{

m=meter;

cm=centimeter;

}

void display()

{

cout<<m<<"m "<<cm<<"cm";

}

void operator-=(Example e)

{

m=m-e.m;

cm=cm-e.cm;

}

};

int main()

{

Example A(2,90),B(3,50);

A-=B; //same as a.operator-=(b);

A.display();

return 0;

}



46)Overload binary minus operator to overload the following expression.

A=2-B where A and B are the objects of a class.

#include <iostream>

using namespace std;

class Distance

{

private:

int l,b,h;

public:

Distance()

{

l=b=h=0;

}

Distance(int len,int br,int he)

{

l=len;

b=br;

h=he;

}

void show()

{

cout<<l<<"length "<<b<<"breadth "<<h<<"height."<<endl;

}

friend Distance operator-(int,Distance);

};

Distance operator-(int n,Distance d)

{

Distance x;

x.l=n-d.l;

x.b=n-d.b;

x.h=n-d.h;

return x;

}

int main()

{

Distance B(4,34,22),A;

B.show();

A=2-B; //same as operator-(2,B)

A.show();

return 0;

}



47)Define two classed Feet and Meter .Use feet and inch as data members in class feet and m and cm in class Meter. Then, write a program to convert an object of class Feet into an object of another class Meter.

#include<iostream>

using namespace std;

class Meter

{

private:

int m;

float cm;

public:

Meter()

{

m=cm=0;

}

Meter(int meter,float centi)

{

m=meter;

cm=centi;

}

void show()

{

cout<<m<<"m\t"<<cm<<"cm";

}

};

class Feet

{

private:

int ft,in;

public:

Feet()

{

ft=in=0;

}

Feet(int feet, int inch)

{

ft=feet;

in=inch;

}

void show()

{

cout<<ft<<"ft"<<in<<"inch";

}

operator Meter()

{

int m;

float total,cm;

total=(ft+in/12.0)/3.33;

m=(int)total;

cm=(total-m)\*100;

return Meter(m,cm);

}

};

int main()

{

Feet f(3,8);

Meter m;

m=f;

m.show();

return 0;

}



48) Write a program to convert Nepalese currency into Dollar.

#include <iostream>

using namespace std;

class Weight

{

int kg;

float g;

public:

Weight()

{

kg=g=0;

}

Weight(float f)

{

kg=int(f);

g=(f-kg)\*1000;

}

display()

{

cout<<kg<<" kg "<<g<<" gm";

}

};

int main()

{

Weight W;

float F;

cout<<"Enter weight: ";

cin>>F;

W=F; //same as W(F).

cout<<"The kg equivalent object is: "<<endl;

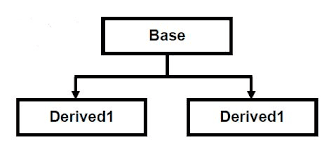
W.display();

return 0;

}



49) Create a class Figure with data members to represent a single and appropriate function members to read data member. Derive two classes Square and Circle from class Figure. Define separate member function in each derived class to calculate and display its corresponding area.



#include<iostream>

using namespace std;

const float PI=3.1416;

class Figure

{

protected:

float l;

public:

void read()

{

cin>>l;

}

};

class Square: public Figure

{

private:

float areasq;

public:

float areaofsq()

{

areasq=l\*l;

return (areasq);

}

};

class Circle: public Figure

{

private:

float areacir;

public:

float areaofcir()

{

areacir=3.14\*l\*l;

return(areacir);

}

};

int main()

{

Circle c;

Square s;

cout<<"Enter radius:"<<endl;

c.read();

cout<<"Enter side for square:"<<endl;

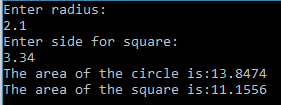
s.read();

cout<<"The area of the circle is:"<<c.areaofcir();

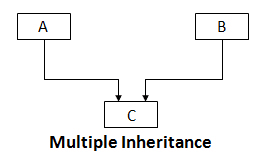
cout<<"\n The area of the square is:"<<s.areaofsq();

return 0;

}



50)Write a program in c++ to illustrate multiple inheritance.



