Constructors

1.bank

#include<iostream>

#include<string>

using namespace std;

class bank

{

public:

string name;

int accountnumber,balance;

bank()

{

cin>>name;

cin>>accountnumber;

cin>>balance;

}

void printdetails()

{

cout<<"\n"<<"name="<<name;

cout<<"\n"<<"accountnumber="<<accountnumber;

cout<<"\n"<<"balance="<<balance;

}

~bank()

{

cout<<"\nobject delected";

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*bank details\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

bank jeevan,poshak;

jeevan.printdetails();

poshak.printdetails();

return 0;

}

2.car

#include<iostream>

#include<string>

using namespace std;

class car

{

public:

char company[100];

int melliage,seater,cost;

car()

{

cin>>company;

cin>>melliage;

cin>>seater;

cin>>cost;

}

void detalis()

{

cout<<endl<<"company = "<<company;

cout<<endl<<"melliage ="<<melliage;

cout<<endl<<"seater="<<seater;

cout<<endl<<"cost ="<<cost;

}

~car()

{

cout<<endl<<"object delected";

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*car detalis\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

car company;

company.detalis();

return 0;

}

3. rect

#include<iostream>

#include<string>

using namespace std;

class book

{

public:

string title,author;

int price;

book()

{

cin>>title;

cin>>author;

cin>>price;

}

void printdetails()

{

cout<<endl<<title;

cout<<endl<<author;

cout<<endl<<price;

}

~book()

{

cout<<endl<<"object delected";

}

};

int main()

{

book title;

title.printdetails();

return 0;

}

4.book

#include<iostream>

#include<string>

using namespace std;

class book

{

public:

string title,author;

int price;

book()

{

cin>>title;

cin>>author;

cin>>price;

}

void printdetails()

{

cout<<endl<<"title="<<title;

cout<<endl<<"author="<<author;

cout<<endl<<"price="<<price;

}

~book()

{

cout<<endl<<"object delected";

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*book details\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

book title;

title.printdetails();

return 0;

}

5. student

#include<iostream>

#include<string>

using namespace std;

class student

{

public:

string name;

int regno,fee,rank;

student()

{

cin>>name;

cin>>regno;

cin>>fee;

cin>>rank;

}

void printstudentdetails()

{

cout<<endl<<"name="<<name;

cout<<endl<<"reg.no="<<regno;

cout<<endl<<"fee="<<fee;

cout<<endl<<"rank="<<rank;

}

~student()

{

cout<<endl<<"object deleceted";

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*student details\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

student name;

name.printstudentdetails();

return 0;

}

Inheritance bottom to first

1.

#include<iostream>

#include<string>

using namespace std;

class per{

public:

string name;

int age;

string gender;

per(){

cin>>name;

cin>>age;

cin>>gender;

}

};

class stud:public per{

public:

int rollno;

string cls;

stud(){

cin>>rollno;

cin>>cls;

}

};

class teacher:public per{

public:

string sub;

int salary;

teacher(){

cin>>sub;

cin>>salary;

}

};

int main(){

stud tr;

cout<<tr.name<<"\n";

cout<<tr.age<<"\n";

cout<<tr.gender<<"\n";

cout<<tr.rollno<<"\n";

cout<<tr.cls<<"\n";

teacher ui;

cout<<ui.name<<"\n";

cout<<ui.age<<"\n";

cout<<ui.gender<<"\n";

cout<<ui.sub<<"\n";

cout<<ui.salary;

}

2.

#include<iostream>

#include<string>

using namespace std;

class shape{

public:

int height;

int width;

shape(){

cin>>height;

cin>>width;

}

};

class tri:public shape{

public:

int area()

{

return 0.5\*height\*width;

}

int peri()

{

return height+width;

}

};

class Rectangle:public shape{

public:

int area(){

return height \* width;}

int peri(){

return 2\*height+width;}

};

int main(){

tri t;

cout<<t.height<<"\n";

cout<<t.width<<"\n";

cout<<t.area()<<"\n";

cout<<t.peri()<<"\n";

Rectangle p;

cout<<p.height<<"\n";

cout<<p.width<<"\n";

cout<<p.area()<<"\n";

cout<<p.peri()<<"\n";

return 0;

}

3.

