Part1: Overriding functions of base class

Function overriding in [C++](https://www.simplilearn.com/c-plus-plus-programming-for-beginners-article) is a concept by which you can define a function of the same name and the same function signature (parameters and their data types) in both the base class and derived class with a different function definition. It redefines a function of the base class inside the derived class, which overrides the base class function. Function overriding is an implementation of the run-time polymorphism. So, it overrides the function at the run-time of the program.

#include<iostream>

using namespace std;

class X

{

public:

void h(float f) { cout << f << endl; }

void ov(int i) { cout << i << endl; }

int a;

};

class Y : public X

{

public:

void h(char c[]) { cout << c << endl; }

void ov(int k)

{

cout << k << " : ";

X::ov(k);

}

void print() { cout << a << ":" << X::a << endl; }

int a;

};

int main()

{

Y y;

y.ov(10); // calling Y's ov(int)

y.X::ov(20);// calling X's ov(int)

return 0;

}

Output:

|  |
| --- |
|  |

Can you call X’s h function in the Y function ?

Output:

|  |
| --- |
|  |

**A Derived Class Object as a Base Class Object**

#include<iostream>

using namespace std;

class X

{

public:

X() { a = 1; }

void print() { cout << "Class X \n"; }

private:

int a;

};

class Y : public X

{

public:

Y() { b = 2; }

void print() { cout << "Class Y \n"; }

private:

int b;

};

class Z : public Y

{

public:

Z() { c = 3; }

void print() { cout << "Class Z \n"; }

private:

int c;

};

int main()

{

X x1; Y y1; Z z1;

x1 = y1;

x1.print();

x1 = z1;

x1.print();

y1 = z1;

x1.print();

return 0;

}

Output:

|  |
| --- |
|  |

Update class Z to be inherited from Class X instead of Class Y.

What is the output:

|  |
| --- |
|  |

**A Derived Class Pointer as a Base Class Pointer**

#include<iostream>

using namespace std;

class X

{

public:

X() { a = 1; }

void print() { cout << "Class X \n"; }

private:

int a;

};

class Y : public X

{

public:

Y() { b = 2; }

void print() { cout << "Class Y \n"; }

private:

int b;

};

class Z : public Y

{

public:

Z() { c = 3; }

void print() { cout << "Class Z \n"; }

private:

int c;

};

void test(X xx)

{

xx.print();

}

int main()

{

X x; X\* Px; Y y; Z z;

Px = &x;

Px->print();

Px = &y;

Px->print();

((Y\*)Px)->print();

Px = &z;

Px->print();

((Z\*)Px)->print();

((Y\*)Px)->print();

}

Output:

|  |
| --- |
|  |

That is the end of labsheet.. Good Luck

Explain the following line :

((Z\*)Px)->print();

|  |
| --- |
|  |

Replace Y instead of Z?!