#include<iostream>

using namespace std;

class Length1

{

protected:

float l1;

public:

void read1()

{

cout<<"Enter first length:";

cin>>l1;

}

};

class Length2

{

protected:

float l2;

public:

void read2()

{

cout<<"Enter second length";

cin>>l2;

}

};

class Total: public Length1, public Length2

{

private:

float l;

public:

float total()

{

l=l1+l2;

return (l);

}

};

int main()

{

Total t;

t.read1();

t.read2();

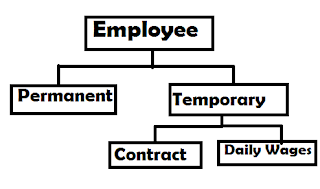
cout<<"The total length is:"<<t.total();

return 0;

}



51)Write a program to represent following classes with following data members.



Employee: Name, Address

Permanent: ID, basic\_salary, grade, insurance\_amount, p\_fund

Temporary: TempEmpID

Contract: Salary, contract\_start\_date, contract\_last\_date

DailyWages:rate\_per\_day

#include <iostream>

#include <string.h>

using namespace std;

class Employee

{

protected:

char Name[30],Address[30];

public:

Employee(char n[],char a[])

{

strcpy(Name,n);

strcpy(Address,a);

}

DisplayEmployee()

{

cout<<endl<<"Name: "<<Name;

cout<<endl<<"Address: "<<Address;

}

};

class Permanent:public Employee

{

protected:

int ID,grade; //ID < 100

float basic\_salary, insurance\_amount, p\_fund;

public:

Permanent(char n[], char a[], int i, float b\_s, int g, float i\_a, float p\_f):Employee(n,a)

{

ID=i;

basic\_salary=b\_s;

grade=g;

insurance\_amount=i\_a;

p\_fund=p\_f;

}

DisplayPermanent()

{

cout<<endl<<"ID: "<<ID;

cout<<endl<<"Basic Salary: "<<basic\_salary;

cout<<endl<<"Grade: "<<grade;

cout<<endl<<"Insurance Amount: "<<insurance\_amount;

cout<<endl<<"Provident Fund: "<<p\_fund;

}

};

class Temporary:public Employee

{

protected:

int TempEmpID; //>=100

public:

Temporary(char n[], char a[], int ti):Employee(n,a)

{

TempEmpID=ti;

}

DisplayTemporary()

{

cout<<endl<<"Temporary Employee ID: "<<TempEmpID;

}

};

class Contract:public Temporary

{

protected:

float Salary;

int contract\_start\_date, contract\_last\_date;

public:

Contract(char n[], char a[], int ti, float s, int cs, int cl):Temporary(n,a,ti)

{

Salary=s;

contract\_start\_date=cs;

contract\_last\_date=cl;

}

DisplayContract()

{

cout<<endl<<"Salary: "<<Salary;

cout<<endl<<"Contract Start Date: "<<contract\_start\_date;

cout<<endl<<"Contract Last Date: "<<contract\_last\_date;

}

};

class DailyWages:public Temporary

{

protected:

float rate\_per\_day;

public:

DailyWages(char n[], char a[], int ti, float r):Temporary(n,a,ti)

{

rate\_per\_day=r;

}

DisplayDailyWages()

{

cout<<endl<<"Rate Per Day: "<<rate\_per\_day;

}

};

int main()

{

Permanent p("Bidhata pandey","Begnastal, Kaski",1,50000.00,7,2200.00,1100.00);

Contract c("Binita bastola","Talchwok, Kaski",100,100000.00,2017,2018);

DailyWages d("namrata Acharya","Kalanki, Kathmandu",150,200);

cout<<"Displaying Permanent Staff:";

p.DisplayEmployee();

p.DisplayPermanent();

cout<<endl<<endl<<"Displaying Contract Staff:";

c.DisplayEmployee();

c.DisplayTemporary();

c.DisplayContract();

cout<<endl<<endl<<"Displaying Daily Wages Staff:";

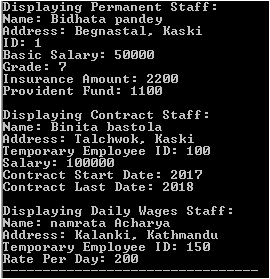
d.DisplayEmployee();

d.DisplayTemporary();

