中山大學
SUN YAT-SEN UNIVERSITY

程序设计II实验

电子与信息工程学院（微电子学院）

庞志勇 高级实验师 助教：焦涵 、桂海田 博士研究生

PPT大纲

- 上次课实验总结
- 本次课的实验介绍

上次课实验总结

2.下面的程序中调用了findmax()函数，该函数寻找数组中的最大元素，将该元素的下标通过参数返回并返回其地址值，用指针编程实现findmax()函数。

```
# include <iostream>
using namespace std;
int * findmax(int * array, int size, int * index);
void main ( )
{
int a[10] = {33,91,54,67,82,37,85,63,19,68};
int * maxaddr;
int idx;
maxaddr = findmax(a, sizeof(a)/sizeof( * a), &idx);
cout<<idx<<endl<<maxaddr << endl<<a[idx] << endl;
}
```

1） 输入输出分析及算法；

2） 编程源代码加注释；

3） 调试与测试；

4） 遇到的问题及解决方法

```
# include <iostream>
using namespace std;
int* findmax(int* array, int size, int* index);
void main()
{
        int a[10] = { 33,91,54,67,82,37,85,63,19,68 };
        int* maxaddr;
        int idx;
        maxaddr = findmax(a, sizeof(a) / sizeof(*a), &idx);
        int size;
        size = sizeof(a) / sizeof(*a);
        cout << size << endl;    //10

        int size1 = sizeof(a);
        cout << size1 << endl;    //40

        int size2 = sizeof(*a);
        cout << size2 << endl;    //4

        cout << idx << endl << maxaddr << endl <<
a[idx] << endl;
}
```

上次课实验总结

2.下面的程序中调用了findmax()函数，该函数寻找数组中的最大元素，将该元素的下标通过参数返回并返回其地址值，用指针编程实现findmax()函数。

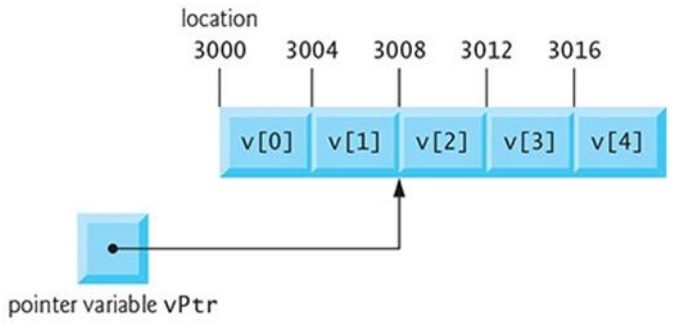
```
# include <iostream>
using namespace std;
int * findmax(int * array, int size, int * index);
void main ( )
{
int a[10] = {33,91,54,67,82,37,85,63,19,68};
int * maxaddr;
int idx;
maxaddr = findmax(a, sizeof(a)/sizeof( * a), &idx);
cout<<idx<<endl<<maxaddr << endl<<a[idx] << endl;
}
```

1） 输入输出分析及算法；

2） 编程源代码加注释；

3） 调试与测试；

4） 遇到的问题及解决方法




```
int*findmax(int*array,int size,int*index){
int* maxAdd=array;//把最大值的地址先设在第一个元素
*index=0;
for(int i=0;i<size;i++)
{
if(array[i]>*maxAdd)//如果有值比当前最大值还大的，就更换最大值地址和下标
{
maxAdd=&array[i];
*index=i;
}
}
return maxAdd;
}
```

中山大學
SUN YAT-SEN UNIVERSITY

上次课实验总结

要做自编教材哪些实验？

第4章 面向过程编程实验

- 实验一、VC6使用与cout输出程序设计
- 实验二、数据类型、常量、变量、表达式
- 实验三、输入输出流
- 实验四、选择结构程序设计
- 实验五 循环结构程序设计
- 实验六 控制结构综合实验
- 实验七 函数实验
- 实验八 作用域、生存期及函数实验
- 实验九 数组实验
- 实验十 指针实验
- 实验十一 结构体（记录）实验

课程设计I

第5章 面向对象编程实验

- 实验一 类与对象
- 实验二 函数重载与运算符重载
- 实验三 继承与派生
- 实验四 多态性与虚函数
- 实验五 模板与STL
- 实验六 流类库与文件操作
- 实验七 异常处理

课程设计II

实验内容

- 实验内容根据理论课内容进行调整
- 每次课实验内容可以根据自己学习情况动态调整
- 课程设计可以自选其他内容

实验一 类与对象

一. 实验目的

1. 理解面向对象的程序设计的特点，理解类的封装性和信息隐藏。
2. 掌握类、类数据成员、类成员函数的定义方式。
3. 理解类成员的Private、public、protect访问控制方式。
4. 掌握对象的定义创建和操作对象成员的方法。
5. 理解构造函数和析构函数的定义与使用VC++的debug调试观察执行过程。
6. 掌握重载构造函数的方法。
7. 了解拷贝构造函数的定义方法。
8. 理解掌握类的组合。

二. 实验原理

实验一、类和对象

二 、实验原理

面向过程 与 面向对象

- 数据类型
- 标识符
- 常量
- 变量
- 运算符和表达式
- 语句
- 顺序选择循环结构

函数

- 输入输出函数
- 库函数
- 自定义函数

对象

数据
函数

面向过程

面向对象

面向过程 与 面向对象

二 、实验原理

面向过程分析（Procedure Oriented Analysis）：是一种以过程为中心的编程思想，以数据流向为主要导向。为了解决问题，将解决问题的业务过程，按照一定的顺序划分成为一个又一个的事件，然后再封装成一个又一个的函数，最后由一个主函数统一的按照顺序一步一步的调用即可。在面向过程分析中，顺序很重要，要实现功能只需要按照一定的顺序相互调用函数即可。

