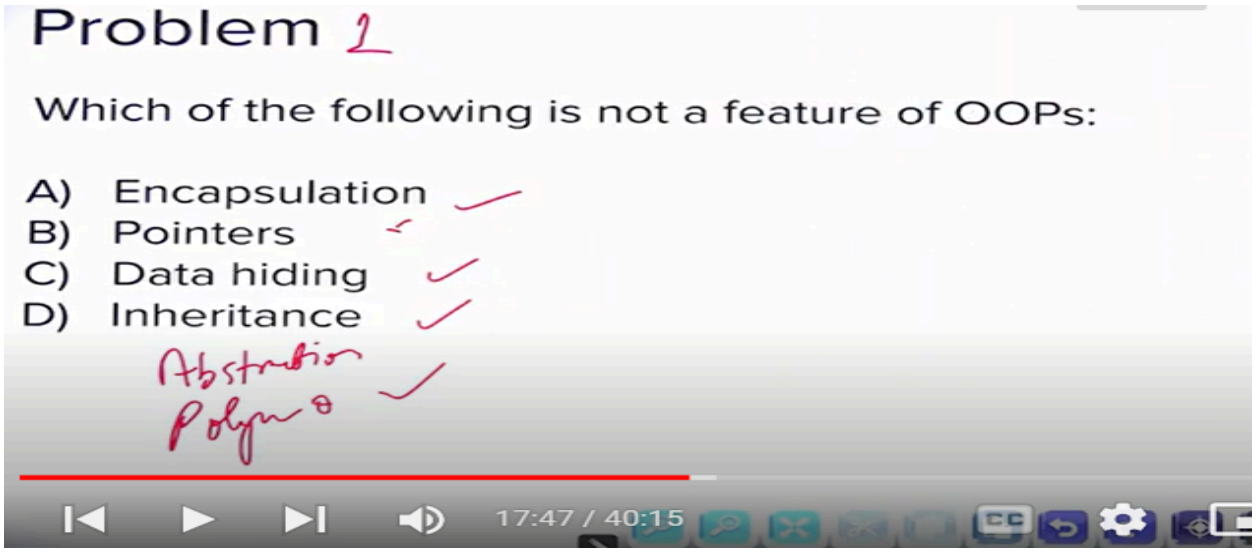
50 Practice Problems in C++ OOPs - Part I | L:29 | C++

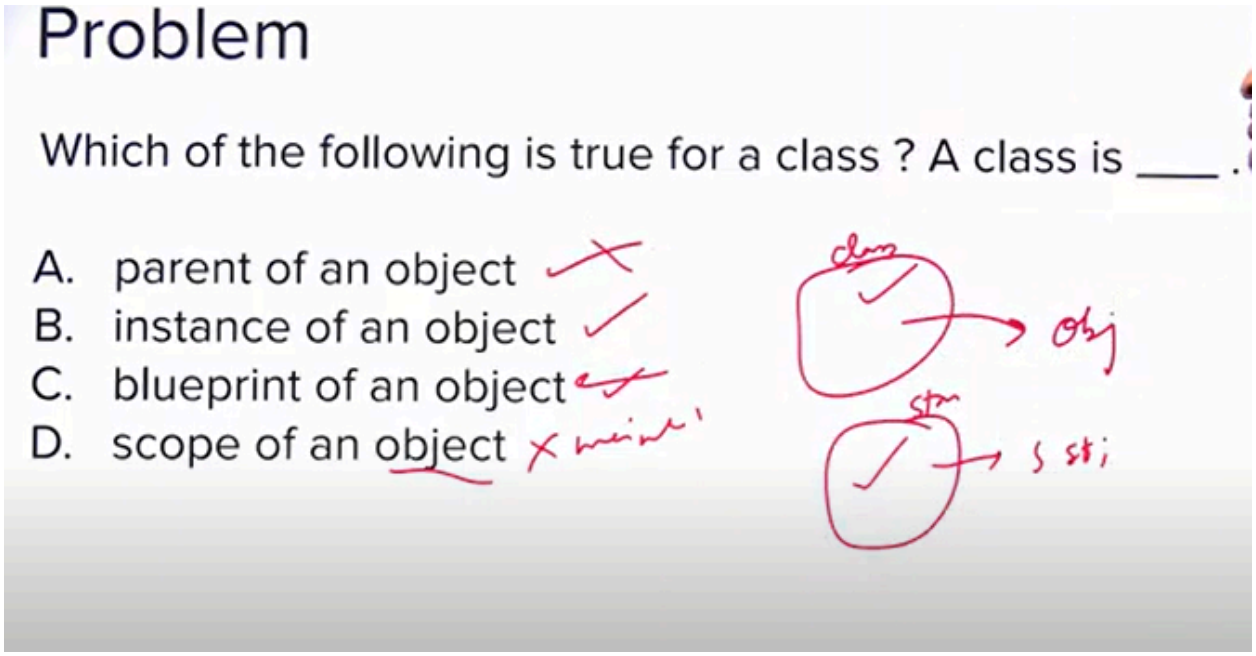
=-==========

Q1:



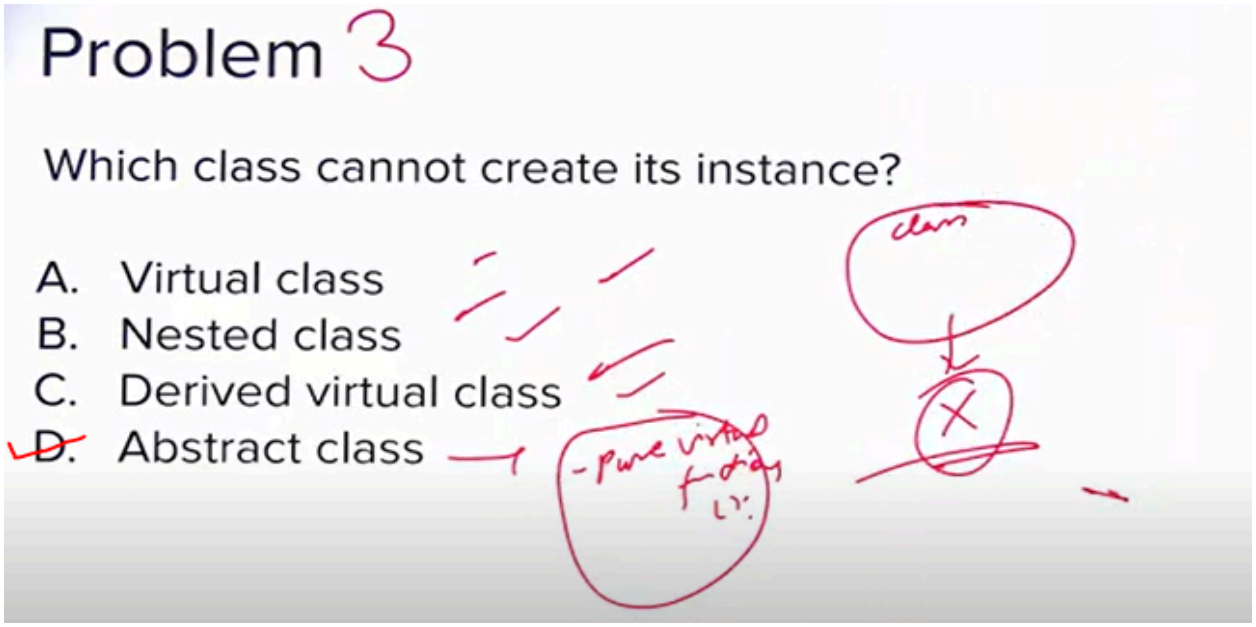
Encapsulation,data hiding,inheritance,abstraction,polymorphism are the features of oops.
pointers===>not the features of oops

Q2:

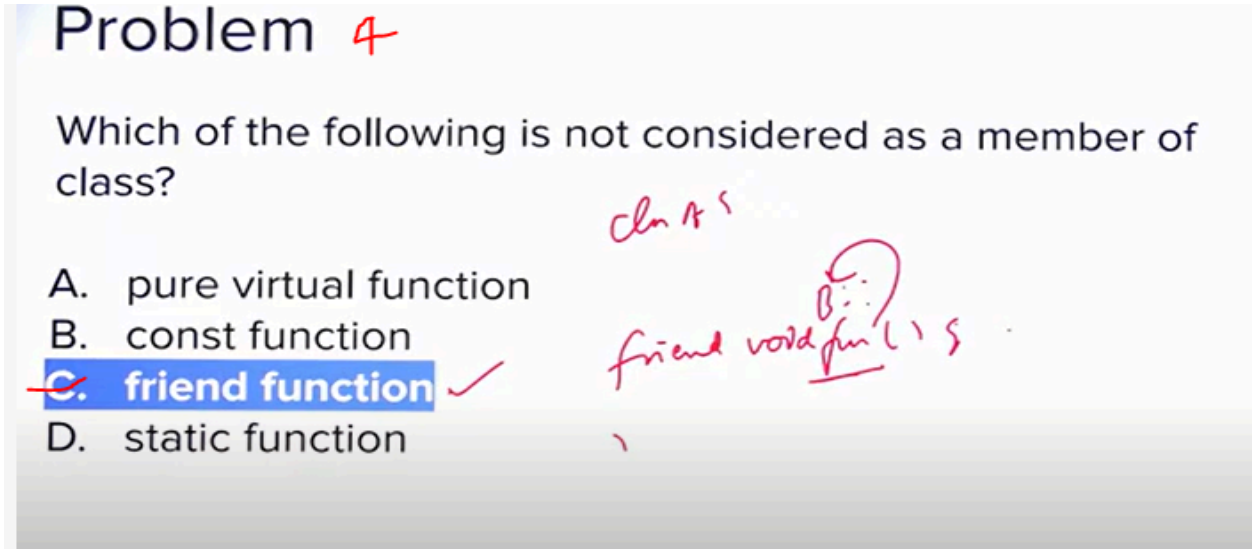


Class is the blueprint of the object.
Objects are instances of the class.

Q3:

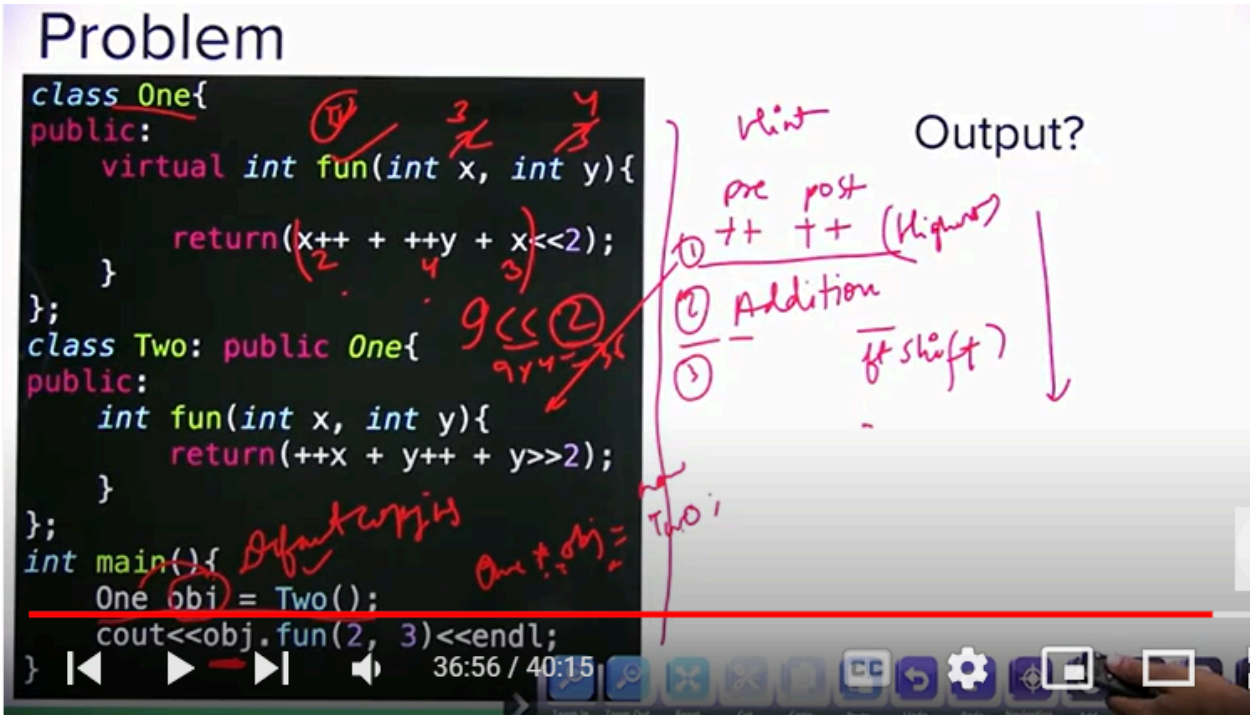


Q4:



friend() function is not a member of the class.
Static function is the member of the class only,not the object.yah hr object ke liye same hota hai.
Constant functions are nothing but the every parameter of the function are constant ,we can not alter any attribute of the function.