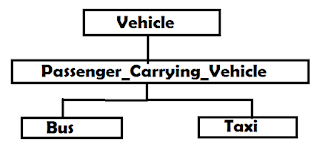
d.DisplayDailyWages();

return 0;

}



52) Create a class Vehicle with data members:VNo, no\_of\_wheel and max\_speed. Derive another class Passenger\_Carrying\_Vehicle with data members no\_of\_passengers. Derive another two classes Bus(with route, fare\_per\_person and helper\_name as private members) and Taxi (with fare\_per\_km as data member). Write a program to use these classes.



#include<iostream>

#include<string.h>

using namespace std;

class Vechile

{

protected:

int VNo,no\_of\_wheel,max\_speed;

public:

Vechile(int vn,int now,int ms)

{

VNo=vn;

no\_of\_wheel=now;

max\_speed=ms;

}

DisplayVechile()

{

cout<<"Vechile Number:"<<VNo;

cout<<"\nNo of Wheels:"<<no\_of\_wheel;

cout<<"\nMaximum speed:"<<max\_speed;

}

};

class Passenger\_Carrying\_Vechile : public Vechile

{

protected:

int num\_of\_passengers;

public:

Passenger\_Carrying\_Vechile(int vn,int now,int ms,int nop ):Vechile(vn,now,ms)

{

num\_of\_passengers=nop;

}

DisplayPassenger\_Carrying\_Vechile()

{

cout<<"\nNo of passengers:"<<num\_of\_passengers;

}

};

class Bus : public Passenger\_Carrying\_Vechile

{

protected:

float fare\_per\_person;

char route[30], Helper\_name[30];

public:

Bus(int vn,int now,int ms,int nop,float fpp,char r[],char h[]):Passenger\_Carrying\_Vechile(vn,now,ms,nop)

{

strcpy(route,r);

strcpy(Helper\_name,h);

fare\_per\_person=fpp;

}

void DisplayBus()

{

cout<<"\nFare per person:"<<fare\_per\_person;

cout<<"\nHelper Name:"<<Helper\_name;

cout<<"\nRoute:"<<route;

}

};

class Taxi : public Passenger\_Carrying\_Vechile

{

protected:

float fare\_per\_km;

public:

Taxi(int vn,int now,int ms,int nop,float fpk):Passenger\_Carrying\_Vechile(vn,now,ms,nop)

{

fare\_per\_km=fpk;

}

DisplayTaxi()

{

cout<<"\nFare per km:"<<fare\_per\_km;

}

};

int main()

{

Bus b(9883,4,150,42,20,"A","Gopal");

Taxi t(8839,4,120,5,50);

cout<<"\nDisplaying Bus:";

b.DisplayVechile();

b.DisplayPassenger\_Carrying\_Vechile();

b.DisplayBus();

cout<<"\n\nDisplaying Taxi:";

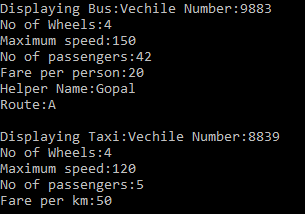
t.DisplayVechile();

t.DisplayPassenger\_Carrying\_Vechile();

t.DisplayTaxi();

return 0;

}



53)Create a class Polygon with data members to represent two dimensions and appropriate member function read its data member. Derive two classes Rectangle and Triangle from above class. Define member function Area() in each derived class to calculate area of corresponding figure. Use base class pointer to call derived class member function Area().

#include<iostream>

using namespace std;

class Polygon

{

protected:

float dim1,dim2;

public:

void readData()

{

cout<<"\nEnter dimension 1:";

cin>>dim1;

cout<<"Enter dimension 2:";

cin>>dim2;

}

};

class Rectangle: public Polygon

{

public:

float Area()

{

return(dim1\*dim2);

}

};

class Triangle: public Polygon

{

public:

float Area()

{

return(0.5\*dim1\*dim2);

}

};

int main()

{

Polygon \*p;

Rectangle r;

cout<<"For Rectangle:";

r.readData();

p=&r;

cout<<"The area of the rectangle is:"<<((Rectangle\*)p)->Area();

Triangle t;

cout<<"\n\nFor Triangle:";

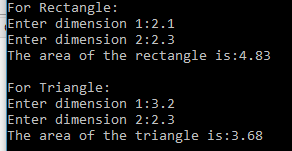
t.readData();

p=&t;

cout<<"The area of the triangle is:"<<((Triangle\*)p)->Area();

return 0;

}



54) Create a program with two classes, one in base and another in child class. Include a function with same name and same signature in each class to illustrate use of function overriding.