面向过程 与 面向对象

二 、实验原理

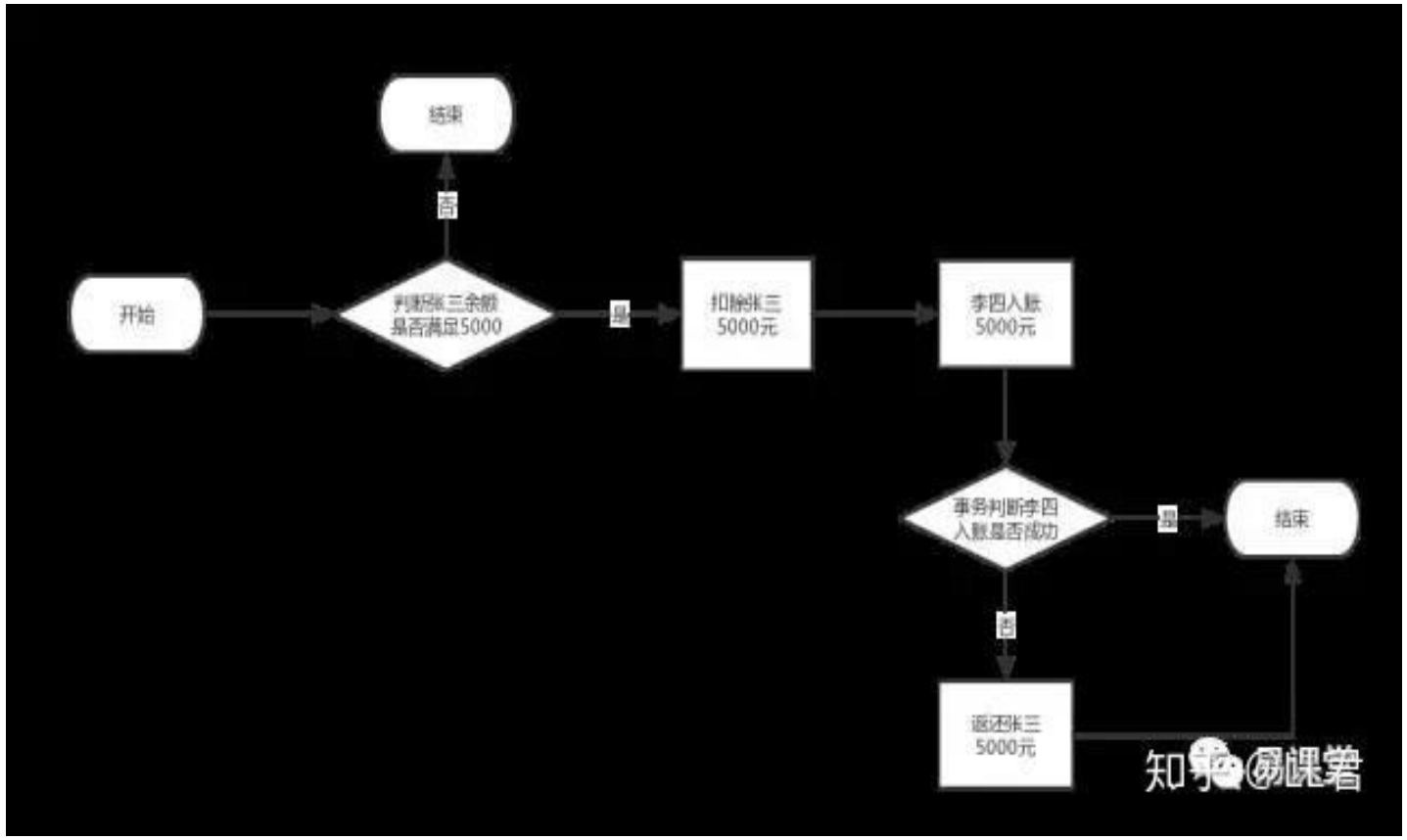
有些书籍比较官方化，让初学者无从理解。我们以一种实例的方式来阐述。

业务场景:

建行卡持有者，张三与李四两人，现在需要张三给李四转账人民币5000元整。

按照业务分析后的流程图如右图所示:

面向过程分析



https://zhuanlan.zhihu.com/p/66846761

二、实验原理

面向过程分析（Procedure Oriented Analysis）：上述业务场景按照面向过程分析出来的结果就是：

1.程序检查张三卡中余额是否足够5000元人民币（事件1，满足则调用事件2）

2.程序从张三卡中扣除5000元人民币（事件2）

3.程序向李四卡中加入5000元人民币（事件3）

4.程序检测李四卡中是否正常入账（事件4，满足则结束整个业务）

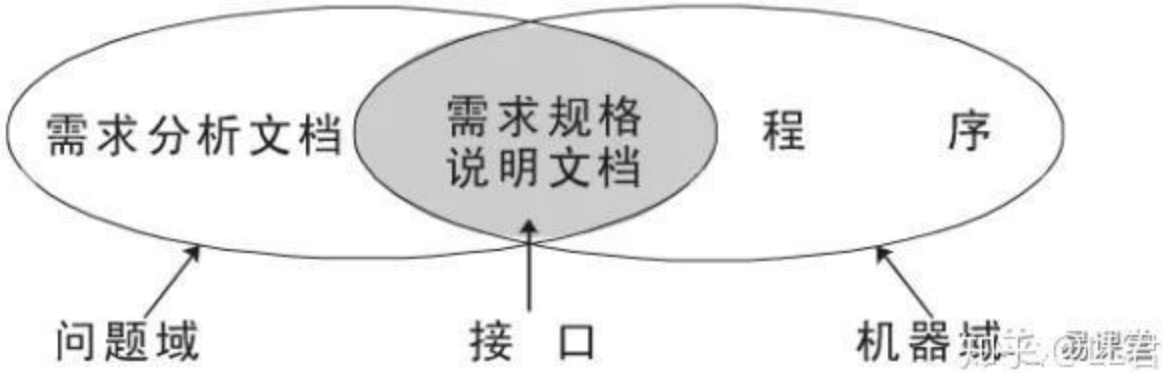
5.程序向张三卡中加入5000元人民币（事件5）

在上述过程中，我们将这个业务过程，分成了5个步骤，也叫做5个事件，那么如果需要在程序中完成该转账业务的话，那么我们只需要按照1-2-3-4-5这样的顺序依次调用函数方法即可。在这个过程中，你会发现我们函数调用的顺序一定是不能变化的，变了就出问题了……

面向过程 与 面向对象

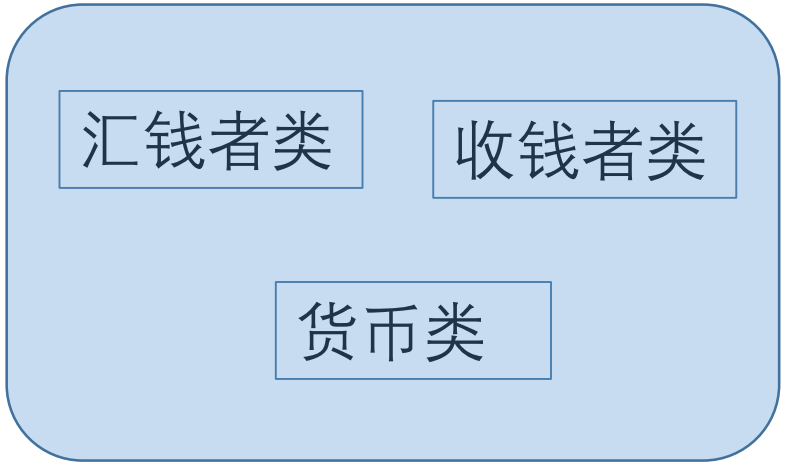
面向对象分析（Object Oriented Analysis）:是一种以对象为中心的编程思想。利用从问题域中的词汇表中找到类与对象。