Q5:

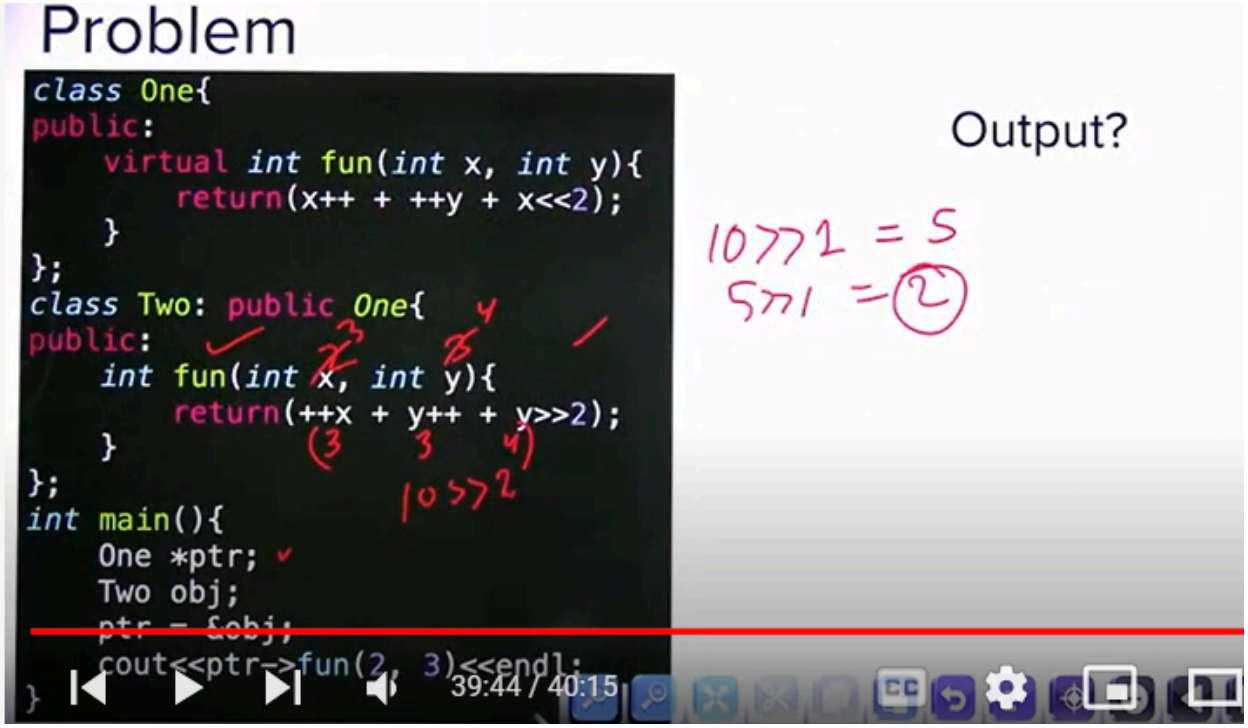


One obj=Two(); ===>
Here we are not creating a pointer to an object.
Here we are creating an object of the base class and we are instantiating it with the object of the derived class.
This is **default copying**,it does not have any pointer and address.
Here overriding will not take place.

One *obj=new Two(); ===> here we are creating the pointer of the base class and we are assigning the address of the derived class object.
Here overriding and everything come into picture.

Note:
+ Have higher priority than <<
2 + 4 + 3 << 2
⇒ (2 + 4 + 3)<<2
⇒ 9<<2
⇒ 36

Q6:



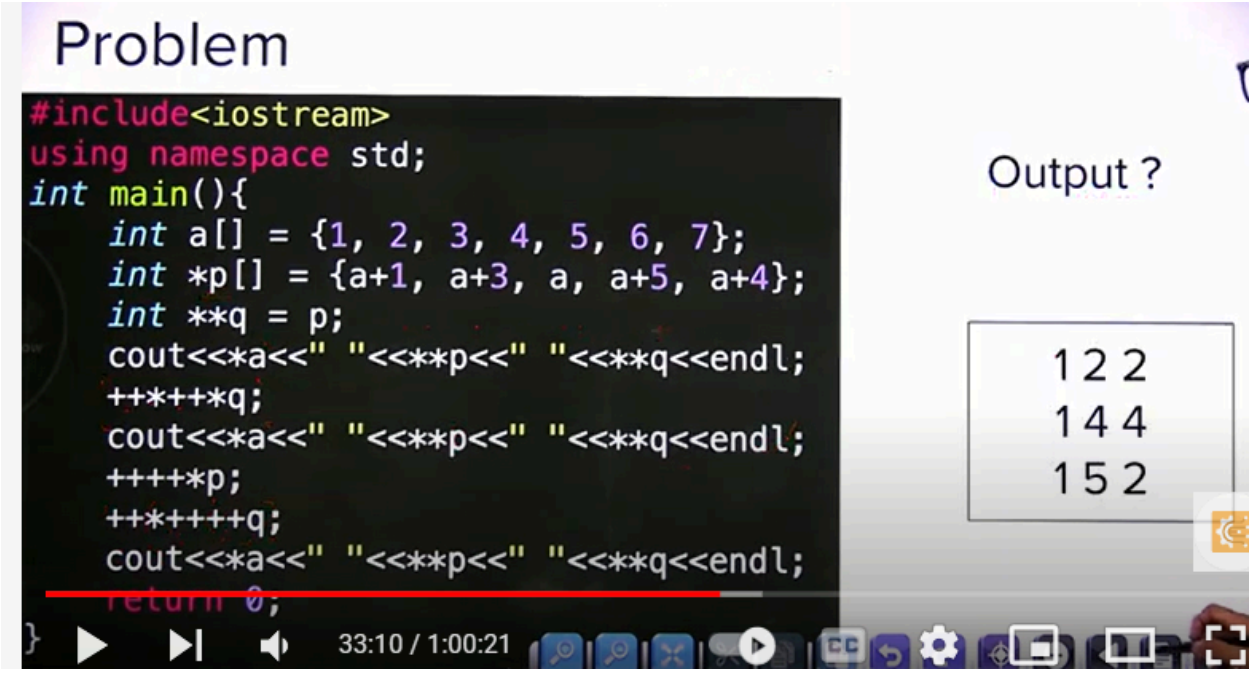
Here we are using the pointer of base class and that pointer are holding the address of the object of the child class.
So here method overriding will be used.

==========

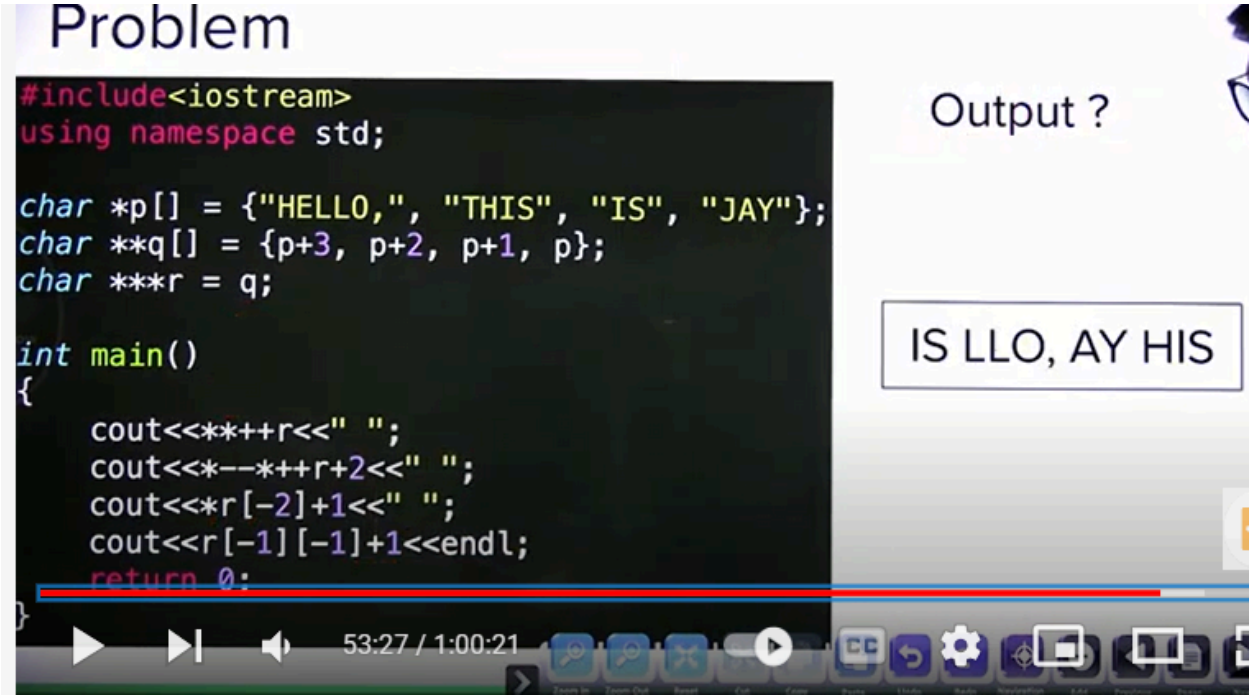
50 Practice Problems in C++ OOPs - Part II | L:30

==========

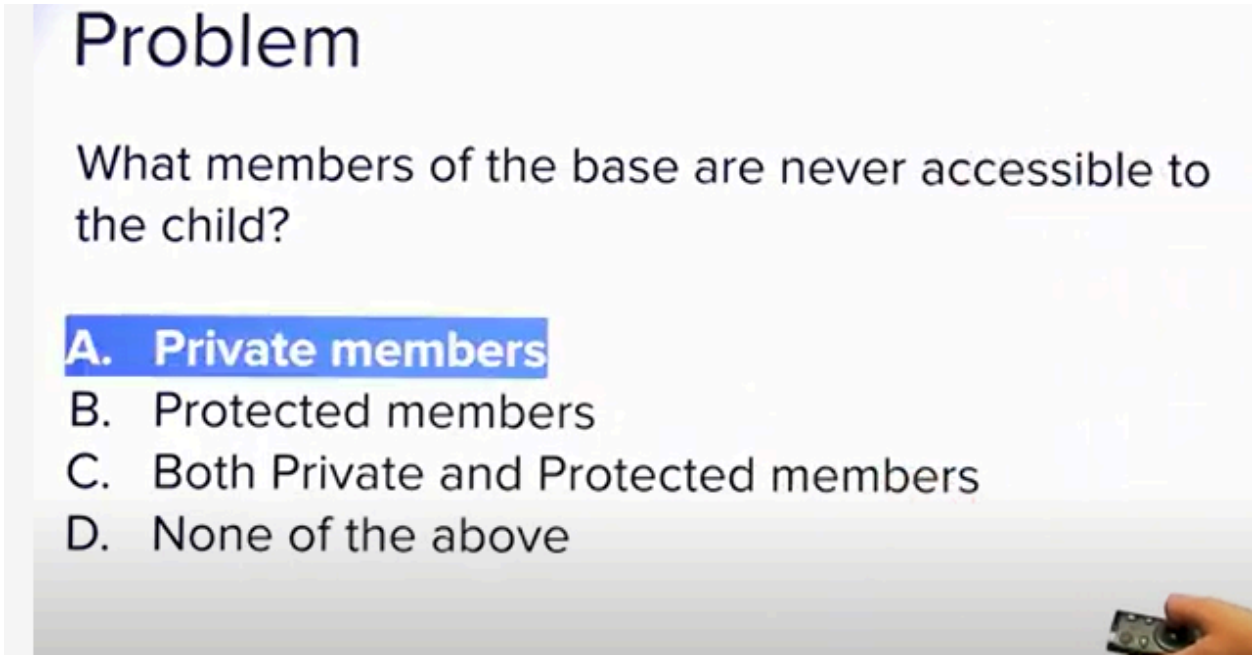
Q1:



Q2:

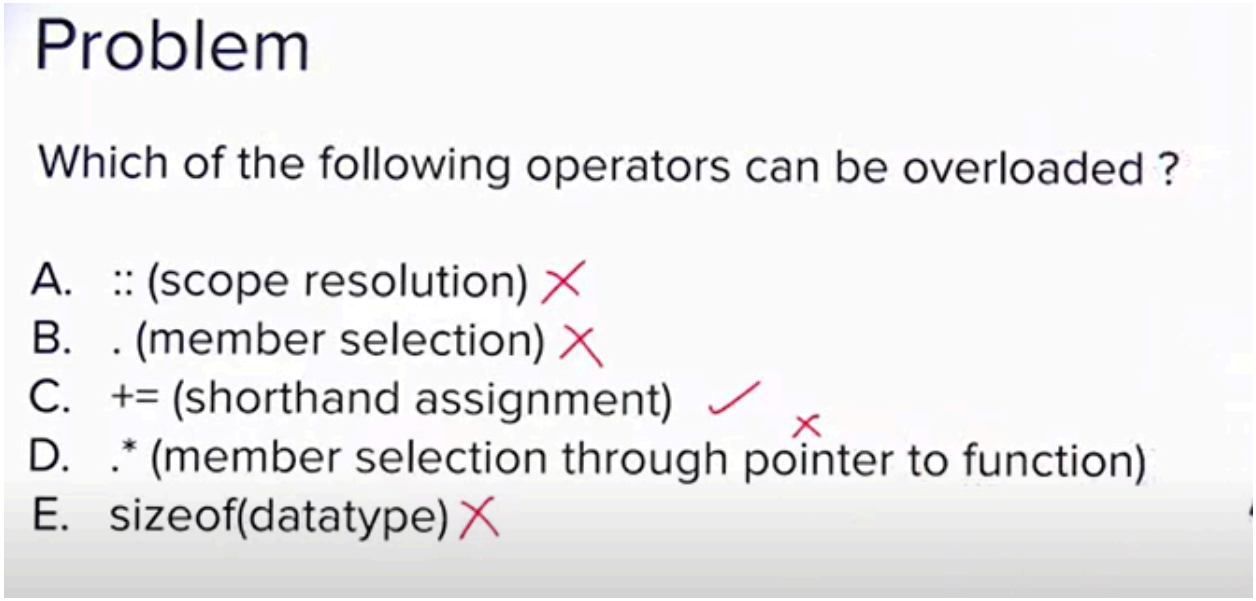


Q3:



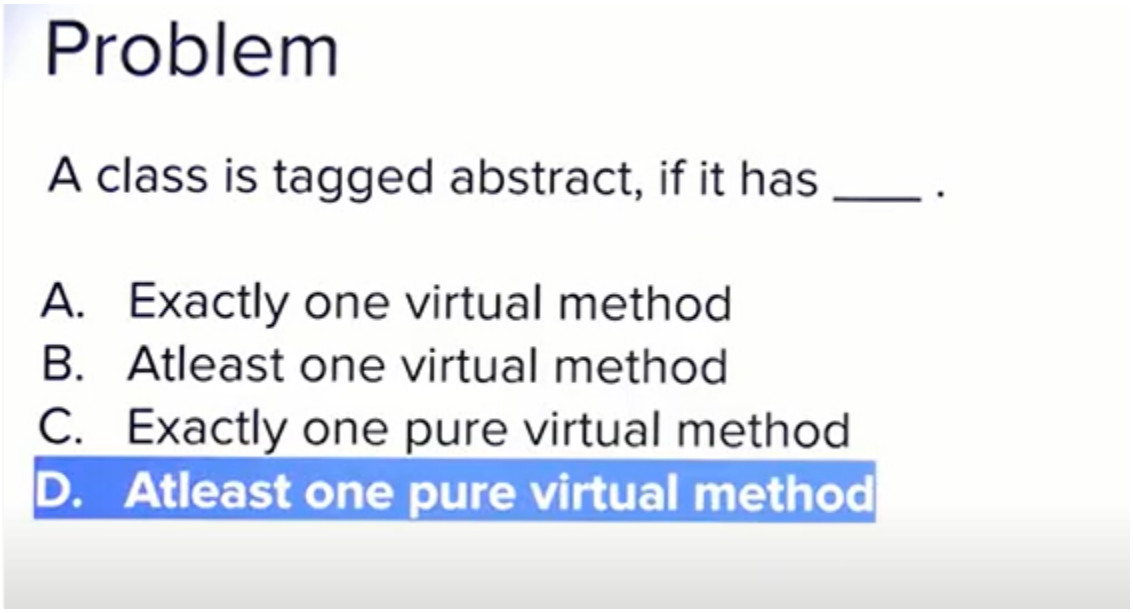
Protected members are accessible to the child.

Q4:



All type of the assignment operator can be overloaded.

Q5:

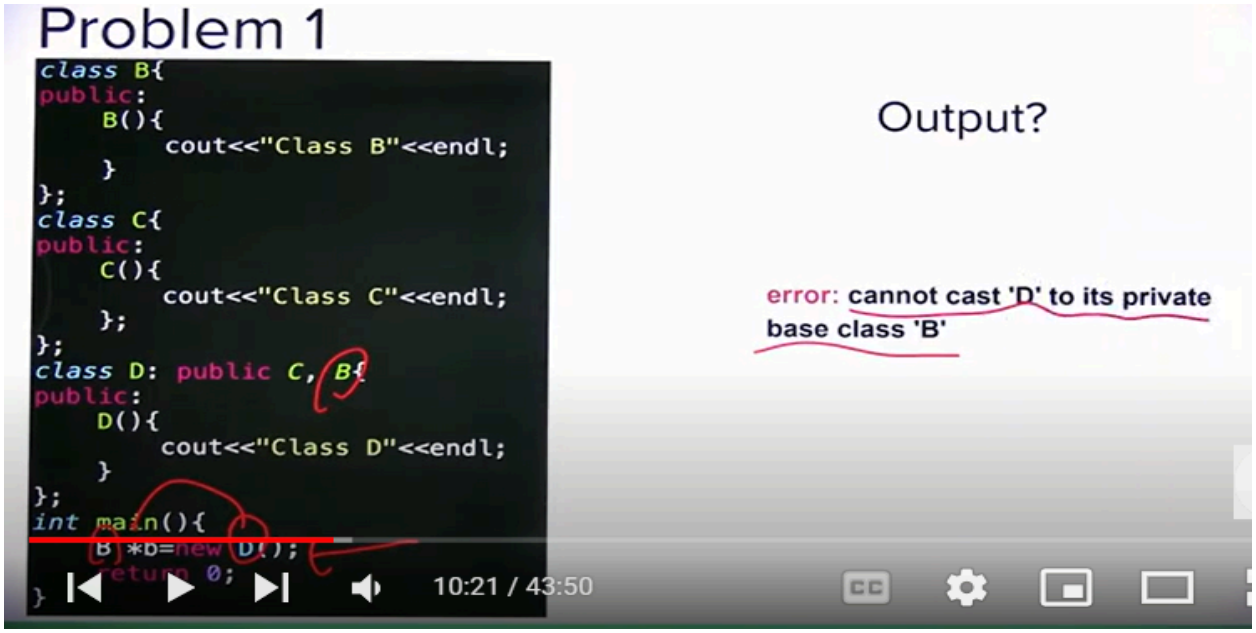

Problem
A class is tagged abstract, if it has ____ .
A. Exactly one virtual method
B. Atleast one virtual method
C. Exactly one pure virtual method
D. Atleast one pure virtual method

==========

50 Practice Problems in C++ OOPs - Part III | L:31

==========

Q1:

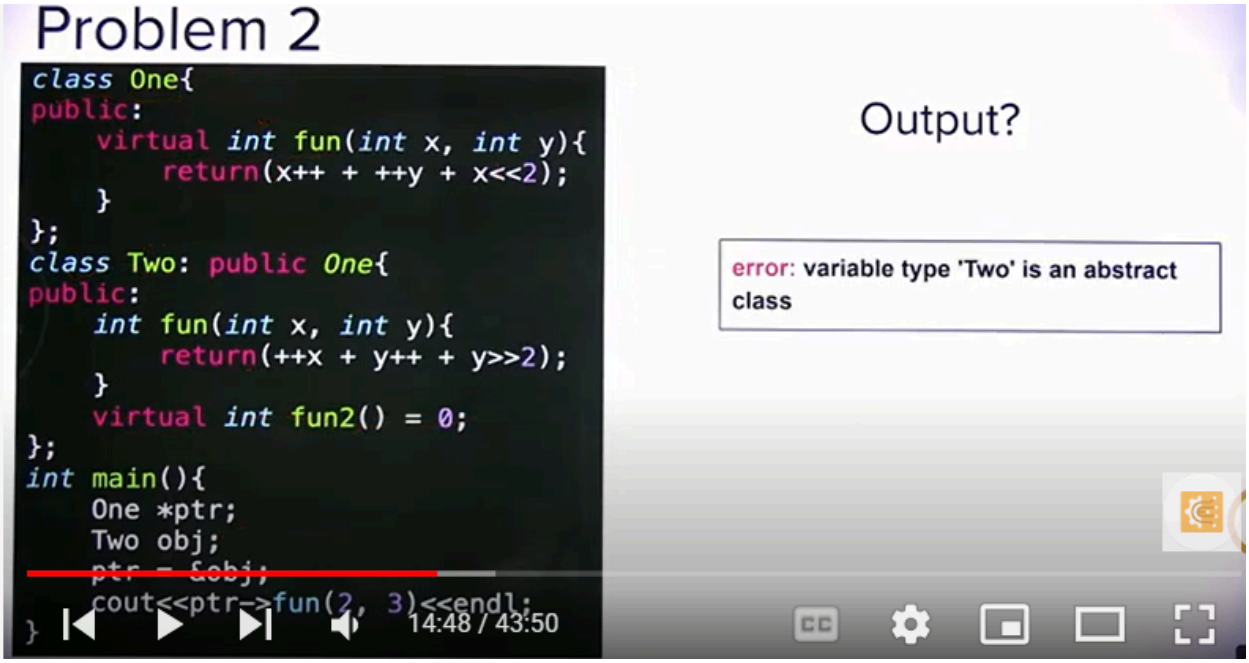


Default inherits private mode

Here class D inherits class B in private mode,so all the member of the class B which are inherited will become private.
constructor D() wants to call the C and D ,it can call the constructor of class C because it is public,but it can not call D because it is private.

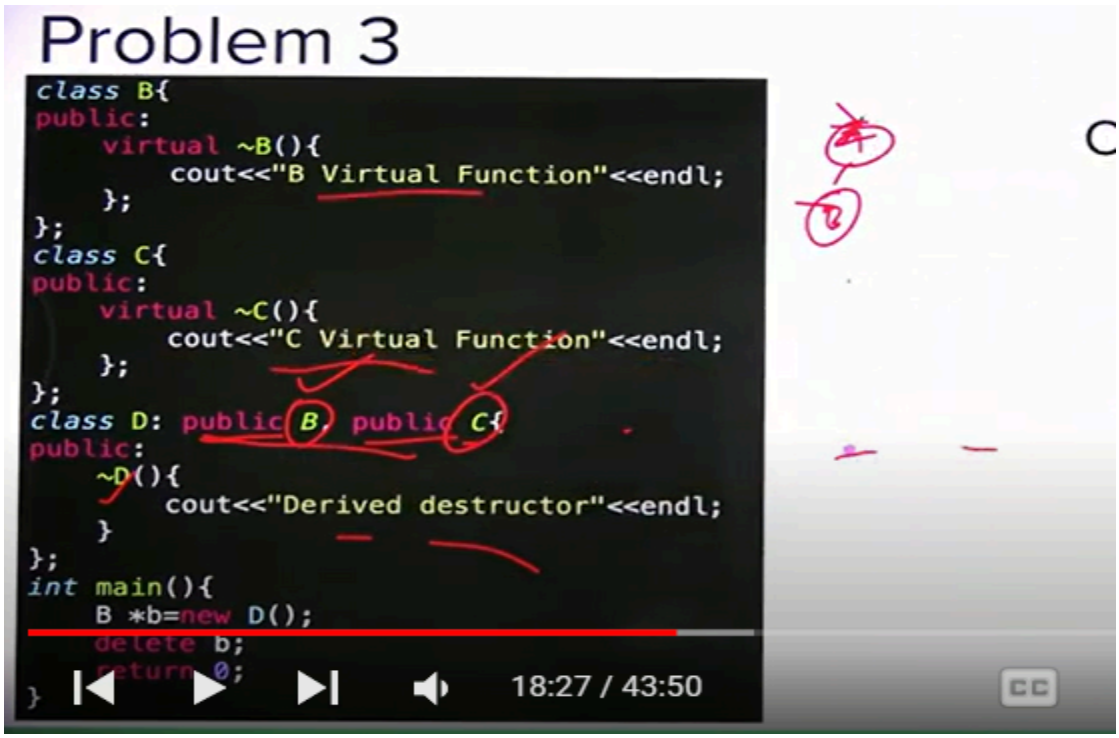
Yadi parent private inheritance kiya ho to ,parent ka pointer child ka address nahi rakh sakta.
Therefore, B *b=new D();

Q2:



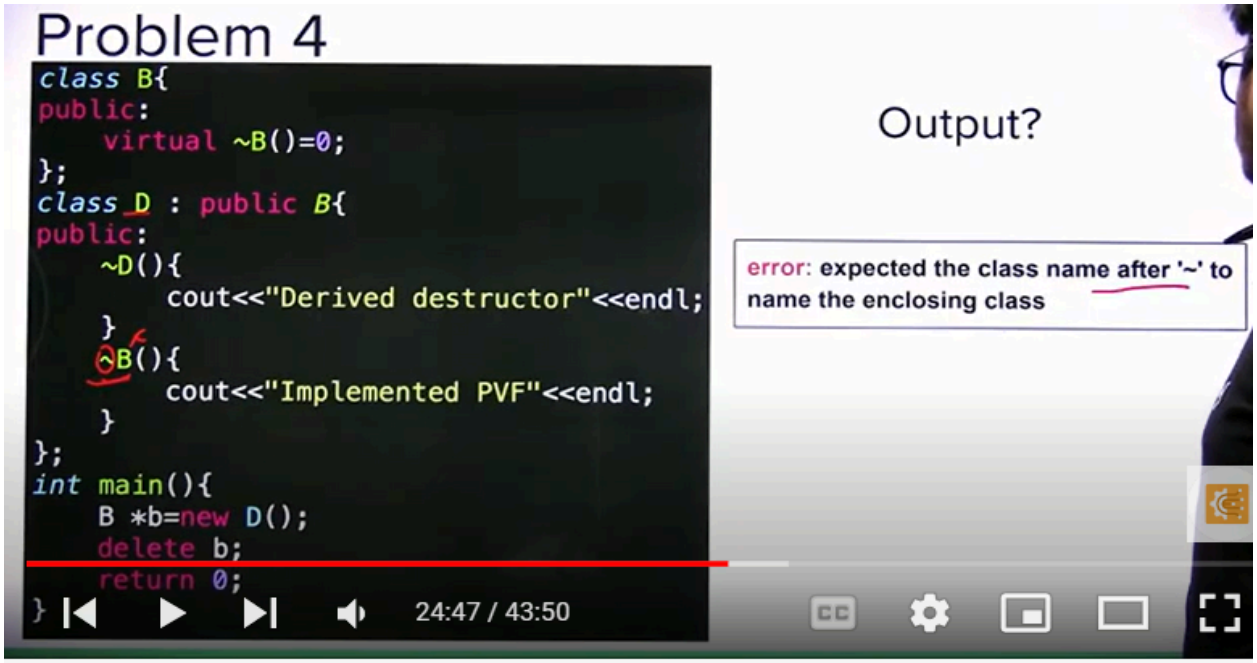
We can not create the object of the abstract class.