#include<iostream>

using namespace std;

class Base

{

protected:

int a,b;

public:

Base(int x, int y)

{

a=x;

b=y;

}

void Display()

{

cout<<"\nYou are in base class.\n";

cout<<"a:"<<a<<"\tb:"<<b;

}

};

class Derived: public Base

{

protected:

int p,q;

public:

Derived(int x,int y,int m,int n): Base(x,y)

{

p=m;

q=n;

}

void Display()

{

cout<<"\nYou are in derived class.\n";

cout<<"p:"<<p<<"\tq:"<<q;

}

};

int main()

{

Derived d(1,2,11,12);

cout<<"Displaying data of derived class:";

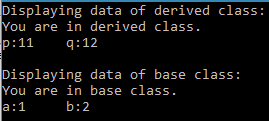
d.Display();

cout<<"\n\nDisplaying data of base class:";

d.Base:: Display();

return 0;

}



55)Create a class Polygon with data members to represent two dimensions and appropriate member function read its data members. Derive two another classes Rectangle and Triangle with member function to calculate and display area. Use these classes to calculate the area of a rectangle and triangle.

#include<iostream>

using namespace std;

class Polygon

{

protected:

float dim1,dim2;

public:

Polygon(int x,int y)

{

dim1=x;

dim2=y;

}

void Display()

{

cout<<"\ndimension1:"<<dim1;

cout<<"\tdimension2:"<<dim2;

}

};

class Rectangle: public Polygon

{

public:

Rectangle(int x,int y):Polygon(x,y)

{

}

float Area()

{

return(dim1\*dim2);

}

};

class Triangle: public Polygon

{

public:

Triangle(int x,int y):Polygon(x,y)

{

}

float Area()

{

return(0.5\*dim1\*dim2);

}

};

int main()

{

Rectangle r(4.4,2.2);

cout<<"For Rectangle:";

r.Display();

cout<<"\nThe area of the rectangle is:"<<r.Area();

Triangle t(3.3,4.8);

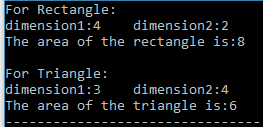
cout<<"\n\nFor Triangle:";

t.Display();

cout<<"\nThe area of the triangle is:"<<t.Area();

return 0;

}



56) Write a program in c++ to illustrate the execution order of constructors and destructors when both the base and derived class has constructors and destructors.

#include<iostream>

using namespace std;

class Polygon

{

protected:

float dim1,dim2;

public:

Polygon(int x,int y)

{

cout<<"I am constructor from polygon.\n";

dim1=x;

dim2=y;

}

void Display()

{

cout<<"\ndimension1:"<<dim1;

cout<<"\tdimension2:"<<dim2;

}

};

class Rectangle: public Polygon

{

public:

Rectangle(int x,int y):Polygon(x,y)

{

cout<<"I am constructor from Rectangle\n";

}

float Area()

{

return(dim1\*dim2);

}

~Rectangle()

{

cout<<"\nFrom Rectangle: I am destructor";

}

};

class Triangle: public Polygon

{

public:

Triangle(int x,int y):Polygon(x,y)

{

cout<<"I am constructor from triangle.\n";

}

float Area()

{

return(0.5\*dim1\*dim2);

}

~Triangle()

{

cout<<"\nFrom Triangle: I am destructor";

}

};

int main()

{

Rectangle r(4.4,2.2);

Triangle t(3.3,4.8);

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

cout<<"For Rectangle:";

r.Display();

cout<<"\nThe area of the rectangle is:"<<r.Area();

cout<<"\n\nFor Triangle:";