#include<iostream>

#include<string>

using namespace std;

class shape{

public:

int height;

int width;

shape(){

cin>>height;

cin>>width;

}

};

class tri:public shape{

public:

int area()

{

return 0.5\*height\*width;

}

int peri()

{

return height+width;

}

};

class Rectangle:public shape{

public:

int area(){

return height \* width;}

int peri(){

return 2\*height+width;}

};

int main(){

tri t;

cout<<t.height<<"\n";

cout<<t.width<<"\n";

cout<<t.area()<<"\n";

cout<<t.peri()<<"\n";

Rectangle p;

cout<<p.height<<"\n";

cout<<p.width<<"\n";

cout<<p.area()<<"\n";

cout<<p.peri()<<"\n";

return 0;

}

4.

#include<iostream>

#include<string>

using namespace std;

class vehicle{

public:

int year;

int model;

vehicle(){

cin>>year;

cin>>model;

}

};

class car:public vehicle{

public:

float fuel;

int seat;

car(){

cin>>fuel;

cin>>seat;

}

};

class truck:public vehicle{

public:

int payload;

int touning;

truck(){

cin>>payload;

cin>>touning;

}

};

int main(){

car c;

cout<<c.year<<"\n";

cout<<c.model<<"\n";

cout<<c.fuel<<"\n";

cout<<c.seat<<"\n";

truck t;

cout<<t.year<<"\n";

cout<<t.model<<"\n";

cout<<t.payload<<"\n";

cout<<t.touning;

}

5.

#include<iostream>

#include<string>

using namespace std;

class emp{

public:

int id;

string name;

int sal;

emp(){

cin>>id;

cin>>name;

cin>>sal;

}

};

class manag:public emp{

public:

float bonus;

string dept;

manag(){

cin>>bonus;

cin>>dept;

}

};

class engineer:public emp{

public:

string special;

int hours;

engineer(){

cin>>special;

cin>>hours;

}

};

int main(){

manag tr;

cout<<tr.id<<"\n";

cout<<tr.name<<"\n";

cout<<tr.sal<<"\n";

cout<<tr.bonus<<"\n";

cout<<tr.dept<<"\n";

engineer ui;

cout<<ui.id<<"\n";

cout<<ui.name<<"\n";

cout<<ui.sal<<"\n";

cout<<ui.special<<"\n";

cout<<ui.hours;

}

6.

#include <iostream>

using namespace std;

int main()

{

int n=30;

int \*p;

p=&n;

cout<<&n<<endl;

cout<<p<<endl;

cout<<\*p;

}

7.

#include <iostream>

using namespace std;

int main()

{

float n=12.35;

float \*p;

p=&n;

cout<<p<<endl;

cout<<n<<endl;

cout<<\*p<<endl;

}

8.

#include <iostream>

using namespace std;

int main()

{

double n=10.321;

double \*p;

p=&n;

cout<<\*p<<endl;

cout<<p<<endl;

cout<<n;

}

9.

#include <iostream>

using namespace std;

int main()

{

char name='a';

char \*p;

p=&name;

cout<<\*p<<endl;

cout<<p<<endl;

cout<<name;

}

10

#include <iostream>

#include<string>

using namespace std;

int main()

{

string n;

string \*p;

p=&n;

cin>>n;

cout<<\*p<<endl;

cout<<p<<endl;

cout<<n;

}

11.

#include<iostream>

using namespace std;

int main()

{

int a[5];

int i;

cout<<"enter 5 numbers \n";

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

{

cin>> \*(a+i);

}

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

{

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

}

return 0;

}

12

#include<iostream>

using namespace std;

int main()

{

char a[5]={'a','b','c','d','e'};

int n=sizeof(a)/sizeof(a[0]);

int i;

int \*m;

m=&n;

char \*ptr;

ptr=&a[0];

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

{

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

}

}

13.

#include<iostream>

using namespace std;

int main()

{

float a[5];

int i;

cout<<"enter 5 numbers \n";

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

{

cin>> \*(a+i);

}

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

{

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

}

return 0;

}

14.

#include <iostream>

using namespace std;

class MyClass

{

public:

int value;

MyClass(int v):value(v){}

void display()

{

cout<<"Value: "<<value<<endl;

}

};

int main()

{

MyClass obj(42);

MyClass\* objPtr = &obj;

cout << "Accessing object attribute via pointer: "<<objPtr->value<<endl;

objPtr->display();

return 0;

}

15.

#include <iostream>

using namespace std;

int add(int a,int b)

{

return a+b;

}

int subtract(int a,int b)

{

return a-b;

}

int main()

{

int (\*Pointer)(int, int);

Pointer=add;

int result=Pointer(5,3);

cout<<"Result of add function: "<<result<<endl;

Pointer = subtract;

result=Pointer(5,3);

cout<<"Result of subtract function: "<<result<<endl;

return 0;

}

Polymorphism

1.