Q3:



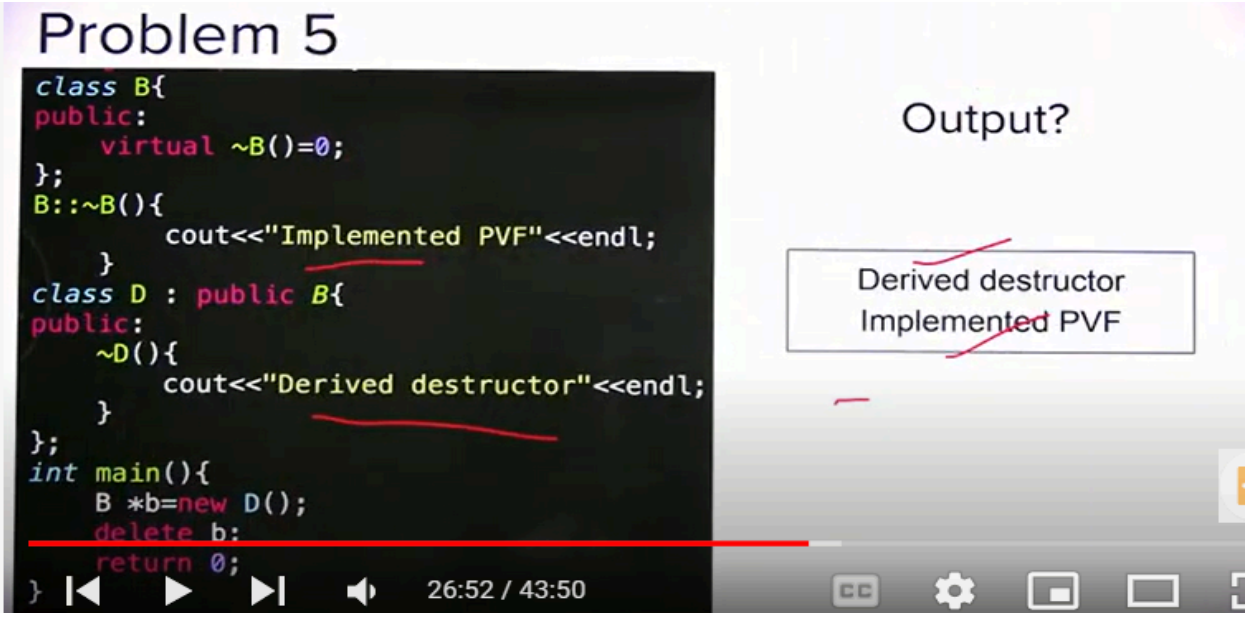
Op:==> destructor can be virtual ……..(see lec remaining of Cpp lec)
Derived destructor
C virtual function
B virtual function

Q4:

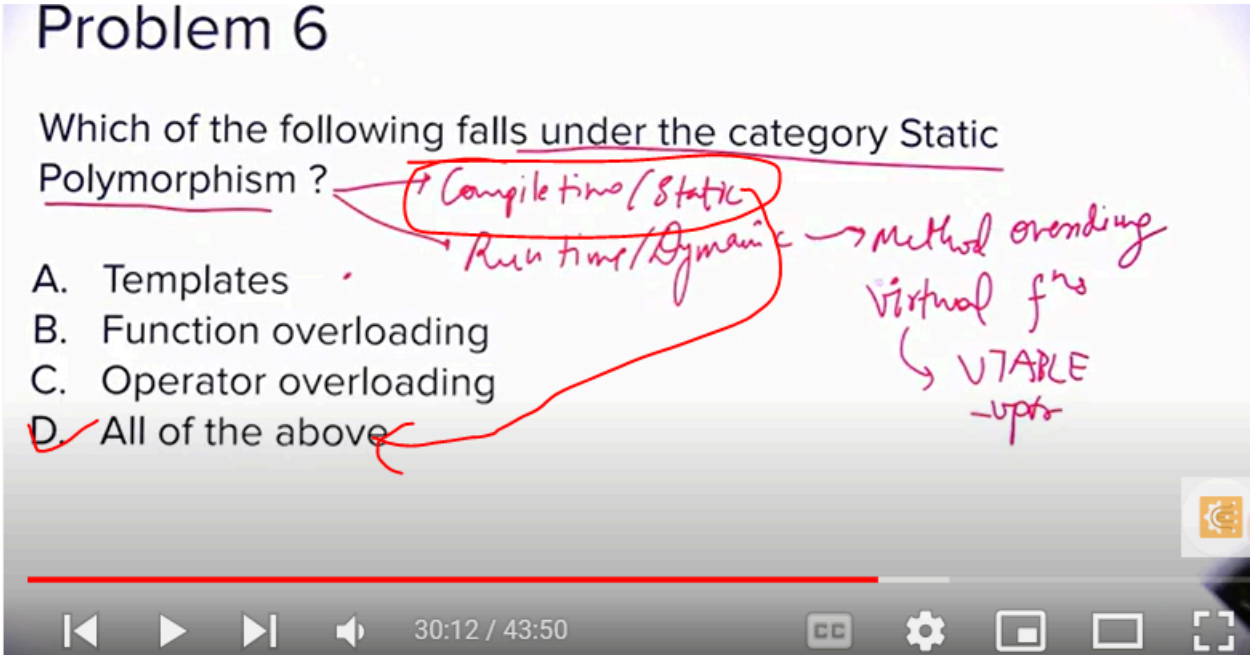


Note:
You can not define the constructor and destructor of any other class into any other class.
Ie. A class can not hold any constructor and destructor of any other other class.
So class D can not hold the destructor of class B.
We can implement the ~B() outside of any class.

Q5:

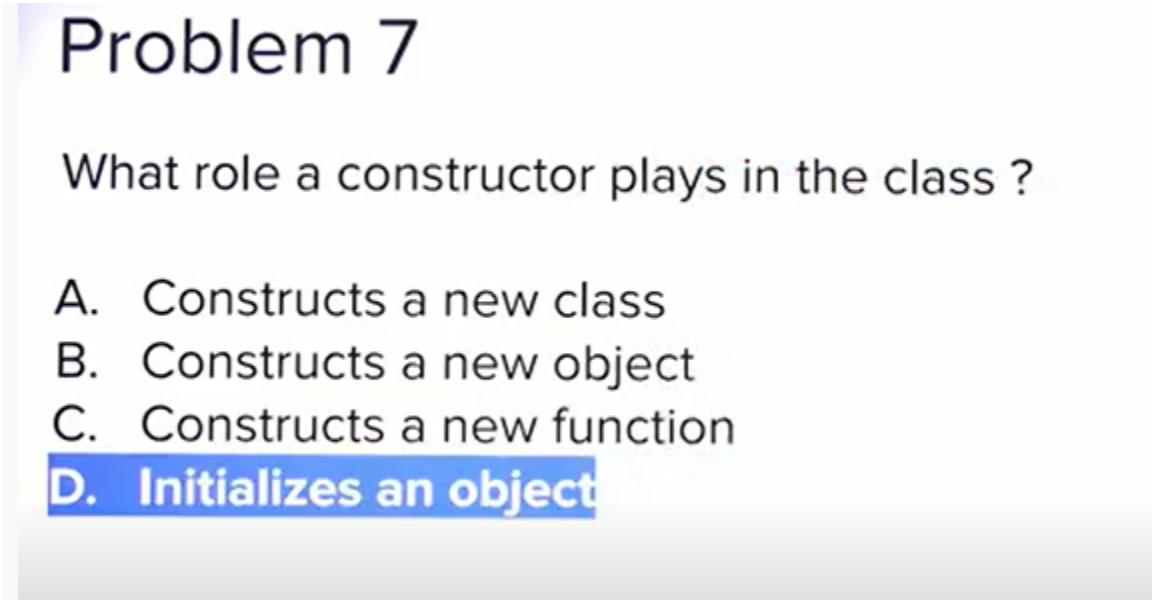


Q6:

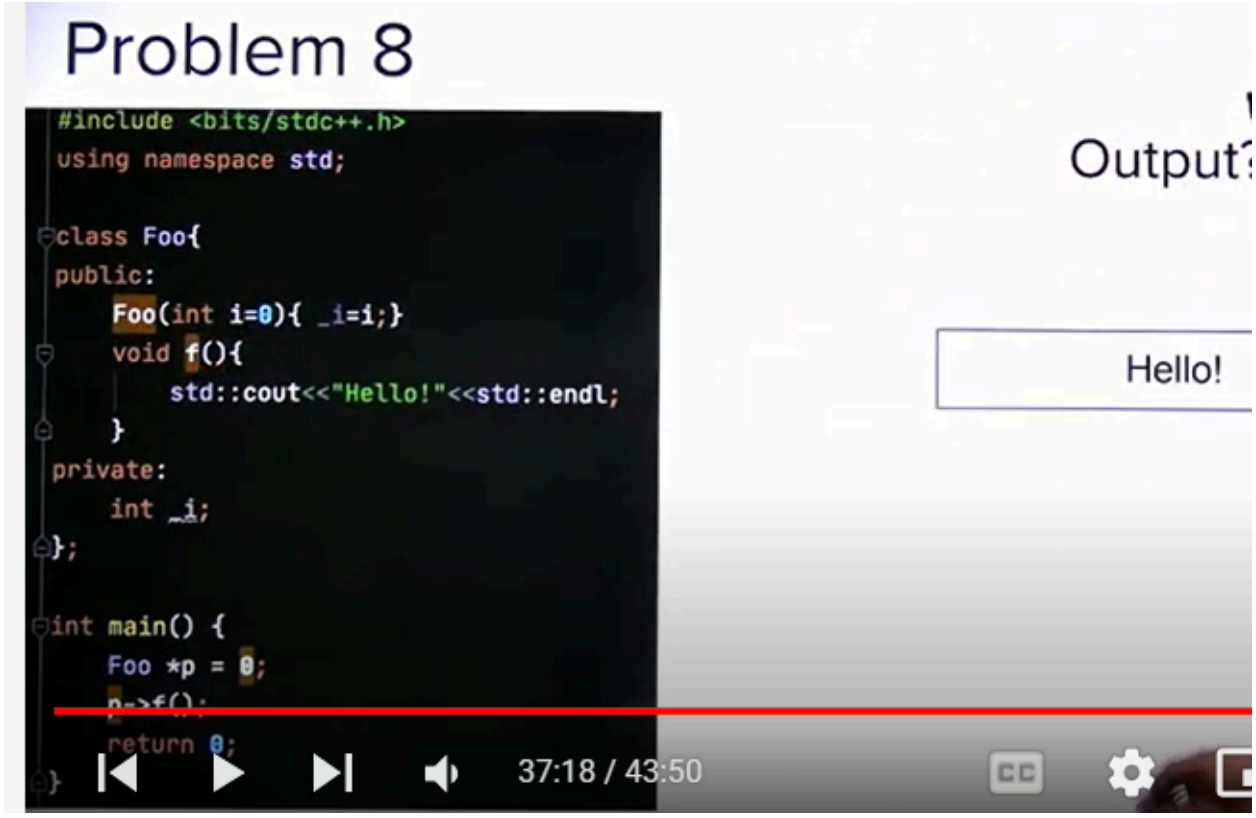


Compile time polymorphism can be obtained by using fo,00,And template .