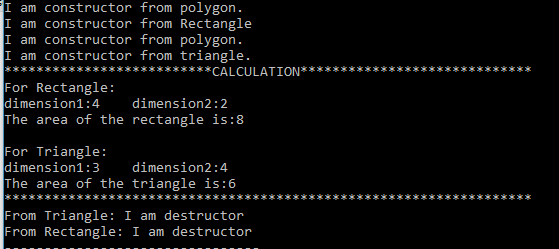
t.Display();

cout<<"\nThe area of the triangle is:"<<t.Area();

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

return 0;

}



57)Write a program to create a class Shape with dim as data member and constructor to initialize its data. Derive two classes Square and Circle from above class. Define a member function FindPerimeter() within Square to find perimeter of a square using dim, defined in base class as side of the square. Similarly, define a member function FindPerimeter() within a class Circle to find Circumference using dim as radius of circle. Implement run time polymorphism using above classes.

#include<iostream>

#define PI 3.1416

using namespace std;

class Shape

{

protected:

float dim;

public:

Shape(float a)

{

dim=a;

}

virtual float FindPerimeter()

{

return(0.0);

}

};

class Circle : public Shape

{

public:

Circle(float a):Shape(a)

{

}

float FindPerimeter()

{

return(PI\* dim\* dim);

}

};

class Square : public Shape

{

public:

Square(float a):Shape(a)

{

}

float FindPerimeter()

{

return(dim\*dim);

}

};

int main()

{

Shape \*s;

Circle c(2.5);

Square sq(3.4);

float c\_area,s\_area;

s=&c;

c\_area=s->FindPerimeter();

s=&sq;

s\_area=s->FindPerimeter();

cout<<"The Area of the circle is:"<<c\_area;

cout<<"\nThe Area of the square is: "<<s\_area;

return 0;

}



58)Write a program in c++ to implement pure virtual function.

#include<iostream>

using namespace std;

class Person

{

public:

virtual void text()=0;

};

class Student: public Person

{

public:

void text()

{

cout<<"I am a student.";

}

};

class Teacher: public Person

{

public:

void text()

{

cout<<"I am a person.";

}

};

class Driver: public Person

{

public:

void text()

{

cout<<"I am a driver.";

}

};

int main()

{

Person \*p[3];

Student s;

Teacher t;

Driver d;

p[0]=&s;

p[1]=&t;

p[2]=&d;

for (int i=0; i<3; i++)

{

p[i]->text();

cout<<endl;

}

return 0;

}



59)Create a class Employee with name and id as data members , ReadData() and DisplayData() as member functions to read and display its data members. Derive two classes : BasicSalary and ExtraAllowanve from class Employee. The class BasicSalary includes basic\_scale and the ExtraAllowance includes bonus, rent and transport as data members. Derive another class Renumeration from the both classes Basic salary and Extraallowance to calculate total salary of an employee. Use these classes to read basic salary, and other allowances to calculate the total salary of an employee.

#include<iostream>

using namespace std;

class Employee

{

protected:

char name[30];

int id;

public:

void ReadData()

{

cout<<"Enter name:";

cin>>name;

cout<<"Enter id:";

cin>>id;

}

void DisplayData()

{

cout<<"Name:"<<name<<endl;

cout<<"ID:"<<id<<endl;

}

};

class BasicSalary: virtual public Employee

{

protected:

float basic\_scale;

public:

void ReadBasic\_scale()

{

cout<<"Enter basic scale:";

cin>>basic\_scale;

}

void DisplayBasic\_scale()

{

cout<<"Basic Scale:\n"<<basic\_scale;

}

};

class ExtraAllowance : virtual public Employee

{

protected:

float bonus, rent, transport;

public:

void ReadExtraAllowance()

{

cout<<"Enter bonus:";

cin>>bonus;

cout<<"Enter rent:";

cin>>rent;

cout<<"Enter transport:";

cin>>transport;

}

void DisplayExtraAllowance()

{

cout<<"Bonus:"<<bonus<<endl;

cout<<"Rent:"<<rent<<endl;

cout<<"Transport"<<transport<<endl;