用一个图来表示：需求分析文档，规格说明文档，以及程序之间的关系：



面向过程 与 面向对象

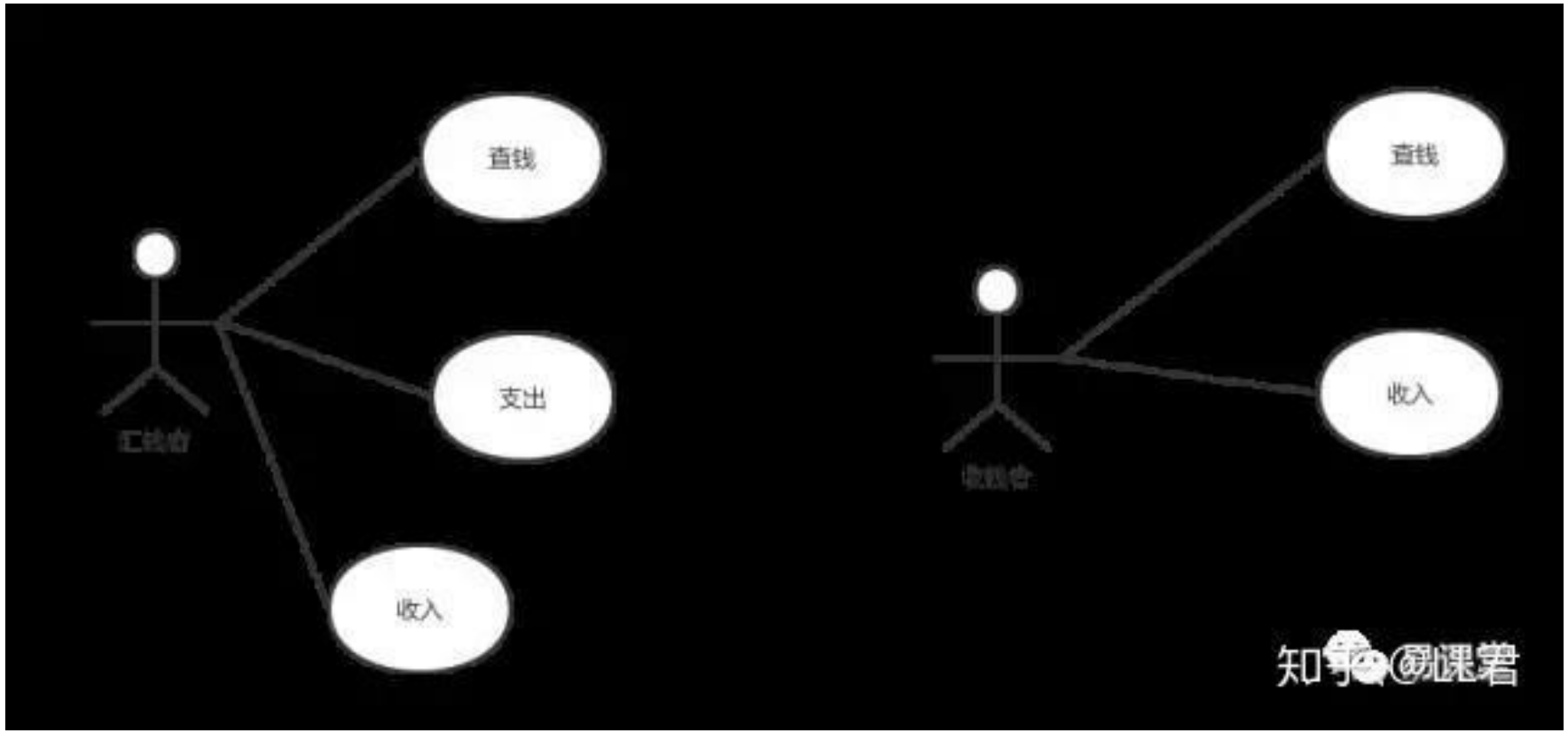
从上述业务场景的分析中，我们可以抽离出的类与对象有：



抽离出来的类型有：汇钱者类，收钱者类，货币类。而张三只能算是汇钱者类中的一个实例，也被称之为汇钱者对象，李四只能算是一个收钱者类中的一个实例，也被称之为收钱者对象。那么额度为5000的人民币也只是货币类的一个具体实例。通常汇钱者类，收钱者类，货币类，我们都统一称之为：领域模型类，张三，李四，5000元人民币我们都统一称之为：领域对象。作为面向对象分析来说，我们最为重要的就是要分析出领域对象的行为。领域对象的行为主要是为后期的面向对象设计（OOD）提供接口依据，而属性作为分析阶段不是我们的重点。

面向过程 与 面向对象

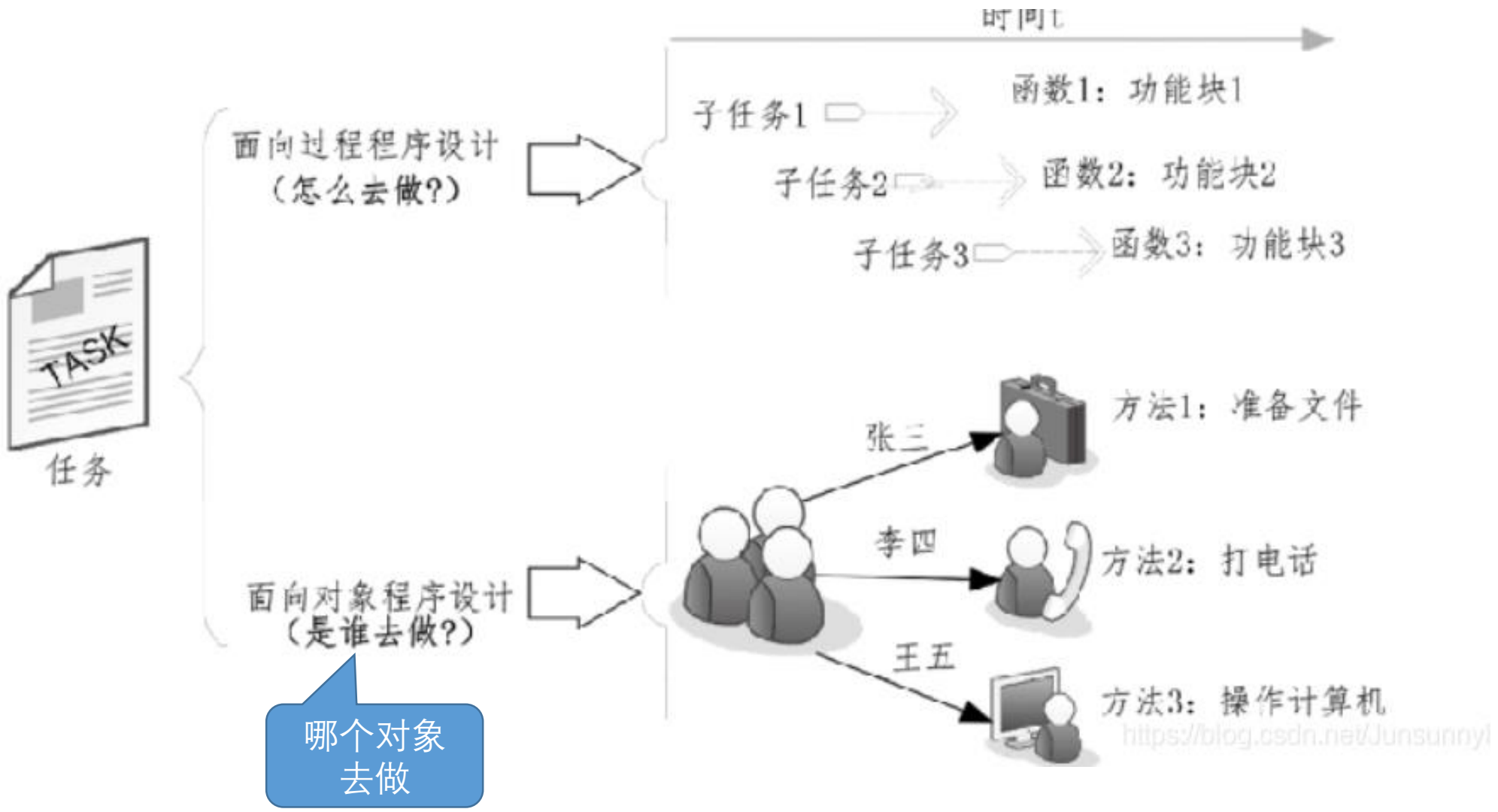
就上述3种领域对象而言，张三这个实例，可以查询自己卡中余额，可以从余额中取出5000元，也可以将外来的钱加入到自己的账户中。
李四这个实例，可以查询自己卡中余额，可以将外来的钱加入到自己的账户中。而5000元人民币只是数据的传输携带者，没有任何行为可言。



面向过程 与 面向对象

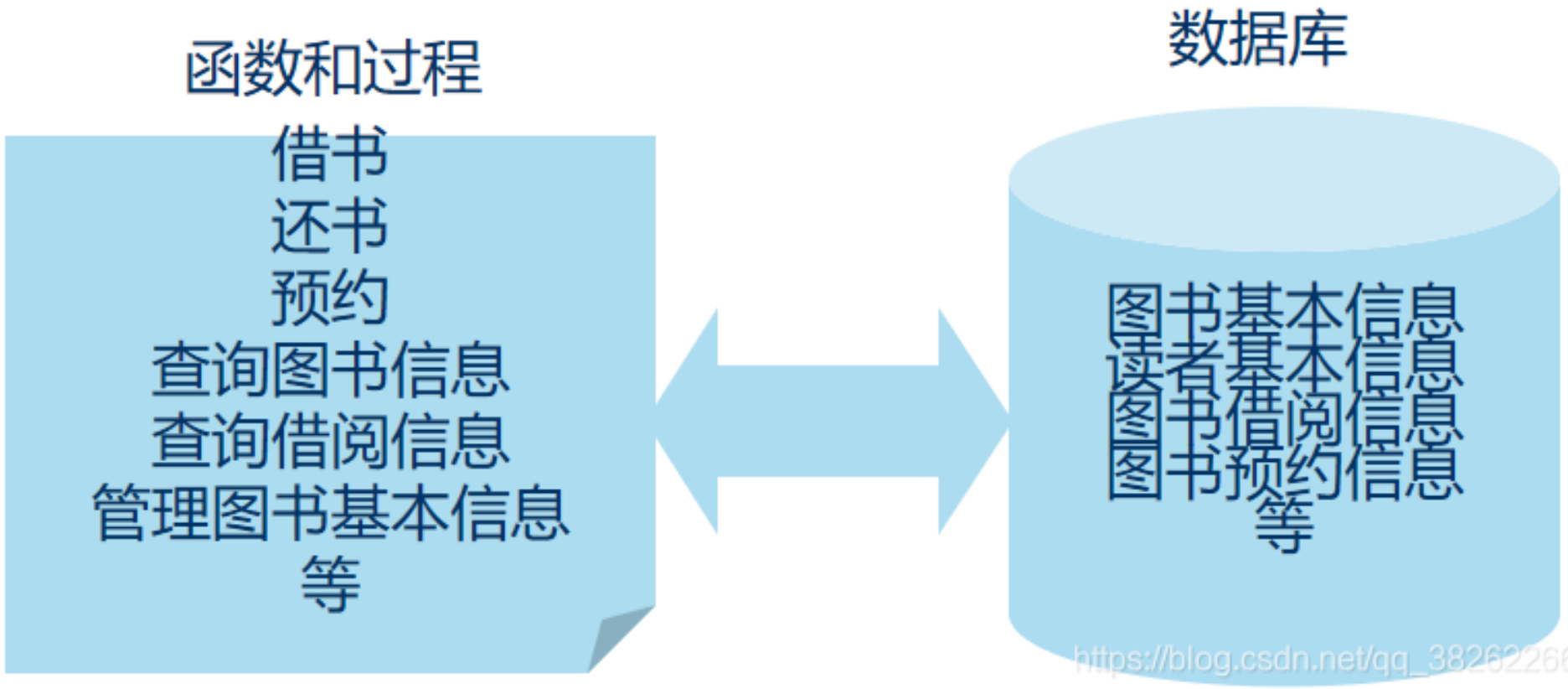
- 面向过程就是分析出解决问题所需要的步骤，然后用函数把这些步骤一步一步实现，使用的时候一个一个依次调用就可以了;
- 面向对象是把构成问题事务分解成各个对象，建立对象的目的不是为了完成一个步骤，而是为了描叙某个事物在整个解决问题的步骤中的行为。