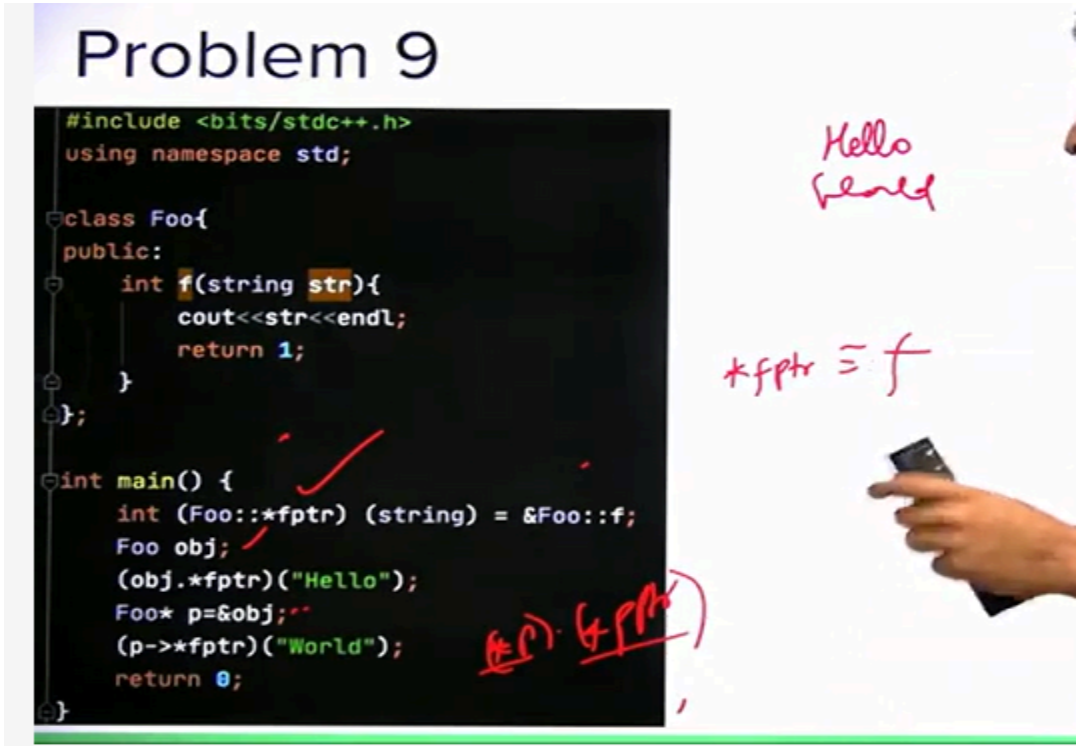
Q7:



Q8:


Problem 8
#include <bits/stdc++.h>
using namespace std;
class Foo{
public:
Foo(int i=0){ _i=i;}
void f(){
std::cout<<"Hello!"<<std::endl;
}
private:
int _i;
};
int main() {
Foo *p = 0;
p->f();
return 0;
}
Output?
Hello!
37:18 / 43:50

Q9:

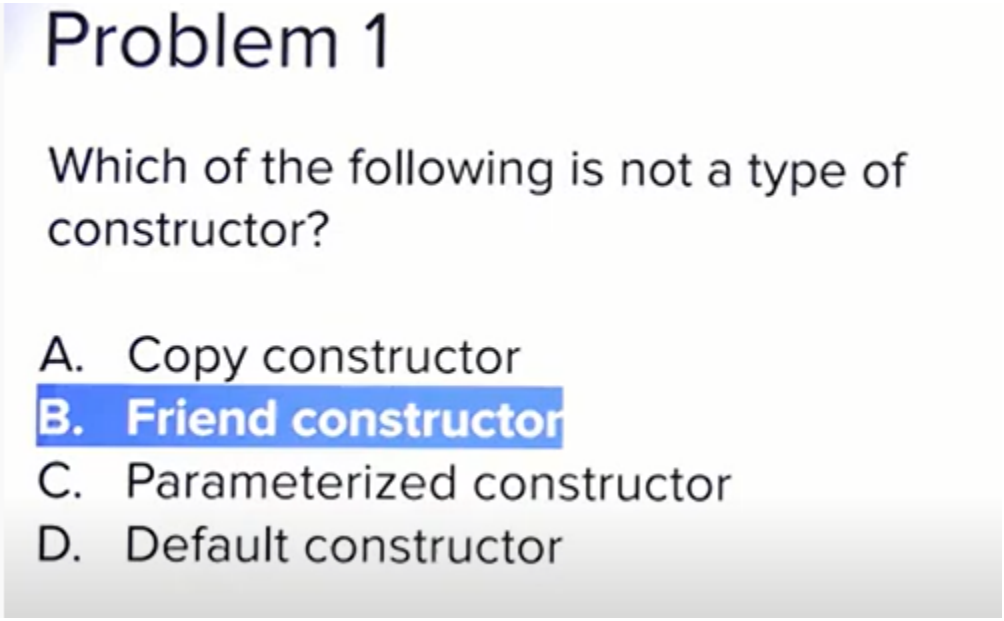

Problem 9
#include <bits/stdc++.h>
using namespace std;
class Foo{
public:
int f(string str){
cout<<str<<endl;
return 1;
}
};
int main() {
int (Foo::*fptr) (string) = &Foo::f;
Foo obj;
(obj.*fptr)("Hello");
Foo* p=&obj;
(p->*fptr)("World");
return 0;
}
Hello
*fptr = f

==========

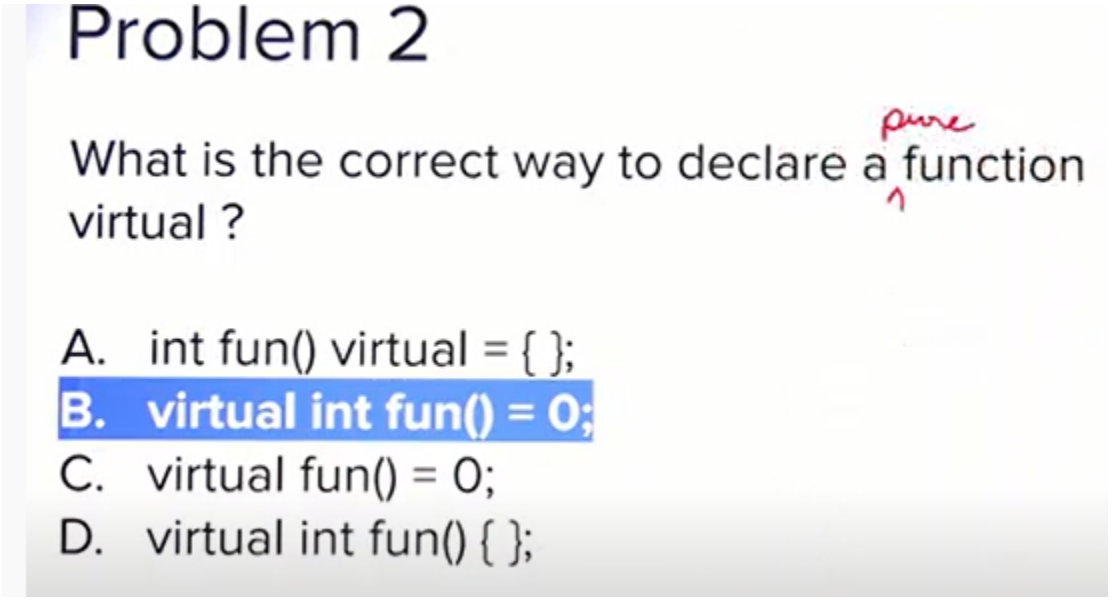
50 Practice Problems in C++ OOPs - Part IV | L:32

==========

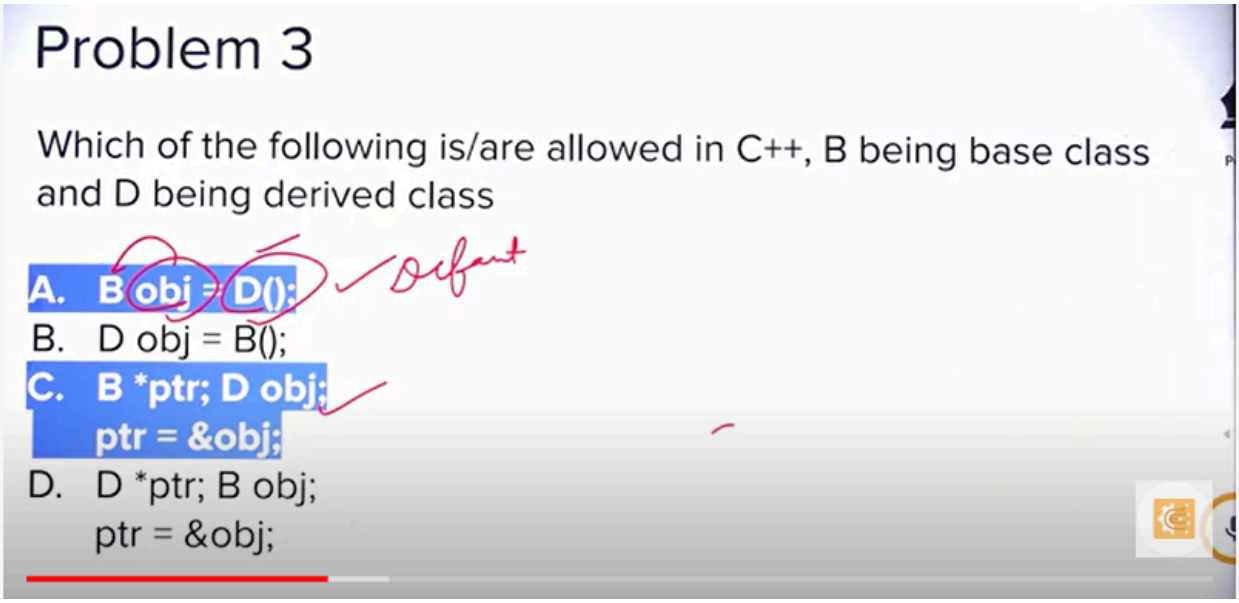
Q1:



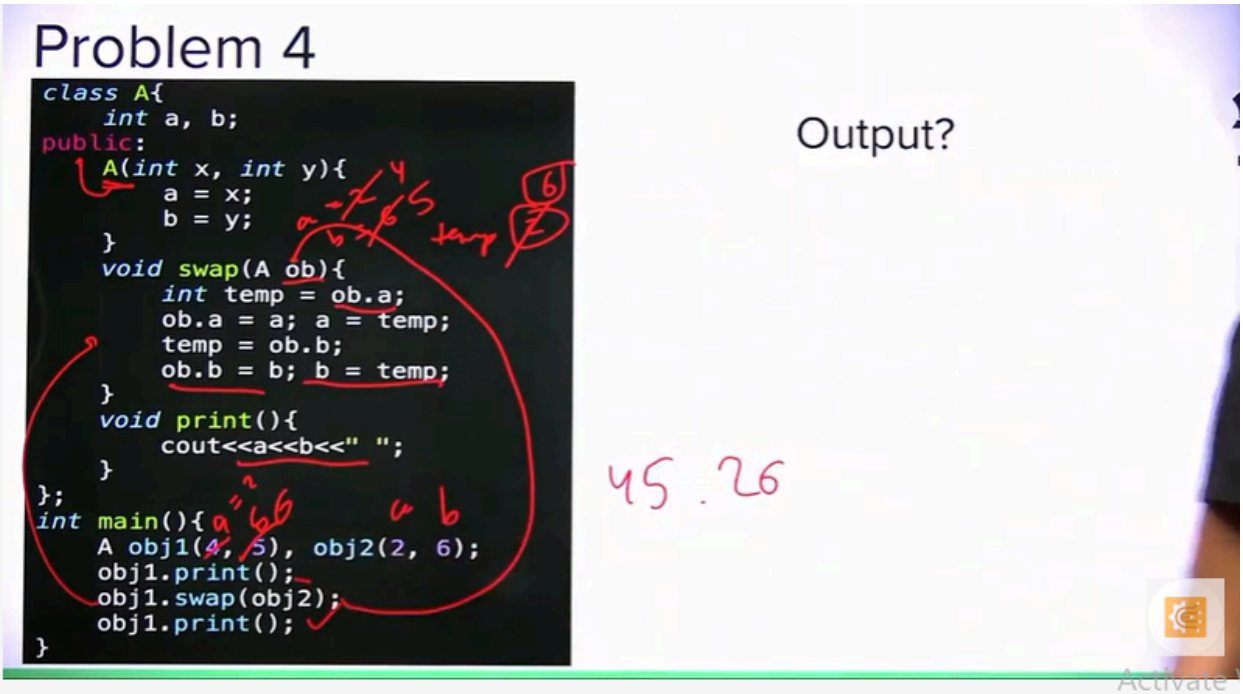
Q2:



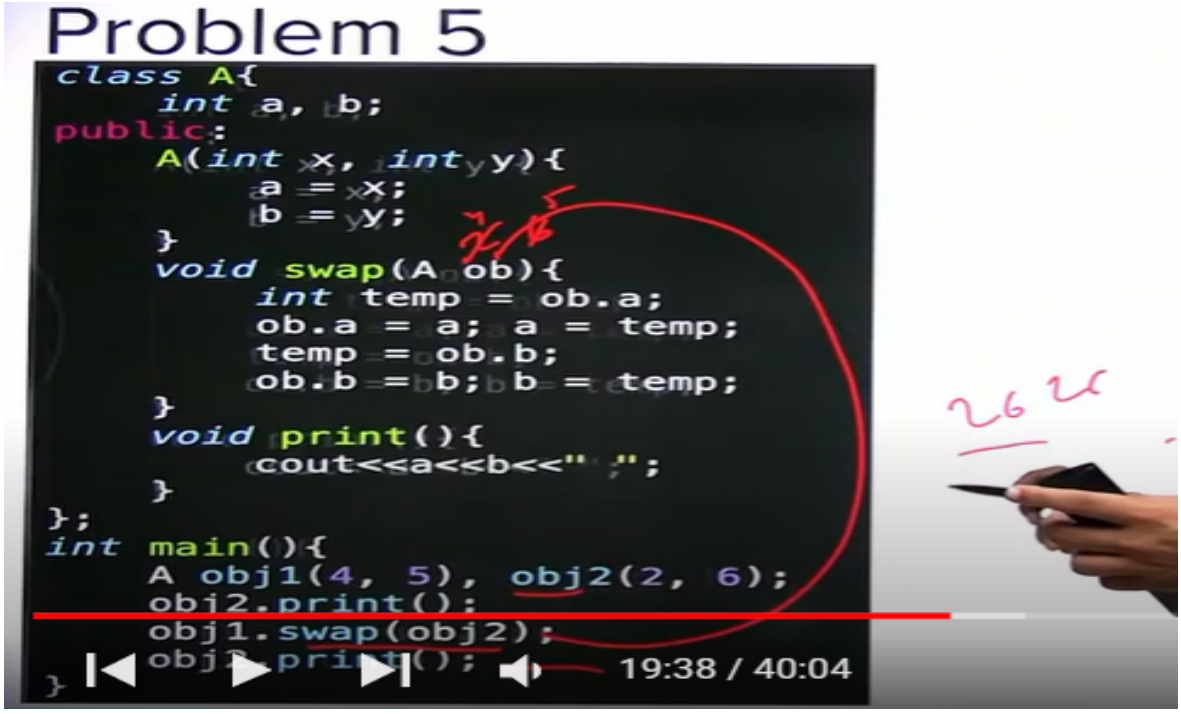
Q3:


Problem 3
Which of the following is/are allowed in C++, B being base class and D being derived class
A. B obj = D();
B. D obj = B();
C. B *ptr; D obj;
ptr = &obj;
D. D *ptr; B obj;
ptr = &obj;

Q4:


Problem 4
class A{
int a, b;
public:
A(int x, int y){
a = x;
b = y;
}
void swap(A ob){
int temp = ob.a;
ob.a = a; a = temp;
temp = ob.b;
ob.b = b; b = temp;
}
void print(){
cout<<a<<b<<" ";
}
};
int main(){
A obj1(4, 5), obj2(2, 6);
obj1.print();
obj1.swap(obj2);
obj1.print();
}
Output?
45 26

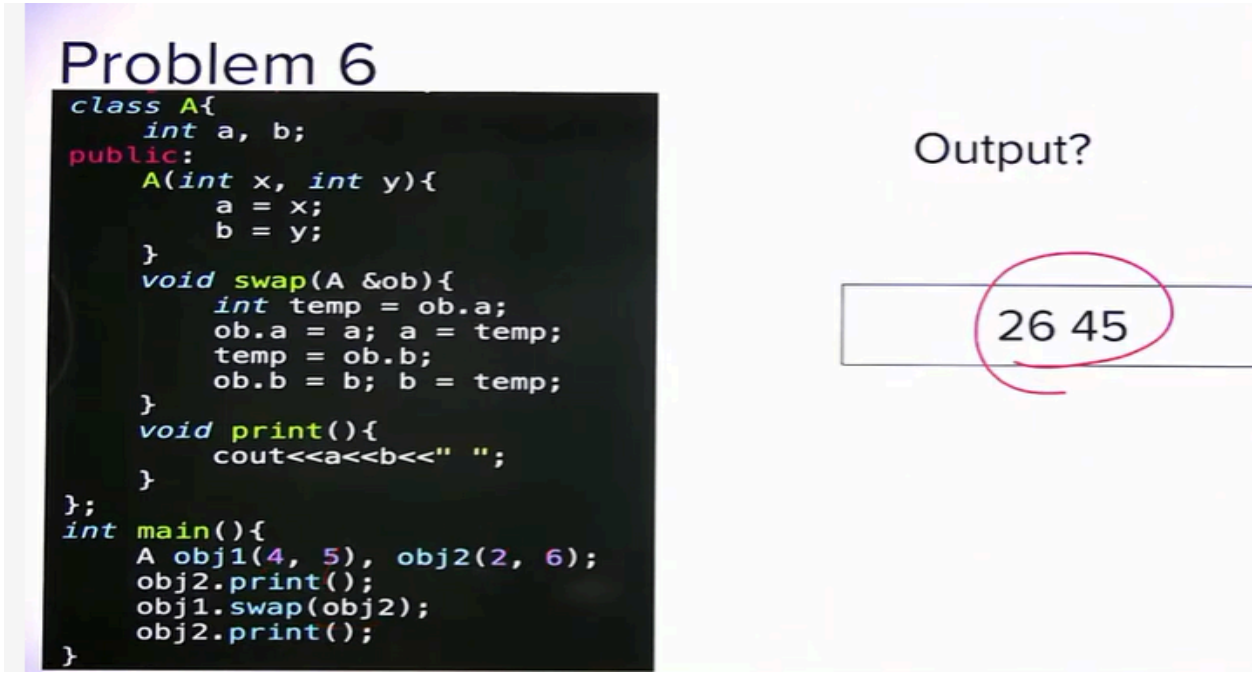
Q5:



Here we are passing the obj2 value⇒it will be the pass by value
So changes into the "ob" will not affect the obj2.,since another copy of the "ob" will be created.

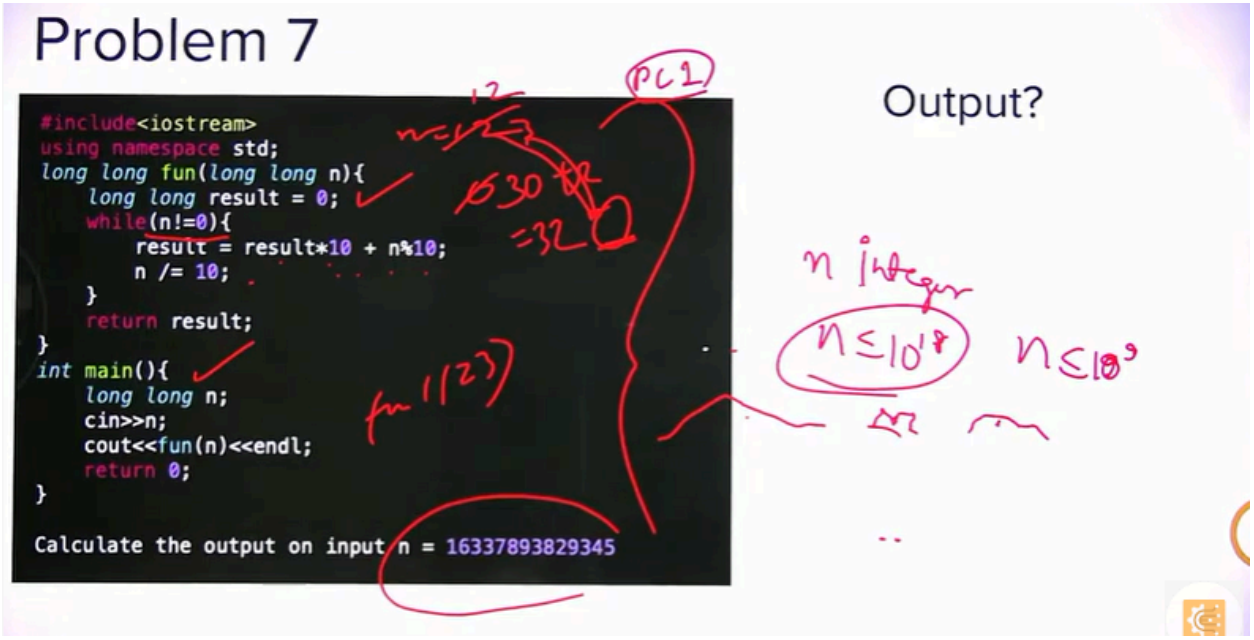
If we use "&" value,then "ob" will point to the same object obj2,no separate object will be created.

Q6:



Q7:reverse the number without converting it into the string.

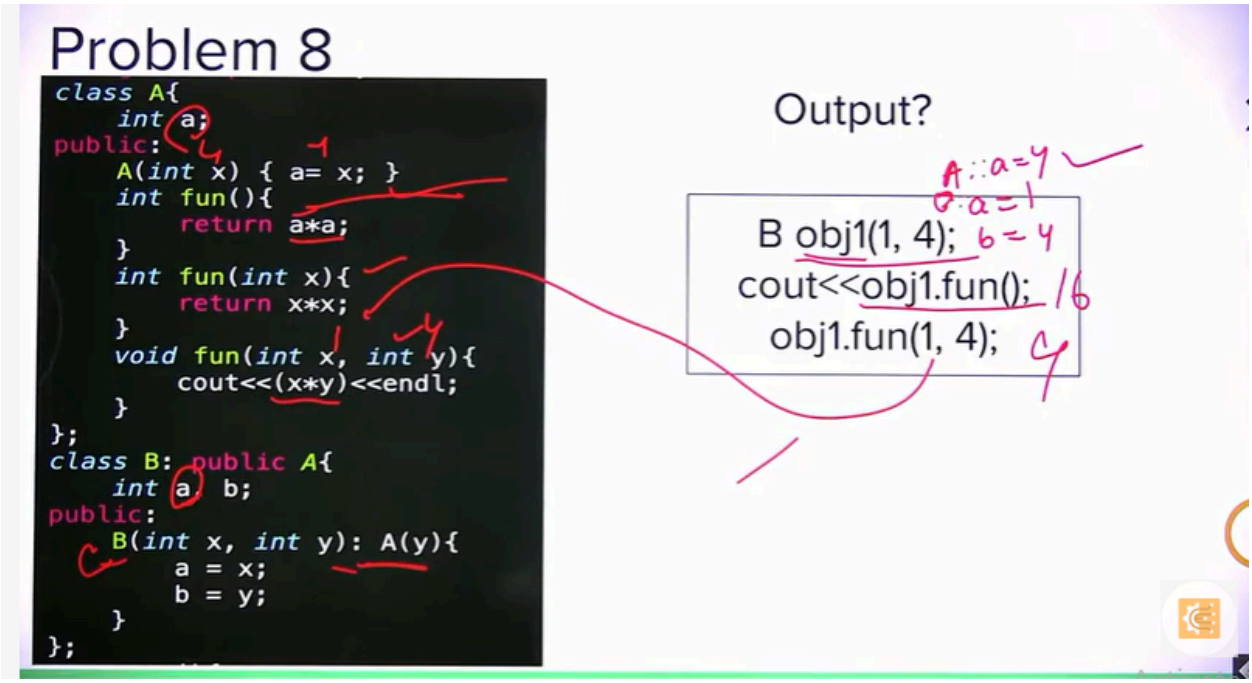
Int8_t,int16_t,int32_t,int64_t
uint8_t,uint16_t,uint32_t,uint64_t