}

};

class Renumeration: public BasicSalary, public ExtraAllowance

{

protected:

float total;

public:

void readRenumeration()

{

ReadData();

ReadBasic\_scale();

ReadExtraAllowance();

total=basic\_scale+bonus+rent+transport;

}

void DisplayRenumeration()

{

DisplayData();

DisplayBasic\_scale();

DisplayExtraAllowance();

cout<<"Total:"<<total;

}

};

int main()

{

Renumeration r;

cout<<"Reading data\n";

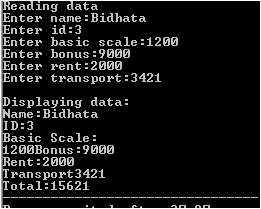
r.readRenumeration();

cout<<"\nDisplaying data:\n";

r.DisplayRenumeration();

return 0;

}



60)Write a program in c++ to use abstract class.

#include<iostream>

using namespace std;

class Hajurba

{

private:

char name1[30];

public:

virtual void read()=0;

virtual void display()=0;

};

class Ba: public Hajurba

{

private:

char name2[30];

public:

void read()

{

cout<<"Enter Ba name:";

cin>>name2;

}

void display()

{

cout<<"Ba Name:"<<name2<<endl;

}

};

class Nati : public Ba

{

private:

char name3[30];

public:

void read()

{

cout<<"Enter Nati name:";

cin>>name3;

}

void display()

{

cout<<"Nati Name:"<<name3<<endl;

}

};

int main()

{

Hajurba \*h;

Ba b;

Nati n;

h=&b;

h->read();

h->display();

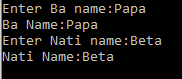
h=&n;

h->read();

h->display();

return 0;

}



61)Write a program in c++ which reads two integers and two fractional numbers from user and calculate quotient for each pair dividing first number by second using single function(i.e. use function template).

#include<iostream>

using namespace std;

template <class A>

A Divide(A x,A y)

{

return(x/y);

}

int main()

{

int i1,i2,iQ;

float f1,f2,fQ;

cout<<"Enter two integers:"<<endl;

cin>>i1>>i2;

cout<<"Enter two fractional numbers:";

cin>>f1>>f2;

iQ=Divide(i1,i2);

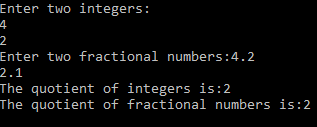
fQ=Divide(f1,f2);

cout<<"The quotient of integers is:"<<iQ<<endl;

cout<<"The quotient of fractional numbers is:"<<fQ<<endl;

return 0;

}



62) Define a single function which returns larger integer if two integers are passed in and larger character if two characters are passed. Use function template to define generic type function.

#include<iostream>

using namespace std;

template <class B>

B Relate(B x,B y)

{

if(x>y)

return x;

else

return y;

}

int main()

{

int i1,i2;

char ch1,ch2;

cout<<"Enter the integers:\n";

cin>>i1>>i2;

cout<<"Enter the characters:\n";

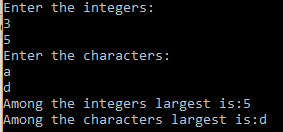
cin>>ch1>>ch2;

cout<<"Among the integers largest is:"<<Relate(i1,i2)<<endl;

cout<<"Among the characters largest is:"<<Relate(ch1,ch2);

return 0;

}



63)Define a class Box with l,b and h as general type data members and appropriate functions to calculate volume of a box. Use above class to calculate volume of two boxes, one having dimensions in integer and other in double.

#include<iostream>

using namespace std;

template <class T>

class Box

{

private:

T l,b,h;

public:

Box()

{

l=b=h=0;

}

Box(T x,T y,T z)

{

l=x;

b=y;

h=z;

}

T area()

{

return(l\*b\*h);

}

};

int main()

{

Box <int> b1(2,3,4);

Box <double> b2(3,4.342,344.34);

cout<<"The total area of int members:"<<b1.area()<<endl;

cout<<"The total area of double members:"<<b2.area();

return 0;

}



64)Write a program to use class template with multiple parameters.