面向过程 与 面向对象


时间t
面向过程程序设计
（怎么去做?）
子任务1
函数1：功能块1
子任务2
函数2：功能块2
子任务3
函数3：功能块3
TASK
任务
面向对象程序设计
（是谁去做?）
张三
方法1：准备文件
李四
方法2：打电话
王五
方法3：操作计算机
哪个对象
去做

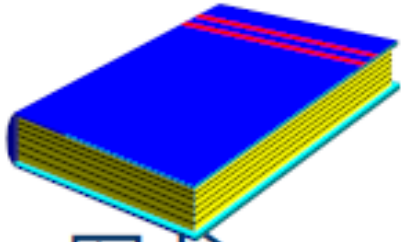
面向过程 与 面向对象

- 结构化方法：



面向过程 与 面向对象

- 面向对象的方法：



图书

读者

图书管理员

现实世界中的对象

面向过程 与 面向对象

学生	图书	教师	管理员
数据： 姓名、学号、班级、入学年份、宿舍、电话等	数据： 书名、ISBN号、出版社、出版日期、作者等	数据： 姓名、工作证号、所在学院、入职年份、职称、电话等	数据： 姓名、工作证号、入职时间
功能： 借书、还书、预约、查询图书信息、查询借阅信息、交罚款等	功能： 告知图书状态、查询借阅时间、查询借阅该书读者信息等	功能： 借书、还书、预约、查询图书信息、查询借阅信息、交罚款等	功能： 图书入库 图书出库 办理借书/还书 收罚款

软件世界中的对象

https://blog.csdn.net/qq_38262266/article/details/86637199

面向对象=对象+类+继承+通信

可见，面向对象不仅是一些具体的软件开发技术与策略，而且是一整套关于如何看待软件系统与现实世界的关系、用什么观点来研究问题并进行问题求解、以及如何进行系统构造的软件方法学。

Coad和Yourdon给出了一个定义：
“面向对象=对象+类+继承+通信”。

如果一个软件系统是使用这样 4 个概念设计和实现的，则我们认为这个软件系统是面向对象的。
一个面向对象的程序的每一成份应是对象，计算是通过新的对象的建立和对象之间的通信来执行的。

面向对象基本思想

面向对象方法是一种运用对象、类、继承、封装、聚合、关联、消息、多态性等概念来构造系统的软件开发方法。

1、从现实世界中客观存在的事物出发来构建软件系统，强调直接以问题域（现实世界）中的事物为中心来思考问题、认识问题，并根据这些事物的本质特征和要解决的具体问题域语境，把它们抽象地表示为软件系统中的对象，作为软件系统的基本构成单位。(对象（object）)

2、用对象的属性表示事物的静态特征；用对象的服务（操作）表示事物的动态特征
（属性（attribute）与服务（operation））

3、对象的属性与服务结合为一体，成为一个独立的、不可分的实体，对外屏蔽其内部细节。
（对象的封装(encapsulation)）

面向对象基本思想

面向对象方法是一种运用对象、类、继承、封装、聚合、关联、消息、多态性等概念来构造系统的软件开发方法。

4、对事物进行分类。把具有相同属性和服务的对象归为一类，类是这些对象的抽象描述，每个对象是它的类的一个实例。（分类(classification)）

5、类具有封闭性，把内部的属性和服务隐藏起来，只有公共的服务对外是可见的。(类的封闭性)

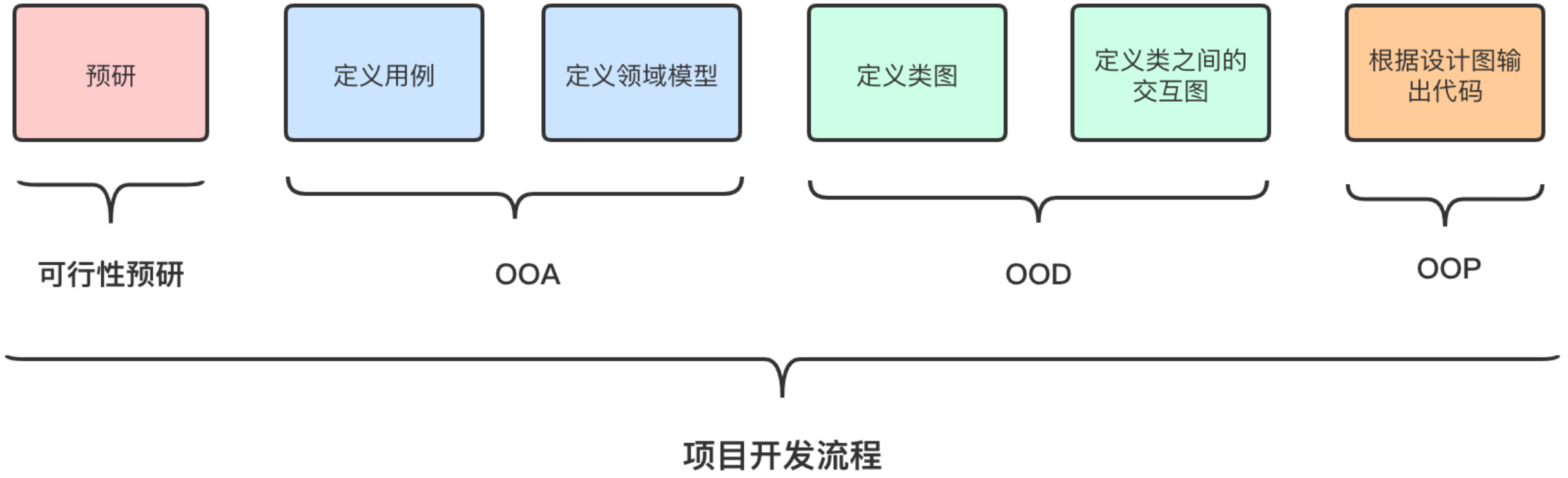
6、复杂的对象可以用简单的对象作为其构成部分。（聚合(aggregation)）

7、通过在不同程度上运用抽象的原则，可以得到较一般的类和较特殊的类。特殊类继承一般类的属性与服务，从而简化系统的构造过程及文档。（继承(inheritance)）

8、通过关联表达类之间的静态关系（关联（association））

9、对象之间通过消息进行通信，以实现对象之间的动态联系。（消息（message））

面向对象项目开发流程



OOA全拼为Object-Oriented Analysis，面向对象分析。

OOD全拼为Object-Oriented Design，面向对象设计。

OOP全程为Object-oriented programming，面向对象编程。

OOA方法具体步骤

在用OOA具体分析一个事物时。大致上遵循如下5个基本步骤;

1，确定对象和类。这里所说的对象是对数据及其处理方式的抽象，它反映了系统保存和处理现实世界总某些事物的信息能力。类是多个对象的共同属性和方法集合的描述，它包括如何在一个类中建立一个新对象的描述。

2，确定结构（structure）。结构是指问题域的复杂性和连接关系。类成员结构反映了泛华—特化关系，整体-部分结构反映整体和局部之间的关系

3，确定主题（subject）。主题是指事物的总体概貌和总体分析模型

4，确定属性（attribute）。属性就是数据元素，可用来描述对象或分类结构的实例，可在图中给出，并在对象的存储中指定。

5，确定方法（method）。方法是在收到消息后必须进行的一些处理方法：方法要在图中定义，并在对象的存储中指定。对于每个对象和结构来说，那些用来增加、修改、删除和选择一个方法本身都是隐含的（虽然它们是要在对象的存储中定义的，但并不在图上给出），而有些则是显示的

OOD:面向对象设计

面向对象设计（Object-Oriented Design，OOD）方法是OO方法中一个中间过渡环节。其主要作用是对OOA分析的结果作进一步的规范化整理，以便能够被OOP直接接受。

OOP:面向对象编程

面向对象编程（Object Oriented Programming，OOP，面向对象程序设计）是一种计算机编程架构。OOP 的一条基本原则是计算机程序是由单个能够起到子程序作用的单元或对象组合而成。OOP 达到了软件工程的三个主要目标：重用性、灵活性和扩展性。为了实现整体运算，每个对象都能够接收信息、处理数据和向其它对象发送信息。

OOD五大原则.

