抽象类：不能有实例对象的类（可以有指针或引用）。

抽象类中至少具有一个纯虚函数。

抽象类的作用是作为基类被继承。

例：程序class\_test28

// 纯虚函数

#include <iostream>

using namespace std;

class Display

{

public:

virtual void init() = 0; // 纯虚函数

virtual void write() = 0; // 纯虚函数

};

class Monochrome : public Display

{

public:

void init()

{

cout << "Monochrome init" << endl;

}

void write()

{

cout << "Monochrome write" << endl;

}

};

class ColorAdapter : public Display // ColorAdapter类仍然是抽象类

{

public:

void write()

{

cout << "ColorAdapter write" << endl;

}

};

int main(int argc, char\* argv[])

{

Monochrome mc;

mc.init();

// ColorAdapter ca; // error，ColorAdapter类只是实现了基类Display中的一个纯虚函数，其仍然是抽象类

return 0;

}

纯虚函数也会触发动态绑定。

例：程序class\_test43

// 纯虚函数

#include <iostream>

using std::cout;

using std::endl;

class Base

{

public:

virtual void f() = 0;

};

class Derive : public Base

{

public:

void f()

{

cout << "Derive::f()" << endl;

}

};

int main(int argc, char\* argv[])

{

Derive d;

Base& b = d;

b.f(); // 执行Derive类的f()函数

return 0;

}

纯虚函数一般只声明，没有具体的实现，但在某些情况下，却需要实现，例如纯虚析构函数。

例：程序class\_test29

// 纯虚析构函数

#include <iostream>

class Base

{

public:

Base()

{

}

virtual ~Base() = 0; // 纯虚析构函数，必须进行实现

};

Base::~Base()

{

}

class Derive : public Base

{

public:

Derive()

{

}

~Derive()

{

}

};

int main(int argc, char\* argv[])

{

Derive d;

return 0;

}

纯虚析构函数必须被实现，原因：派生类的析构函数会自动调用其基类的析构函数。这个过程是递归的，最终，抽象类的纯虚析构函数也会被调用，所以必须实现。