#include<iostream>

using namespace std;

template <class T1,class T2>

class Distance

{

private:

T1 m;

T2 cm;

public:

Distance()

{

m=cm=0;

}

Distance(T1 x,T2 y)

{

m=x;

cm=y;

}

void display()

{

cout<<m<<"m"<<cm<<"cm\n";

}

};

int main()

{

Distance <int,int> d1(2,50);

Distance <int,float> d2(3,50.60);

d1.display();

d2.display();

return 0;

}



65) Write a program to use list standard template library to insert ten numbers and sort them in an order.

#include<iostream>

#include<list>

using namespace std;

int main()

{

list<int> l;

l.push\_back(1);

l.push\_back(10);

l.push\_back(11);

l.push\_back(14);

l.push\_back(51);

l.push\_back(81);

l.push\_back(99);

l.push\_back(43);

l.push\_back(76);

l.push\_back(98);

l.sort();

while(! l.empty())

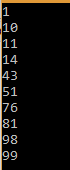
{

cout<<l.front()<<"\n";

l.pop\_front();

}

}



66)Write a program in c++ for handling exception when a number is divided by zero.

#include<iostream>

using namespace std;

int main()

{

float num1,num2,n;

cout<<"Enter two numbers:";

cin>>num1>>num2;

try

{

if(num2!=0)

{

n=num1/num2;

cout<<"The result is:"<<n;

}

else

throw num2;

}

catch(float n)

{

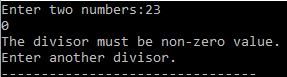
cout<<"The divisor must be non-zero value."<<endl;

cout<<"Enter another divisor.";

}

return 0;

}



67)Write a program to illustrate the use of multiple catch blocks.

#include<iostream>

using namespace std;

int main()

{

char msg[]="Divide by zero";

float num[]={4,54,24,646,763};

int n,counter;

try

{

cout<<"Enter a number for divisor";

cin>>n;

cout<<"How many results do you want?";

cin>>counter;

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

{

if (i>5) throw i;

if(n==0) throw msg;

cout<<"The quotient is:"<<num[i]/n<<endl;

}

}

catch (char str[])

{

cout<<"Exception: "<<str;

}

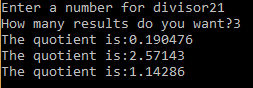
catch ( int index)

{

cout<<"Index :"<<index<<" is out of range.";

}

}



68)Write a program in c++ which can throw multiple types of errors and received by a single catch block.

#include <iostream>

using namespace std;

int main()

{

char msg[]="Divide by Zero error";

int divisor,i;

int num[]={4,21,20,44,141}; //5 data

try

{

cout<<"Enter divisor: ";

cin>>divisor;

for(i=0;i<=5;i++)

{

if(divisor==0)

throw msg;

if(i==5)

throw i;

cout<<endl<<num[i]/divisor;

}

}

catch(...) //single catch block

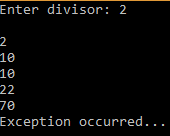
{

cout<<endl<<"Exception occurred...";

}

return 0;

}



69)Write a program to illustrate nested try/catch constructor.

#include <iostream>

using namespace std;

int main()

{

int d;

char msg[]="Divide by zero error.";

try

{

cout<<"Enter divisor: ";

cin>>d;

if(d==0)

throw d;

else

cout<<"Quotient="<<100/d;

}

catch(int d)

{

cout<<msg;

cout<<endl<<"Enter divisor again: ";

cin>>d;

try

{

if(d==0)

throw d;

else

cout<<"Quotient="<<1000/d;

}

catch(int x)

{

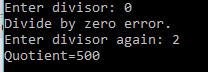
cout<<"OOPs !!! You entered "<<x<<" again..";

}

}

return 0;

}



70)Write a program to read a content of a existing file. If file doesn't exist, throw an error in catch block to display message "File doesn't exist….."

#include<iostream>

#include <fstream>

using namespace std;

int main()

{

char msg[]="File doesn't exist....";

char ch;

ifstream file;

try

{

file.open(shyam.cpp");

if(file.fail())

throw msg;

file>>ch;

while(file.eof()==0)

{

cout<<ch;

file>>ch;

}

}

catch(char str[])

{

cout<<str;

}

file.close();

return 0;

}

