single level inheritance

class circle{

public:

int radius;

float ar;

circle()

{

radius=0;

}

circle(int r)

{

radius=r;

}

void area()

{

ar=3.14\*radius\*radius;

}

};

class showarea: public circle

{

public:

showarea():circle()

{

}

showarea(int r): circle(r)

{

}

void show()

{

cout<<ar;

}

};

int main()

{

showarea cir(5);

cir.area();

cir.show();

return 0;

}

🔽️MULITPLE INHERITANCE

#include<iostream>

using namespace std;

class A{

public:

void func(){

cout<<" i am in class A"<<endl;

}

};

class B{

public:

void func(){

cout<<"i am in class B"<<endl;

}

};

class c : public A,public B{

};

int main()

{

c obj;

obj.A::func();

obj.B::func();

return 0;

}

🔽️MULTILEVEL INHEIRTANCE

class circle{

public:

int radius;

circle()

{

radius=0;

}

circle(int r)

{

radius=r;

}

};

class areacalculate: public circle{

public :

float ar;

areacalculate():circle()

{

}

areacalculate(int r):circle(r)

{

}

void area()

{

ar=3.14\*radius\*radius;

}

}

class showarea: public areacalculate

{

public:

showarea():areacalulate()

{

}

showarea(int r): areacalculate(r)

{

}

void show()

{

cout<<ar;

}

};

int main()

{

showarea cir(5);

cir.area();

cir.show();

return 0;

}

hybird inheritance

class A {

public:

void displayA() {

cout << "This is class A" << endl;

}

};

class B : public A {

public:

void displayB() {

cout << "This is class B" << endl;

}

};

class C : public A {

public:

void displayC() {

cout << "This is class C" << endl;

}

};

class D : public B, public C {

public:

void displayD() {

cout << "This is class D" << endl;

}

};

int main() {

B objB;

C objC;

D objD;

objB.displayA();

objB.displayB();

objC.displayA();

objC.displayC();

objD.displayA();

objD.displayB();

objD.displayC();

objD.displayD();

return 0;

}

hierarical inheritance

#include <iostream>

using namespace std;

class A {

public:

void displayA() {

cout << "This is class A" << endl;

}

};

class B : public A {

public:

void displayB() {

cout << "This is class B" << endl;

}

};

class C : public A {

public:

void displayC() {

cout << "This is class C" << endl;

}

};

int main() {

B objB;

C objC;

objB.displayA();

objB.displayB();

objC.displayA();

objC.displayC();

return 0;

}