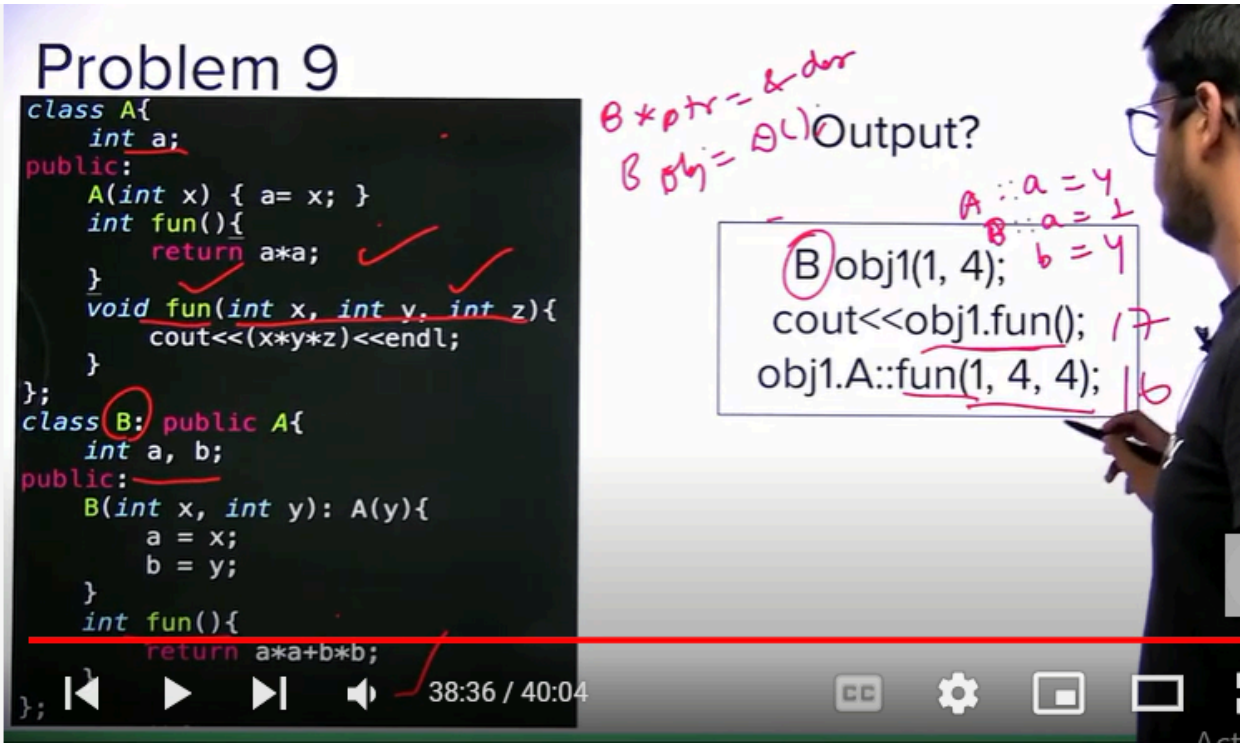

```cpp
#include <iostream>
using namespace std;
uint64_t reverse(uint64_t n)
{
    uint64_t result=0;
    while(n!=0)
    {
        result=result*10+n%10;
        n=n/10;
    }
    return result;
}
int main()
{
    cout<<reverse(1234567899999);

    return 0;
}
```

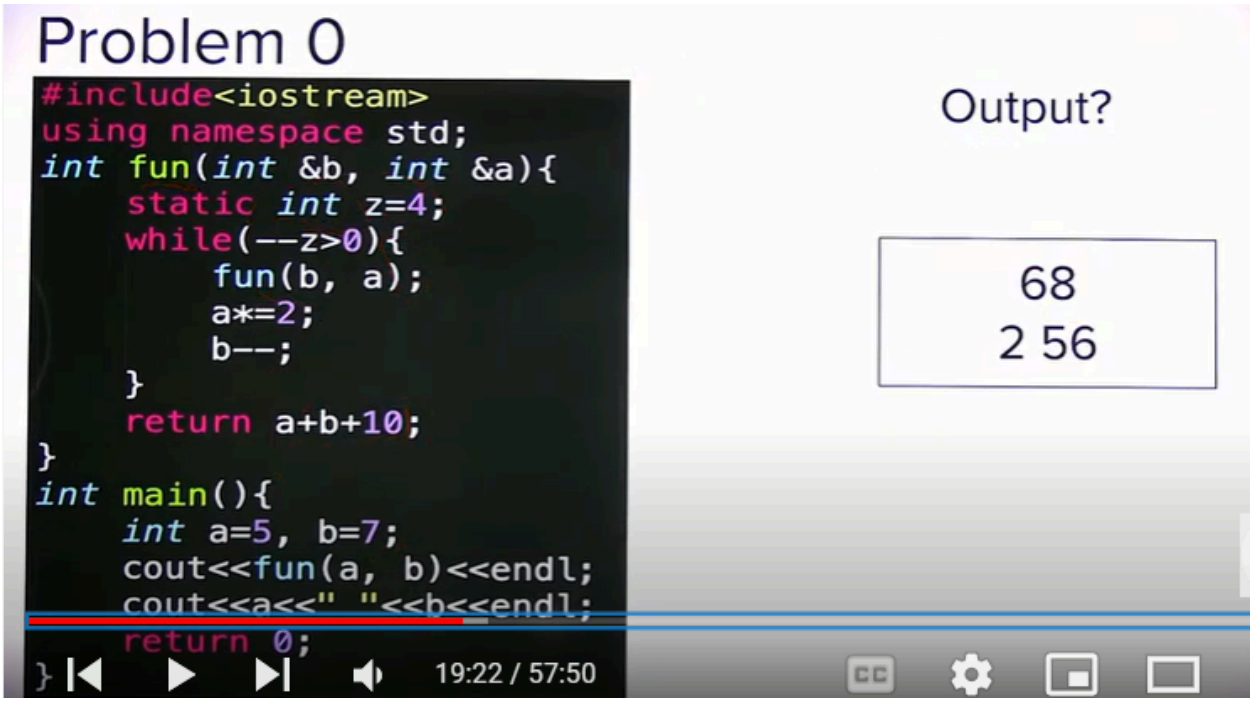
```
9999987654321

...Program finished with exit code 0
Press ENTER to exit console.
```

Q8:


Problem 8
class A{
int a;
public:
A(int x) { a= x; }
int fun(){
return a*a;
}
int fun(int x){
return x*x;
}
void fun(int x, int y){
cout<<(x*y)<<endl;
}
};
class B: public A{
int a, b;
public:
B(int x, int y): A(y){
a = x;
b = y;
}
};
Output?
B obj1(1, 4);
cout<<obj1.fun();
obj1.fun(1, 4);

Q9:


Problem 9
class A{
int a;
public:
A(int x) { a= x; }
int fun(){
return a*a;
}
void fun(int x, int y, int z){
cout<<(x*y*z)<<endl;
}
};
class B: public A{
int a, b;
public:
B(int x, int y): A(y){
a = x;
b = y;
}
int fun(){
return a*a+b*b;
}
};
Output?
B obj1(1, 4);
cout<<obj1.fun();
obj1.A::fun(1, 4, 4);
38:36 / 40:04

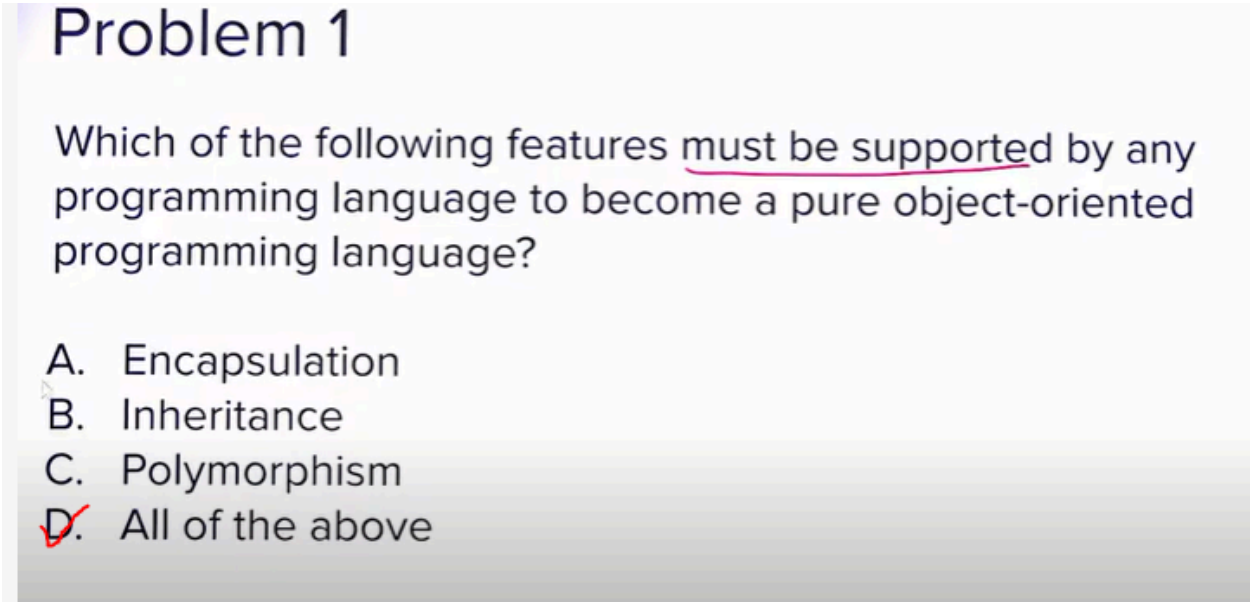


==========

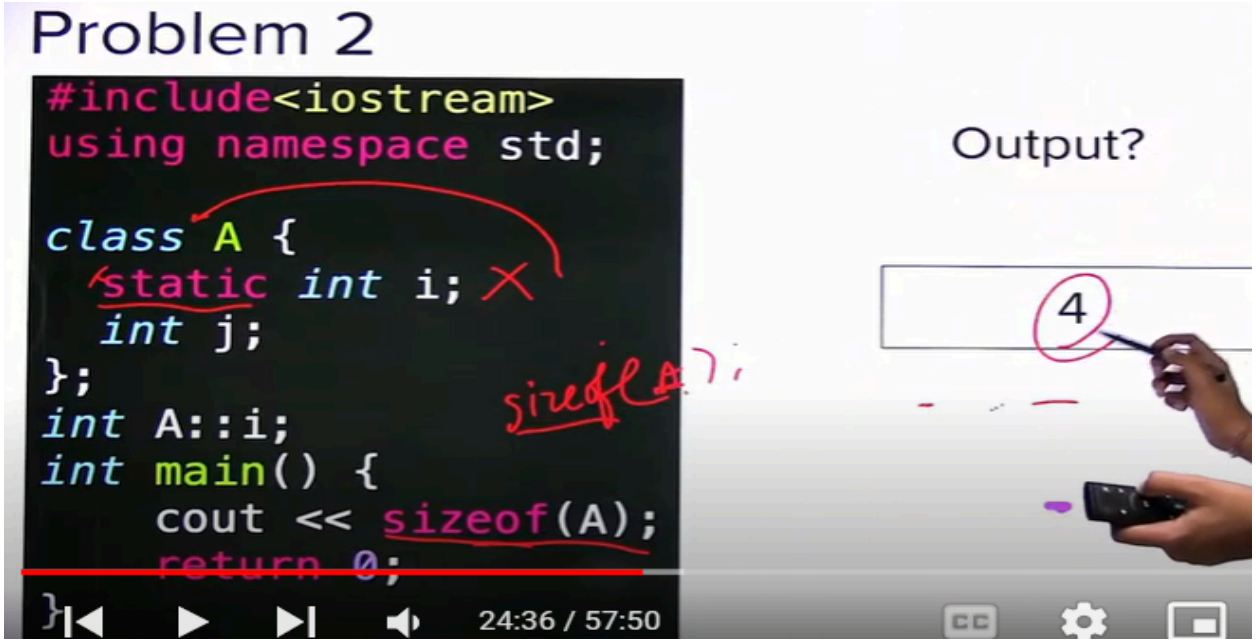
50 Practice Problems in C++ OOPs - Part V | L:33

==========

Q1:

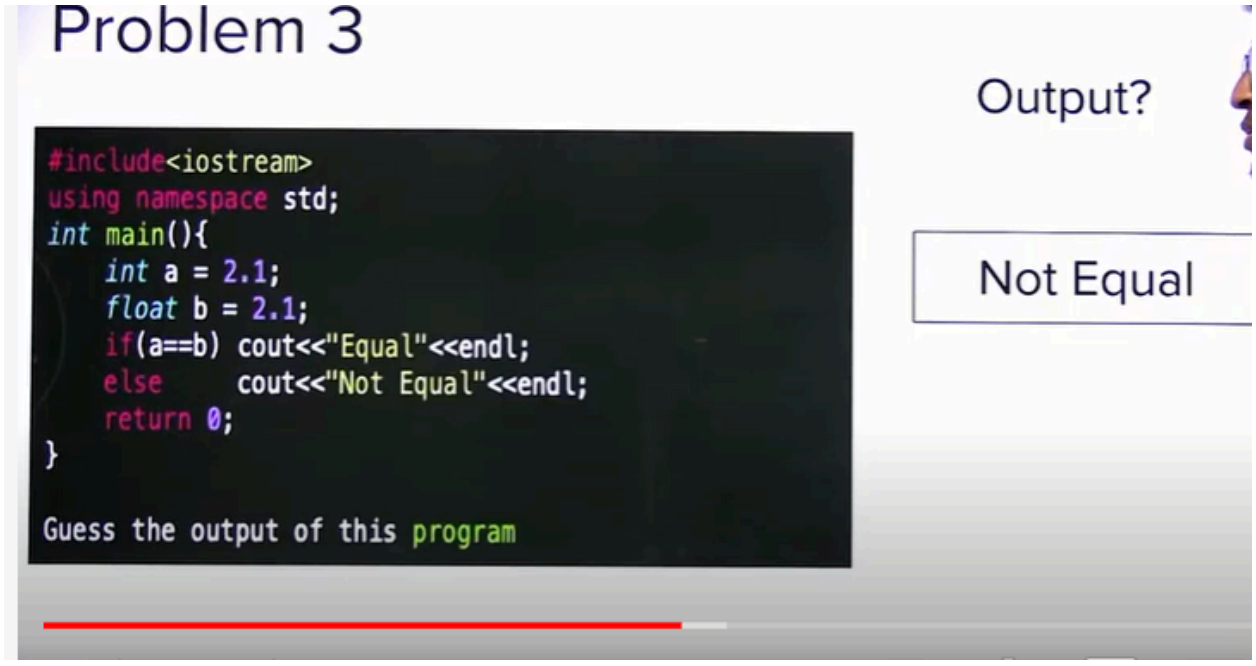


Q2:

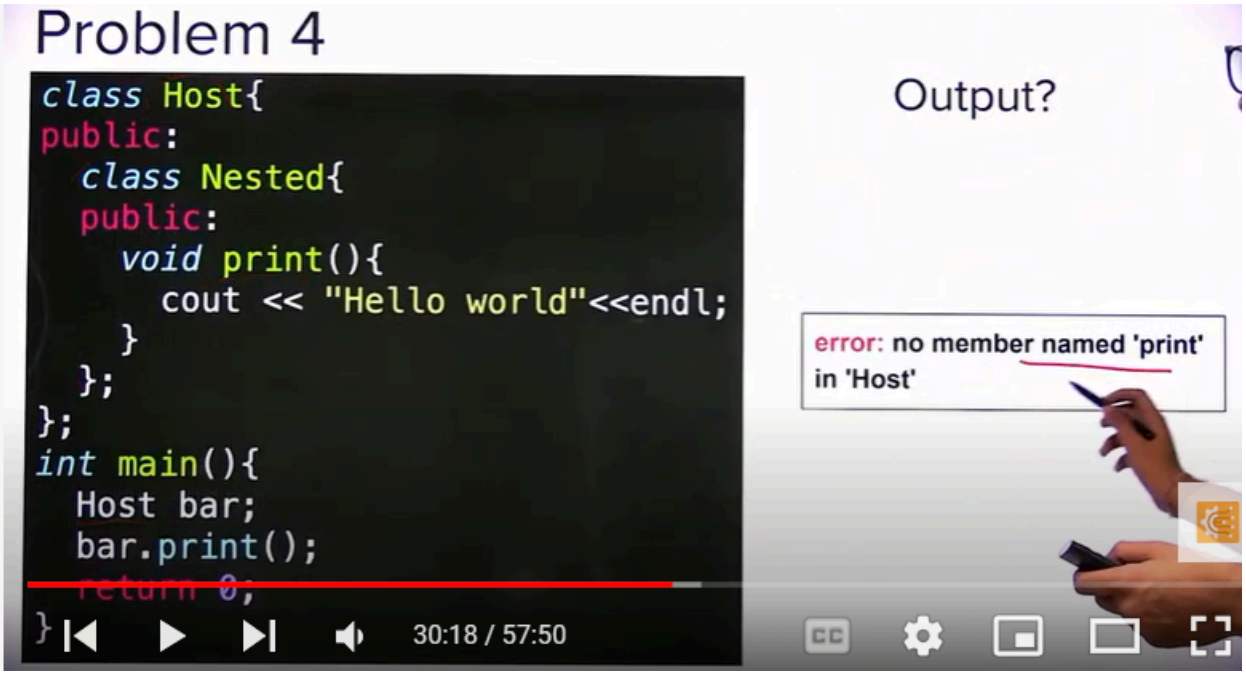


sizeof(class)==>give the size of any object present into the class.
Static member of the class does not belong to any object of the class.

Q3:



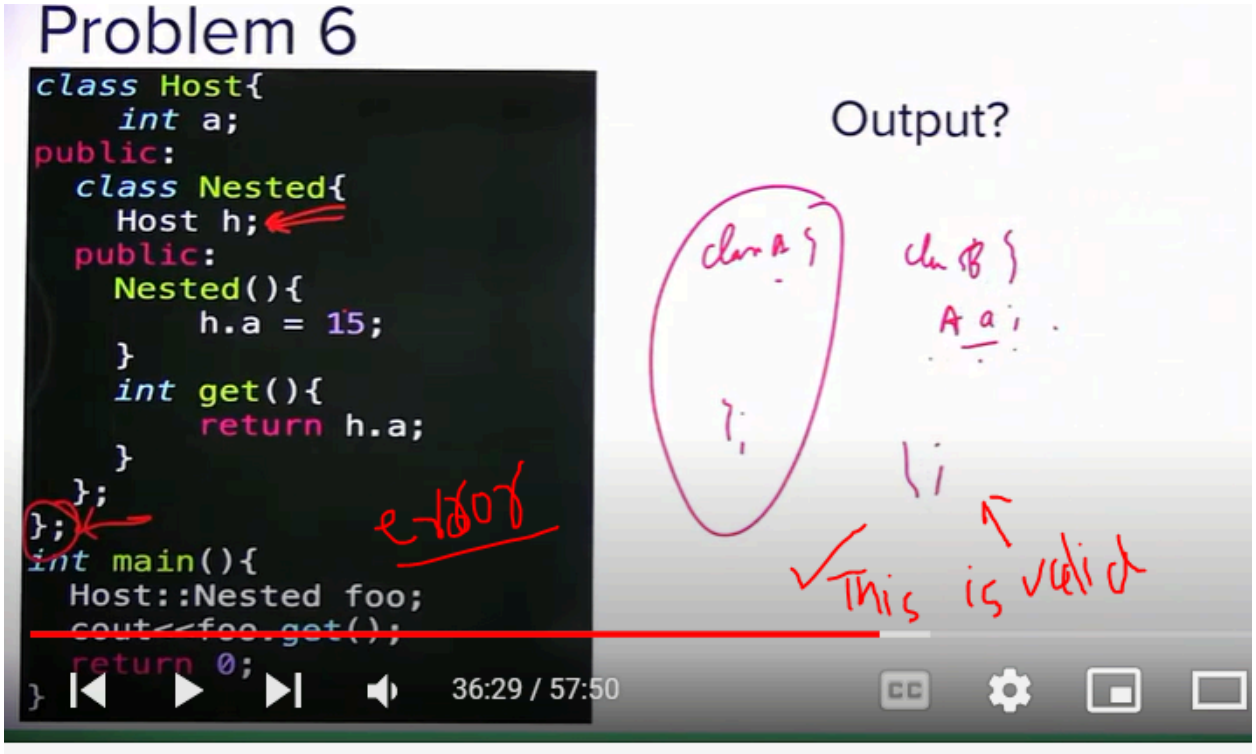
Q4:



Class host does not have any right on the method print(),it belongs to nested class only.

Q5:

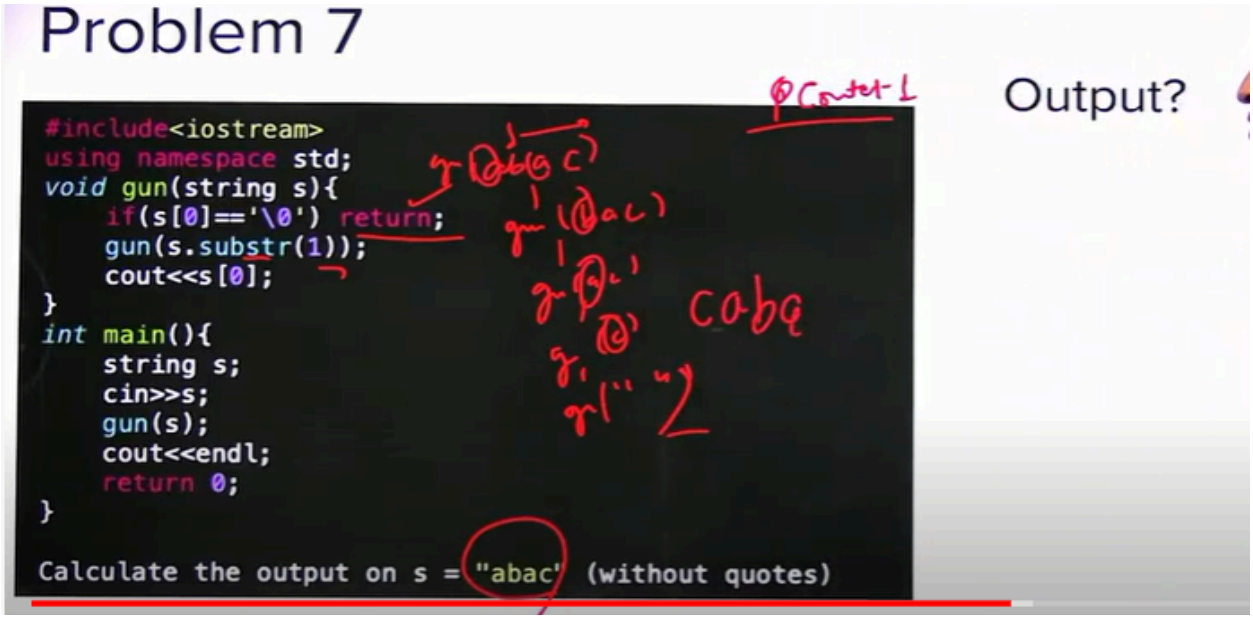
Q6:



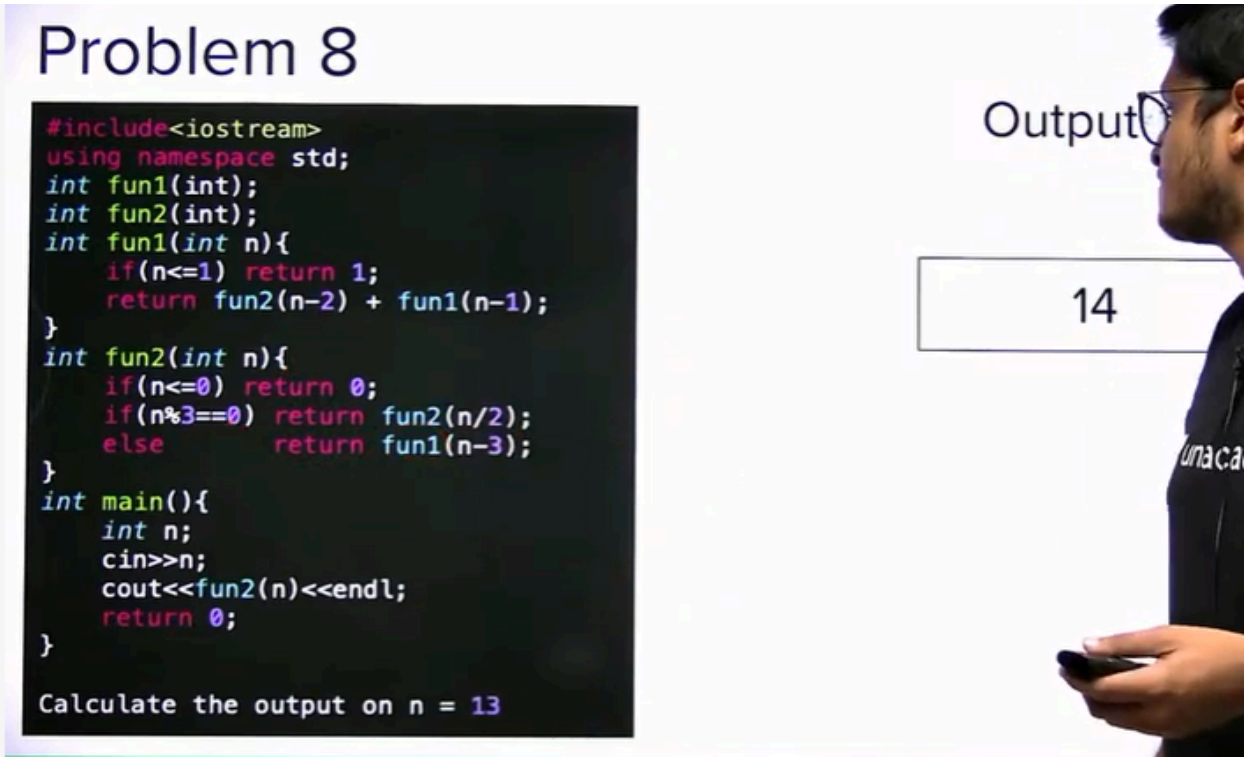

```
error: field has incomplete type 'Host'
(definition of 'Host' is not complete until the closing '}')
```

You can not create the object of the class before defining it completely.

Q7:



Q8:


Problem 8
#include<iostream>
using namespace std;
int fun1(int);
int fun2(int);
int fun1(int n){
if(n<=1) return 1;
return fun2(n-2) + fun1(n-1);
}
int fun2(int n){
if(n<=0) return 0;
if(n%3==0) return fun2(n/2);
else return fun1(n-3);
}
int main(){
int n;
cin>>n;
cout<<fun2(n)<<endl;
return 0;
}
Calculate the output on n = 13
Output
14