S = 单一职责原则 Single Responsibility Principle
O = 开放闭合原则 Opened Closed Principle
L = Liscov替换原则 Liscov Substitution Principle
I = 接口隔离原则 Interface Segregation Principle
D = 依赖倒置原则 Dependency Inversion Principle

类和对象

- **类：**具有同种属性的对象称为类。比如，“人”就是一类，其中的人名比如小明，小红等都是对象。类相当于一个模板，他定义了它所包含的全体对象的公共特征和功能，对象是类的实例化。所以我们一般在做程序的时候一般都不用类名的，比如我们在叫小明的时候，不会喊“人，你干嘛呢！”而是说的是“小明，你在干嘛呢！”
- **对象：**现实中任何事物都可以称之为对象，有自己的独特的特点。 属性是用来描述具体某个对象的特征。例如小明身高180，体重70千克，身高和体重就是属性。 面向对象的思想就是把一切事物都看成对象，而对象一般都是由属性和方法组成。 属性属于对象静态的一面，用来形容对象的一些特性。 方法属于对象动态的一面，例如，小明会跑，会说话。跑，说话这些行为就是对象的方法!
- 面向对象有三大特性：封装性，继承性和多态性

类和对象

- C++ 在 C 语言的基础上增加了面向对象编程，C++ 支持面向过程和面向对象程序设计。类是 C++ 的核心特性，通常被称为用户定义的类型。

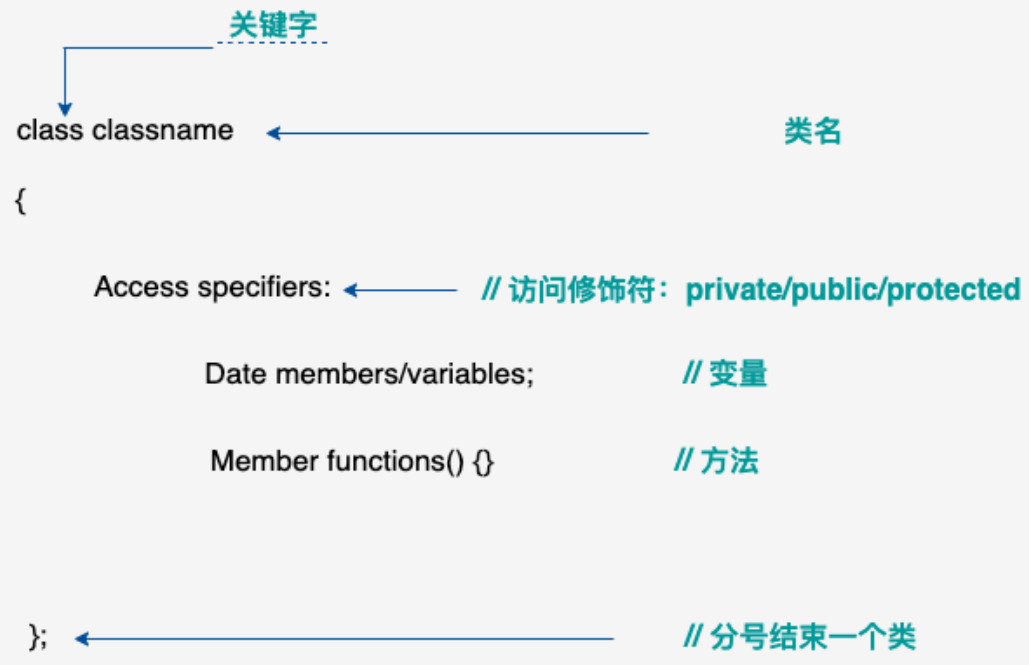
- 类用于指定对象的形式，是一种用户自定义的数据类型，它是一种封装了数据和函数的组合。类中的数据称为成员变量，函数称为成员函数。

- 类可以被看作是一种模板，可以用来创建具有相同属性和行为的多个对象。

类和对象

A class is a type whose variables are objects. These objects can have both member variables and member functions.

类是一种类型，其变量是对象。类和对象既可以有成员变量，也可以有成员函数。类定义的语法如下：



Classes and Objects

A **class** is a type whose variables are **objects.** These objects can have both member variables and member functions. The syntax for a class definition is as follows.

SYNTAX

```
class Class_Name
{
public:
    Member_Specification_1
    Member_Specification_2
            .
            .
            .
    Member_Specification_n
private:
    Member_Specification_n+1
    Member_Specification_n+2
        .
        .
        .
};
```

Each *Member_Specification_i* is either a member variable declaration or a member function declaration. (Additional *public* and *private* sections are permitted.)

EXAMPLE

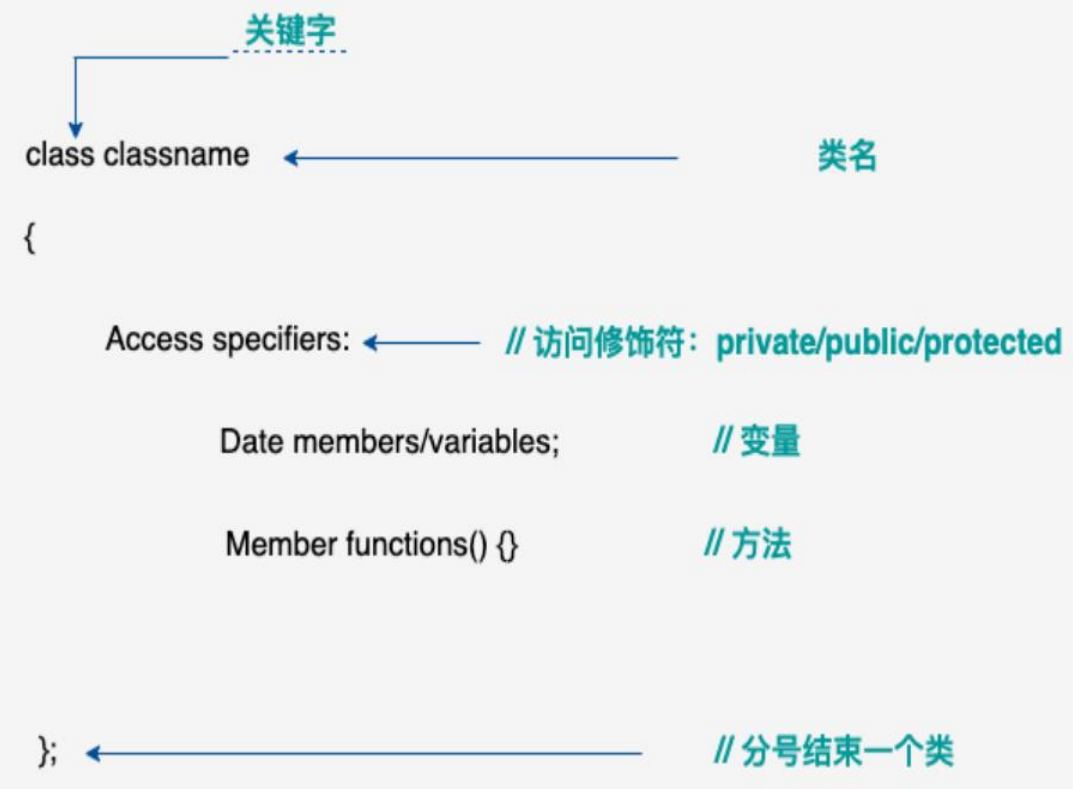
```
class Bicycle
{
public:
    char get_color();
    int number_of_speeds();
    void set(int the_speeds, char the_color);
private:
    int speeds;
    char color;
};
```