#include<iostream>

using namespace std;

class animal

{

string name;

int age;

string species;

public:

virtual void speak()

{

cin>>name;

cin>>age;

cin>>species;

cout<<"name="<<name<<endl;

cout<<"age="<<age<<endl;

cout<<"species="<<species<<endl;

}

};

class cat :public animal

{

string colour;

void speak()

{

animal::speak();

cin>>colour;

cout<<"colour="<<colour<<endl;

}

};

class dog :public animal

{

string colour;

int weight;

void speak()

{

animal::speak();

cin>>colour;

cin>>weight;

cout<<"colour="<<colour<<endl;

cout<<"weight="<<weight<<endl;

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*cat detalis\*\*\*\*\*\*\*\*\*\*\*"<<endl;

animal \*ptr;

cat ca;

ptr=&ca;

ptr->speak();

cout<<"\*\*\*\*\*\*\*\*\*\*\*dog details\*\*\*\*\*\*\*\*\*\*"<<endl;

dog d;

ptr=&d;

ptr->speak();

}

2.

#include<iostream>

using namespace std;

class person

{

public:

string name;

int age;

string gender;

virtual void greet()

{

cin>>name;

cin>>age;

cin>>gender;

cout<<"name="<<name<<endl;

cout<<"age="<<age<<endl;

cout<<"gender="<<gender<<endl;

}

};

class student : public person

{

public:

int roll\_number;

string clases;

void greet()

{

person::greet();

cin>>roll\_number;

cin>>clases;

cout<<"Roll\_Number="<<roll\_number<<endl;

cout<<"Class="<<clases<<endl;

}

};

class teacher : public person

{

int salary;

string subject;

void greet()

{

person :: greet();

cin>>salary;

cin>>subject;

cout<<"salary="<<salary<<endl;

cout<<"subject="<<subject<<endl;

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*student details\*\*\*\*\*\*\*\*\*\*"<<endl;

person \*ptr;

student stu;

ptr=&stu;

ptr->greet();

cout<<"\*\*\*\*\*\*\*\*\*teacher details\*\*\*\*\*\*\*\*\*\*"<<endl;

teacher tea;

ptr=&tea;

ptr->greet();

}

3.

#include<iostream>

using namespace std;

class shape

{

public:

int len,bre;

virtual void area()

{

}

};

class rectangle:public shape

{

public:

int are,len,bre;

void area()

{

cin>>len;

cin>>bre;

are=(len\*bre);

cout<<"area of Rectangle="<<are<<endl;

}

};

class circle:public shape

{

public:

float radius,are;

void area()

{

cin>>radius;

are=3.14\*radius\*radius;

cout<<"area of circle="<<are<<endl;

}

};

int main()

{

shape \*ptr;

rectangle rec;

ptr=&rec;

ptr->area();

circle cr;

ptr=&cr;

ptr->area();

}

4.

#include<iostream>

using namespace std;

class vehicle

{

public:

string model;

int year;

virtual void drive()

{

cin>>model;

cin>>year;

cout<<"model="<<model<<endl;

cout<<"year="<<year<<endl;

}

};

class car:public vehicle

{

public:

int seater,fuel;

void drive()

{

vehicle::drive();

cin>>seater;

cin>>fuel;

cout<<"seater="<<seater<<endl;

cout<<"fuel="<<fuel<<endl;

}

};

class truck:public vehicle

{

public:

int payload;

int towing\_capacity;

void drive()

{

vehicle::drive();

cin>>payload;

cin>>towing\_capacity;

cout<<"payload="<<payload<<endl;

cout<<"towing\_capacity="<<towing\_capacity<<endl;

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*car details\*\*\*\*\*\*\*"<<endl;

vehicle \*ptr;

car ca;

ptr=&ca;

ptr->drive();

cout<<"\*\*\*\*\*\*\*truck details\*\*\*\*\*\*\*"<<endl;

truck tru;

ptr=&tru;

ptr->drive();

}

5.

#include<iostream>

using namespace std;

class animal

{

public:

string name;

string species;

virtual void move()

{

cin>>name;

cin>>species;

cout<<"name="<<name<<endl;

cout<<"species="<<species<<endl;

}

};

class bird:public animal

{

public:

string colour;

void move()

{

animal::move();

cin>>colour;

cout<<"colour="<<colour<<endl;

}

};

class fish:public animal

{

public:

string weight;

void move()

{

animal::move();

cin>>weight;

cout<<"weight="<<weight<<endl;

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*bird\*\*\*\*\*\*\*\*\*\*"<<endl;

animal \*ptr;

bird bi;

ptr=&bi;

ptr->move();

cout<<"\*\*\*\*\*\*\*\*\*fish\*\*\*\*\*\*\*\*\*\*"<<endl;

fish fi;

ptr=&fi;

ptr->move();

}

6.

