If you define a base class and some child classes, then you want to declare the child class by declaring a pointer of the base class. It’s very simple.

For example:

Class Shape is the base class, class Circle is the child

You can do this: Shape \*s = new Circle;

But if you have a function in the base class and you inherit the function from the base class to different child classes. Now you want this function do different jobs in different child class but without changing the name of function (use the same name inherit from the base class). You need to use virtual function.

To declare a virtual function, just add “virtual” in from of the function name. Remember once you defined a virtual function in the base, you have to defined the content of the function in each child class ( Because this function might do different jobs in different child class )

Now let’s talk about abstract class. If you define a base class which you just want to inherit the variables and functions to the child class but you will never declare an object using the bass class, you can declare a virtual function = 0 in the base class. (Now the virtual in base class become a pure virtual function). If there is a pure function in the base class, you will not allowed to declare an object of the bass class. So now the base class become an abstract class.

To combine all the cases above, let’s see the example ~

#include <iostream>

#include <string>

using namespace std;

//abtract base class

class Shape

{

protected:

string name;

public:

Shape(string s)

{

name = s;

}

string getName()

{

return name;

}

//Shape become a abtract class because you set

//this virtual function = 0 (become a pure function)

virtual double getArea() = 0;

};

// First child class

class Circle : public Shape

{

private:

double radius;

public:

Circle(string s, double r) :Shape(s)

{

radius = r;

}

//Since you define getArea is a virtual function

//You need to define getArea in each child class specifically

//Because getArea now will do different job in different child class

virtual double getArea()

{

return 3.14 \* radius \* radius;

}

};

//Second child class

class Rectangle : public Shape

{

private:

double length, width;

public:

Rectangle(string s, double l, double w) : Shape(s)

{

length = l;

width = w;

}

//Since you define getArea is a virtual function

//You need to define getArea in each child class specifically

//Because getArea now will do different job in different child class

virtual double getArea()

{

return length \* width;

}

};

int main()

{

Shape \*c1 = new Circle("Circle", 10);

cout << "The area of " << c1->getName() << " is " << c1->getArea() << endl;

Shape \*c2 = new Rectangle("Rectangle", 5.5, 10.7);

cout << "The area of " << c2->getName() << " is " << c2->getArea() << endl;

system("pause");

return 0;

}