Once a class is defined, an object (which is just a variable of the class type) can be declared in the same way as variables of any other type. For example, the following declares two objects of type `Bicycle`:

```
Bicycle my_bike, your_bike;
```

类和对象

➢ 实验原理—C++类和对象语法格式




```
class Box
{
  public:
    double length;   // 盒子的长度
    double breadth;  // 盒子的宽度
    double height;   // 盒子的高度
};
```

定义 C++ 对象

```
Box Box1;        // 声明 Box1，类型为 Box
Box Box2;        // 声明 Box2，类型为 Box
```

对象 Box1 和 Box2 都有它们各自的数据成员。

类定义是以关键字 class 开头，后跟类的名称。类的主体是包含在一对花括号中。类定义后必须跟着一个分号或一个声明列表。

类提供了对象的蓝图，对象是根据类来创建的。声明类的对象，就像声明基本类型的变量一样。

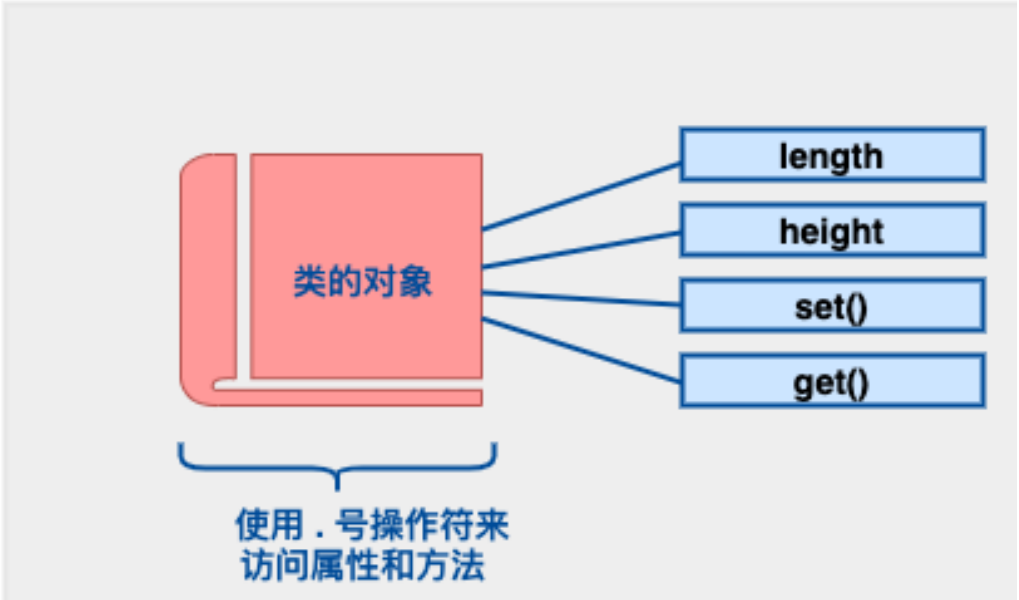
前面实验我们已经用到的“类和对象”

- Cin
- Cout
- Ifstream
- string

- 类是数据类型，其变量就是对象，例如：
  - Int int1;
  - Ifstream　infile;
  - String　　str1;
- 对象是变量，其除了有数据变量（成员变量），还有函数（成员函数）

类和对象举例

类的对象的公共数据成员可以使用直接成员访问运算符 . 来访问。



面向对象设计多举生活中的实例“对象”进行编程练习，例如理论课学生类、时间类、图形类、游戏类。。

```
#include <iostream>
 using namespace std;

class Box
{
   public:
      double length;   // 长度
      double breadth;  // 宽度
      double height;   // 高度
      // 成员函数声明
      double get(void);
      void set( double len, double bre, double hei );
};
// 成员函数定义
double Box::get(void)
{
    return length * breadth * height;
}

void Box::set( double len, double bre, double hei)
{
    length = len;
    breadth = bre;
    height = hei;
}
```

类和对象举例

```
int main( )
{
   Box Box1;        // 声明 Box1，类型为 Box
   Box Box2;        // 声明 Box2，类型为 Box
   Box Box3;        // 声明 Box3，类型为 Box
   double volume = 0.0;     // 用于存储体积

   // box 1 详述
   Box1.height = 5.0;
   Box1.length = 6.0;
   Box1.breadth = 7.0;

   // box 2 详述
   Box2.height = 10.0;
   Box2.length = 12.0;
   Box2.breadth = 13.0;

   // box 1 的体积
   volume = Box1.height * Box1.length * Box1.breadth;
   cout << "Box1 的体积：" << volume <<endl;

   // box 2 的体积
   volume = Box2.height * Box2.length * Box2.breadth;
   cout << "Box2 的体积：" << volume <<endl;


   // box 3 详述
   Box3.set(16.0, 8.0, 12.0);
   volume = Box3.get();
   cout << "Box3 的体积：" << volume <<endl;
   return 0;
}
```

赋值符号 = 和 相等符号 ==

◆ **赋值符号=**

It is perfectly legal to use the assignment operator = with objects or with structures.

suppose that the objects due_date and tomorrow are declared as follows:

```
DayOfYear due_date, tomorrow;
```

The following is then perfectly legal (provided the member variables of the object tomorrow have already been given values):

```
due_date = tomorrow;
```

The previous assignment is equivalent to the following:

```
due_date.month = tomorrow.month;
due_date.day = tomorrow.day;
```

Moreover, this is true even though the member variables named month and day are private members of the class DayOfYear 运算符= 会自动应用于对象。

◆ **相等符号 ==**

The operator == can be used to test two values of a simple type to see if they are equal. Unfortunately, the predefined operator == does not automatically apply to objects.运算符==可以用来测试一个简单类型的两个值，看看它们是否相等。不幸的是，预定义的运算符==不会自动应用于对象。

充分利用网络资源进行学习

下面的列表中还列出了其他一些 C++ 类和对象相关的概念，可以点击相应的链接进行学习。用生活中简单易懂的“对象”进行编程练习。

概念	描述
类成员函数	类的成员函数是指那些把定义和原型写在类定义内部的函数，就像类定义中的其他变量一样。
类访问修饰符	类成员可以被定义为 public、private 或 protected。默认情况下是定义为 private。
构造函数 & 析构函数	类的构造函数是一种特殊的函数，在创建一个新的对象时调用。类的析构函数也是一种特殊的函数，在删除所创建的对象时调用。
C++ 拷贝构造函数	拷贝构造函数，是一种特殊的构造函数，它在创建对象时，是使用同一类中之前创建的对象来初始化新创建的对象。
C++ 友元函数	**友元函数**可以访问类的 private 和 protected 成员。
C++ 内联函数	通过内联函数，编译器试图在调用函数的地方扩展函数体中的代码。
C++ 中的 this 指针	每个对象都有一个特殊的指针 **this**，它指向对象本身。
C++ 中指向类的指针	指向类的指针方式如同指向结构的指针。实际上，类可以看成是一个带有函数的结构。
C++ 类的静态成员	类的数据成员和函数成员都可以被声明为静态的。

https://www.runoob.com/cplusplus/cpp-classes-objects.html

抽象

面向对象程序设计方法的基本特点：抽象、封装、继承和多态。这里我们简单介绍一下，抽象和封装。

抽象： 面向对象方法中的抽象是指对具体问题即对象进行概括，抽出一类对象的共性并加以描述的过程。面向对象的软件开发中，首先应该对要解决的问题抽象成类，然后才是解决问题的过程。抽象有两个方面：数据抽象（数据结构）和行为抽象（算法）。数据抽象是描述某类对象的属性或状态，行为抽象是描述某类对象的共同行为或共同功能。

就拿时钟举例，要实现有关时钟的程序，首先要对时钟进行抽象。时钟有时、分、秒，我们用三个整型变量来存储，这就是数据抽象（数据结构）。时钟有显示时间和设置时间等功能，这就是行为抽象（算法）。用C++语言描述就是：