#include<iostream>

using namespace std;

class person

{

public:

string name;

int id;

int salary;

virtual void work()

{

cin>>name;

cin>>id;

cin>>salary;

cout<<"name="<<name<<endl;

cout<<"id="<<id<<endl;

cout<<"salary="<<salary<<endl;

}

};

class employee:public person

{

public:

string specialty;

int hours;

void work()

{

person::work();

cin>>specialty;

cin>>hours;

cout<<"specialty="<<specialty<<endl;

cout<<"hours="<<hours<<endl;

}

};

class manager:public person

{

public:

string deperament;

int bonous;

void work()

{

person::work();

cin>>deperament;

cin>>bonous;

cout<<"deperament="<<deperament<<endl;

cout<<"bonous="<<bonous<<endl;

}

};

int main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*employee details\*\*\*\*\*\*\*\*"<<endl;

person \*ptr;

employee emp;

ptr=&emp;

ptr->work();

cout<<"\*\*\*\*\*\*\*\*\*manager details\*\*\*\*\*\*\*\*\*"<<endl;

manager man;

ptr=&man;

ptr->work();

}

Exception bottom to first

1.

#include<iostream>

using namespace std;

int main()

{

int n,i,f=1;

cout<<"enter number";

cin>>n;

try

{

if(n<=0)

{

throw(n);

}

else

{

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

{

f=f\*i;

}

cout<<f<<endl;

}

}

catch(n)

{

cout<<"factorial error";

}

2.

#include<iostream>

using namespace std;

int main()

{

int b,d;

float a;

cout<<"enter a number \n";

cin>>a>>b;

try

{

if(b==0)

{

throw(b);

}

else

{

throw(a);

}

}

catch(int b)

{

cout<<"division is not possible"<<endl;

}

catch(float a)

{

d=a/b;

cout<<d;

}

}

3.

#include<iostream>

using namespace std;

int main()

{

int a[5],n,i;

cout<<"enter size array \n";

cin>>n;

try

{

if(n>5)

{

throw(n);

}

else

{

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

{

cout<<"enter a number \n";

cin>>a[i];

}

}

}

catch(int n)

{

cout<<"array limit existed";

}

}

6.

#include<iostream>

using namespace std;

int main()

{

int a,b,d;

cout<<"enter no \n";

cin>>a>>b;

try

{

if(b==0)

{

throw(b);

}

else

{

d=a/b;

cout<<d;

}

}

catch(int b)

{

cout<<"division error";

}

}

7.

#include<iostream>

using namespace std;

int main()

{

int n;

cout<<"enter no \n";

cin>>n;

try

{

if(n==0)

{

throw(n);

}

else if(n>0)

{

cout<<"positive no";

}

else

{

cout<<"negative no";

}

}

catch(int n)

{

cout<<"number doesn't exist";

}

}

Operator overloading bottom to first

1.

#include<iostream>

using namespace std;

class incre

{

public:

int i;

incre(int x)

{

i=x;

}

int operator ++()

{

return ++i;

}

};

int main()

{

incre i(10);

cout<<++i;

return 0;

}

2.

// C++ program to demonstrate

// Operator Overloading or

// Compile-Time Polymorphism

#include <iostream>

using namespace std;

class Complex {

private:

int real, imag;

public:

Complex(int r = 0, int i = 0)

{

real = r;

imag = i;

}

// This is automatically called

// when '+' is used with between

// two Complex objects

Complex operator+(Complex const& obj)

{

Complex res;

res.real = real + obj.real;

res.imag = imag + obj.imag;

return res;

}

void print() { cout << real << " + " << imag <<"i"<< endl; }

};

// Driver code

int main()

{

Complex c1(10, 5), c2(2, 4);

// An example call to "operator+"

Complex c3 = c1 + c2;

c3.print();

}

4.

#include<iostream>

using namespace std;

class Equal

{

public:

int i;

Equal() //constructer

{}

Equal(int x)

{

i=x;

}

int operator ==(Equal a){

return i==a.i;

}

};

int main()

{

Equal a1(10),a2(20);

if(a1==a2){

cout<<"equal";}

else

{

cout<<"not equal";

}

return 0;

}

6.

