1. 类和对象，派生和继承，虚函数和多态性

//关与类和对象的测试

//测试.h

#pragma once

#include <cstdio>

#include <math.h>

#include <cstring>

#include <algorithm>

#include <iostream>

#include <string>

#include <queue>

#include <stack>

#include <vector>

#include <map>

#include <set>

using namespace std;

class me

{

public:

me(int a, int b, int c) { //构造函数来处理对象的初始化，构造函数必须与类名相同，在建立对象时自动执行，没有类型没有返回值，不能被用户调用，但可以带参数

age = a;

height = b;

}

me() { //构造函数的重载

age = 0;

height = 0;

}

~me() {}; //析构函数

inline void display() { //在这里直接定义的叫内置成员函数，inline可以省略

cout << name << endl;

}

inline int displayage();

string findgirlfriend(); //公共成员函数是用户使用类的公用接口，或说类的对外接口

static int sth; //静态数据成员：各对象该数据的值时一样的。（等于把他跟成员函数放在了一起）

static int count() { //静态成员函数是类的一部分，而不是对象的一部分，用来访问静态成员函数，不能访问非静态成员函数。用 类名::函数名调用

return sth;

}

private:

string name;

int age;

int height;

string lover;

};

inline int me::displayage() {

cout << age << endl;

return 0;

}

//如果在类体外定义的内置函数，必须将类定义和成员函数的定义都放在一个头文件中（或者写在一个源文件中）

//内置函数在编译时将所调用函数的代码直接嵌入到主调函数中，这种函数称为内置函数（内嵌函数，内联函数），少了调用的时空开销，更方便

string me::findgirlfriend() { //函数外定义成员函数，要加上类名me和成员限定符::

cout << lover << endl;

}

//不同对象值是不同的，代码是相同的。成员函数无论在哪定义，都不占对象的存储空间

//测试.cpp

#include <cstdio>

#include <math.h>

#include <cstring>

#include <algorithm>

#include <iostream>

#include <string>

#include <queue>

#include <stack>

#include <vector>

#include <map>

#include <set>

#include "测试.h"

using namespace std;

static int a;//全局变量中的静态外部变量 只允许本文件使用（文件内有效）

extern int num;//extern在多文件的程序中声明外部变量。对一个已经定义的外部变量做说明，以扩展其作用域。

int all;//全局变量中的外部变量（其他文件可用）

//全局变量的有效范围从定义变量的位置到“本源文件”结束

//1.在程序的全部执行过程中都站存储单元 2.降低函数的通用性，可移植性 3.降低程序的清晰性

int cal(int a) {

static int c = 1;//static声明静态局部变量（函数内有效），函数调用结束后c的值保留，内存空间不释放，下次调用时不会被static定义语句重新赋值。如果多次输入a=1则输出3 4 5 ...

c++;

return c + a;

}

int main() {

int x, y;//局部变量只在本函数范围内有效，没有static声明，编译系统为他动态分配存储空间

extern int m, n;//extern在一个文件里声明全局变量

me luoxiao(21, 181, 6);

me haoziqi;

int sth = me::sth;//可以直接通过类名引用静态数据成员

sth = luoxiao.sth;//也可以通过已定义的对象来访问

}

//全局变量局部变量重定义，以局部变量为准。

public部分都可以继承；protected部分可以继承，不能直接使用；private部分不能继承，不能直接使用

虚函数：在基类声明函数是虚拟的，并不是实际存在的函数，然后在派生类中才正式定义次函数。在声明派生类时被重载，这时派生类中的同名函数就取代了其基类中的虚函数。

用法：

1. 在基类中用virtual定义的成员函数为虚函数。在类外定义虚函数时，不必再加virtual。
2. 再派生类中重新定义此函数，函数名函数类型以及参数必须与基类相同，函数体根据派生类的需要制定。

每一个成员函数被声明成为虚函数后，其派生类中的同名函数都自动成为虚函数，因此在派生类中重新定义该函数时前面可以加virtual也可以不加，但一般为了程序更清晰，都会加。

1. 定义一个指向基类对象的指针变量，使它指向同一类族中需要调用该函数的对象，通过指针调用此虚函数，就是调用指针变量所指对象的同名函数。