时钟（Clock）：

数据抽象：

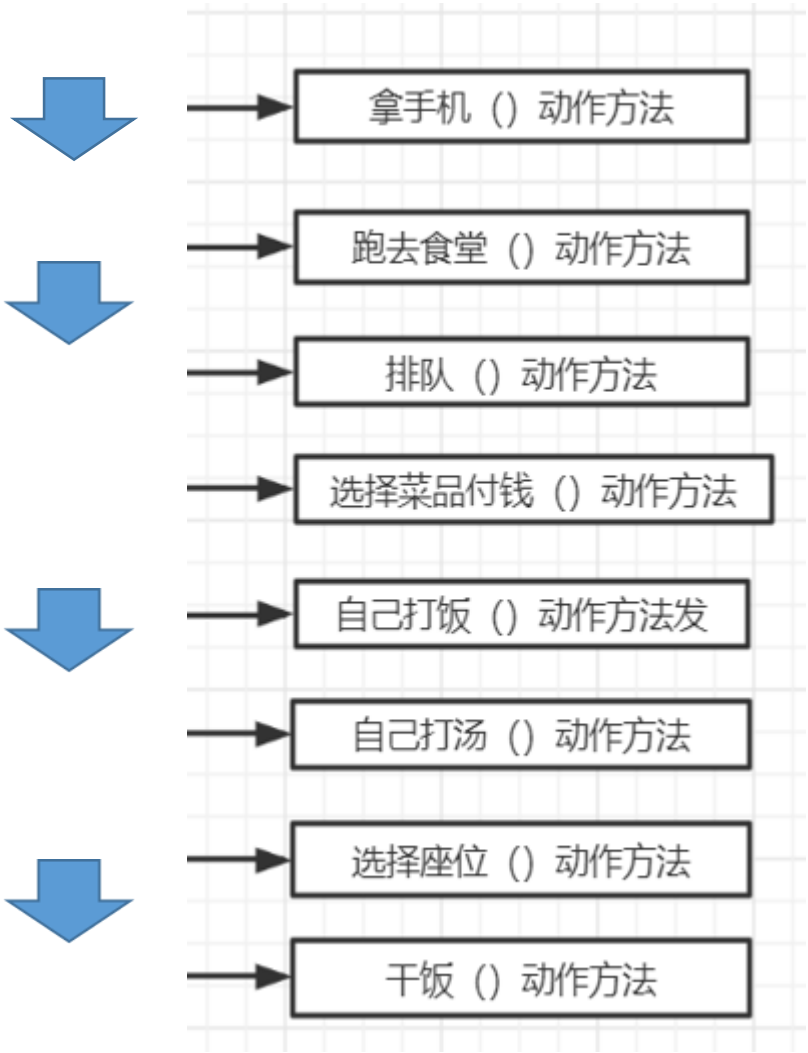
int Hout; int Minute; int Second;

行为抽象：

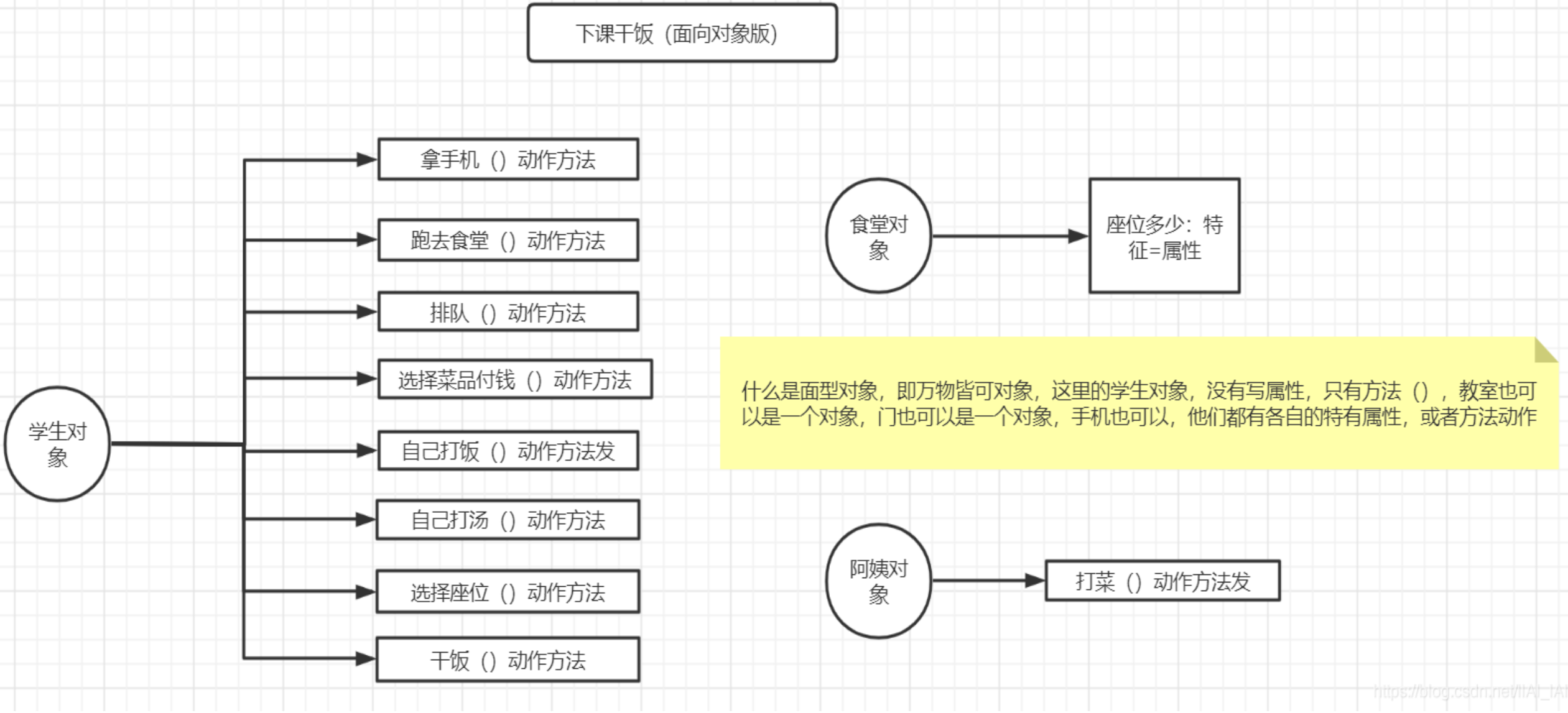
ShowTime(); SetTime();

上面并不是真正的C++代码，只是简单的列出了数据成员和函数成员的代码片段。

面向过程程序设计实例(吃饭)



面向对象程序设计实例(吃饭)



三、 实验内容1

1.用C++编写程序：请定义一个矩形类(Rectangle)，私有数据成员为矩形的长度(len)和宽度(wid)，包括求矩形周长、求矩形面积、取矩形长度和宽度、修改矩形长度和宽度为对应形参的值、输出矩形尺寸等公有成员函数。要求输出矩形尺寸的格式为“length：长度，width：宽度”。编写主函数对定义的类进行测试。

```
// classes example
#include <iostream>
using namespace std;

class Rectangle {
    int width, height;
  public:
    void set_values (int,int);
    int area() {return width*height;}
};

void Rectangle::set_values (int x, int y) {
  width = x;
  height = y;
}
int main () {
  Rectangle rect;
  rect.set_values (3,4);
  cout << "area: " << rect.area();
  return 0;
}
```

三、 实验内容2

定义一个类来表示银行帐户。包括以下成员:

数据成员: 1） 存款人姓名2）账号3）账户类型4）账户余额。

成员函数: 1） 分配初始值2）存入金额3）检查余额后提取金额4）显示姓名和余额。

编写一个主程序来测试程序。

```
Enter Details:
-----------------------
Accout No. 45789123

Name : Mangala

Account Type : Saving

Balance : 50000

Enter Deposit Amount = 5000

Enter Withdraw Amount = 10000

-----------------------
Accout No. : 45789123
Name : Mangala
Account Type : Saving
Balance : 45000
```

- UML(Unified Modeling Language) 统一建模语言，又称标准建模语言。是用来对软件密集系统进行可视化建模的一种语言。UML类图，属于UML图中的一种，是在面向对象语言用中用来表示一个类的图形

- 设计一个类Account，要将mName、mSN、mBalance均作为其成员数据，将deposit、withdraw、getBalance均作为其成员函数。类图UML设计图如右图

- 三个格子从上至下分别表示：
  - 类名（如果是接口，就使用斜体表示）
  - 类的特性/成员变量（一般是类的字段和属性,可没有这一行）
  - 类的操作/成员函数（一般是类的方法或行为）默认构造函数、带参数的构造函数、复制构造函数、析构函数