// CPP program to illustrate overloading the

// [ ] operator

#include <iostream>

using namespace std;

class overload {

int a[3];

public:

overload(int i, int j, int k)

{

a[0] = i;

a[1] = j;

a[2] = k;

}

int operator[](int i)

{

return a[i];

}

};

int main()

{

overload ob(1, 20, 3);

cout << ob[1]<<endl; // displays 2

cout<< ob[2];

return (0);

}

7.

#include<iostream>

using namespace std;

class dist

{

public:

int feet=0;

int inches=0;

dist (){}

dist (int f,int i)

{

feet=f;

inches=i;

}

dist operator()(int a,int b,int c)

{

dist d;

d.feet=a+c+10;

d.inches=b+c+100;

return d;

}

void printdetails()

{

cout<<"feet"<<feet<<"\n"<<"inches="<<inches;

}

};

int main()

{

dist d1(11,9);

cout<<"first diatance"<<endl;

d1.printdetails();

dist d2=d1(12,5,10);

cout<<endl<<"second distance";

d2.printdetails();

}

8.

#include<iostream>

using namespace std;

class Sub

{

public:

int i;

Sub(int x)

{

i=x;

}

int operator -(Sub a)

{

return i-a.i;

}

};

int main()

{

Sub a1(20),a2(30);

cout<<a1-a2;

return 0;

}

9.

#include<iostream>

using namespace std;

class addition{

public:

int i(int a,int b)

{

return a+b;

}

float f(float a,float b)

{

return a+b;

}

};

int main()

{

addition a;

cout<<a.i(5,10)<<endl;

cout<<a.f(10.25,35.25);

}

9.

#include<iostream>

using namespace std;

class cal

{

public:

int add(int a,int b)

{

return a+b;

}

int equal(int a,int b)

{

return a=b;

}

};

int main()

{

cal c1;

cout<<c1.add(10,5);

cout<<"\n"<<c1.equal(10,20);

return 0;

}

10

#include<iostream>

using namespace std;

class Max

{

public:

int i;

Max(){}

Max(int x)

{

i=x;

}

int operator > (Max a)

{

return i>a.i;

}

};

int main()

{

Max a1(10),a2(20);

if(a1>a2)

{

cout<<"a is bigger";

}

else

{

cout<<"a2 is bigger";

}

return 0;

}

13.

#include<iostream>

using namespace std;

class area

{

int a;

int l,b;

double r;

public:

void area\_square(int);

void area\_rectangle(int,int);

void area\_circle(double);

};

void area::area\_square(int x)

{

a=x;

int result=a\*a;

cout<<"area of square="<<result<<endl;

}

void area::area\_rectangle(int x,int y)

{

l=x;

b=y;

int result=l\*b;

cout<<"area of rectangle="<<result<<endl;

}

void area::area\_circle(double x)

{

r=x;

double result=3.14\*r\*r;

cout<<"area of circle="<<result<<endl;

}

int main()

{

area a1;

a1.area\_square(4);

a1.area\_rectangle(4,6);

a1.area\_circle(2);

}

Absolute

#include<iostream>

using namespace std;

int abso(int a)

{

int x=abs(a);

return x;

}

double abso(double b)

{

double y=abs(b);

return y;

}

int main()

{

int a;

double b;

cout<<"enter a no \n";

cin>>a>>b;

cout<<abs(a)<<endl;

cout<<abs(b);

}

Power

#include<iostream>

#include<math.h>

using namespace std;

int main()

{

int a,b;

float c,d;

cout<<"ENTER TWO INTEGERS AND TWO DECIMAL NUMBERS : "<<endl;

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

cout<<"POWER OF INTEGERS IS "<<power(a,b)<<endl;

cout<<"POWER OF FLOATING NUMBERS IS "<<power(c,d)<<endl;

}

power(int a, int b)

{

int p;

p=a\*\*b;

return p;

}

power(float a,float b)

{

int w;

w=a\*\*b;

return w;

}

Factorial

#include<iostream>

using namespace std;

int main()

{

int m;

double n;

cout<<"\nENTER AN INTEGER : ";

cin>>m;

cout<<"\nFACTORIAL OF INTEGER IS "<<abs(m);

cout<<"\nENTER A FLOATING NUMBER : ";

cin>>n;

cout<<"\nFACTORIAL OF FLOATING NUMBER IS "<<abs(n);

}

int abs(int m)

{

int i,f=1;

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

{

f=f\*i;

}

return f;

}

double abs(double n)

{

int i,f=1;

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

{

f=f\*i;

}

return f;

}