它们前边的符号有以下几类：“+”表示public，“-”表示private，“#”表示protected

银行账户程序的 Account 类的 UML 图

```
//Program to demonstrate the class BankAccount.
#include <iostream>
using namespace std;

//Class for a bank account:
class BankAccount
{
public:
    void set(int dollars, int cents, double rate);
    //Postcondition: The account balance has been set to $dollars.cents;
    //The interest rate has been set to rate percent.

    void set(int dollars, double rate);
    //Postcondition: The account balance has been set to $dollars.00.
    //The interest rate has been set to rate percent.

    void update( );
    //Postcondition: One year of simple interest has been
    //added to the account balance.

    double getBalance( );
    //Returns the current account balance.

    double getRate( );
    //Returns the current account interest rate as a percentage.

    void output(ostream& outs);
    //Precondition: If outs is a file output stream, then
    //outs has already been connected to a file.
    //Postcondition: Account balance and interest rate have
    //been written to the stream outs.
private:
    double balance;
    double interestRate;

    double fraction(double percent);
    //Converts a percentage to a fraction. For example, fraction(50.3)
    //returns 0.503.
};
```

The member function set is overloaded.

```
int main( )
{
    BankAccount account1, account2;
    cout << "Start of Test:\n";
    account1.set(123, 99, 3.0);
    cout << "account1 initial statement:\n";
    account1.output(cout);
    account1.set(100, 5.0);
    cout << "account1 with new setup:\n";
    account1.output(cout);
```

Calls to the overloaded member function set

(continued)

```
    account1.update( );
    cout << "account1 after update:\n";
    account1.output(cout);

    account2 = account1;
    cout << "account2:\n";
    account2.output(cout);
    return 0;
}
```

请同学们把类的定义补充完整，并进行调试测试。

多读多写

- ✓ 同一道程序设计题目，自己设计好之后，多在网上书上找其他人的代码，进行优缺点比较，并且在自己PC上调试运行体验
- ✓ 如果PPT上已经给了程序实现代码，尝试着进行改进。。。

测试结果

```
Enter Details:
----------------------
Accout No. 45789123

Name : Mangala

Account Type : Saving

Balance : 50000

Enter Deposit Amount = 5000

Enter Withdraw Amount = 10000

----------------------
Accout No. : 45789123
Name : Mangala
Account Type : Saving
Balance : 45000
```

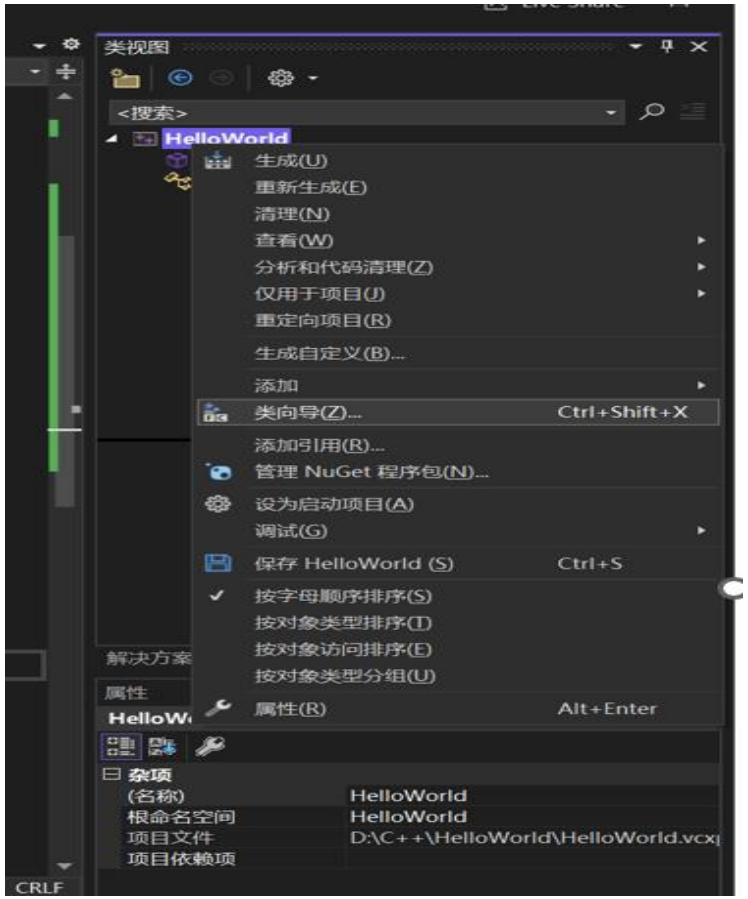
思考：还有哪些改进？

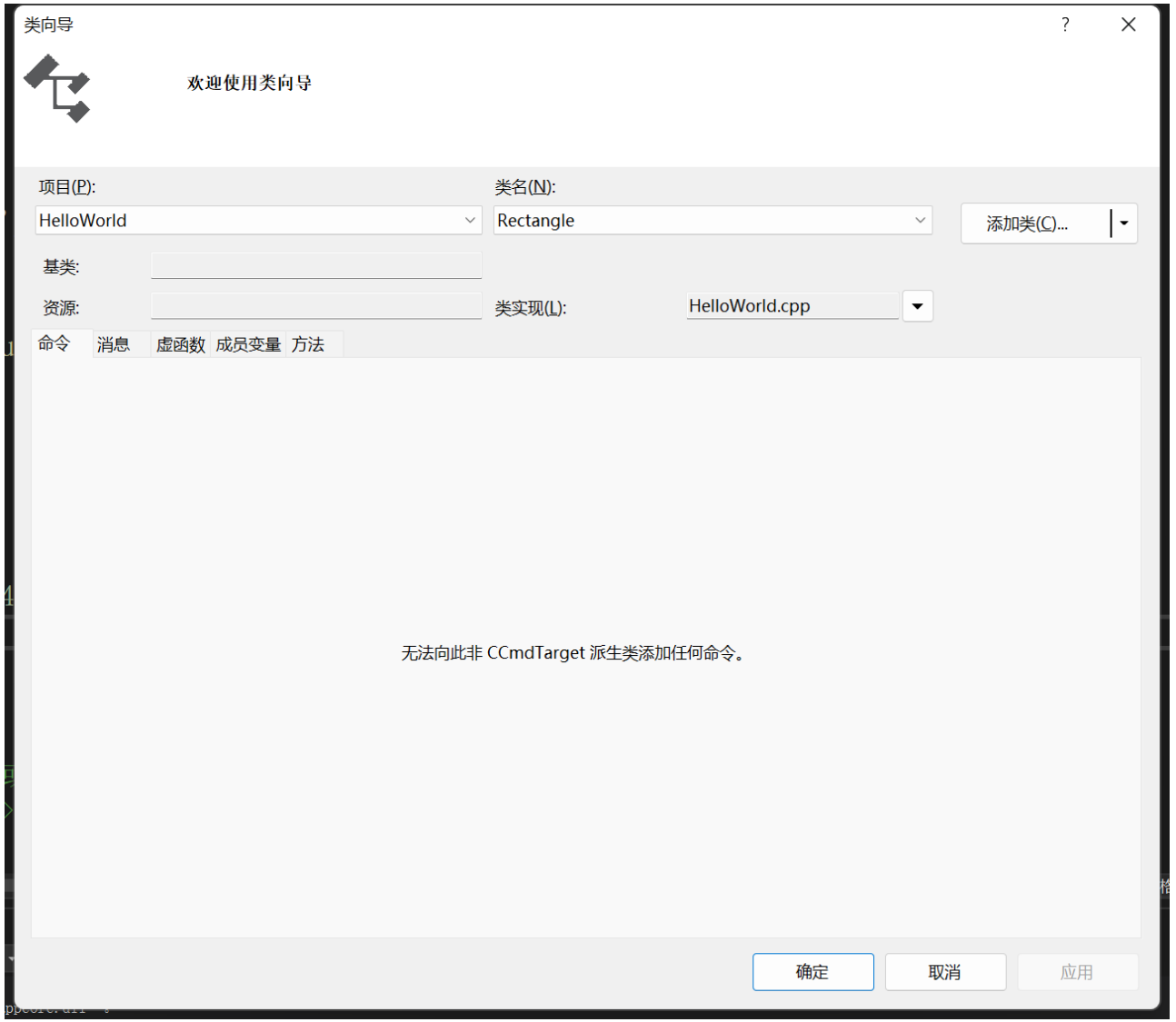
- ✓ 增加利率功能
- ✓ 增加定期存款、活期存款功能

VS2022类向导

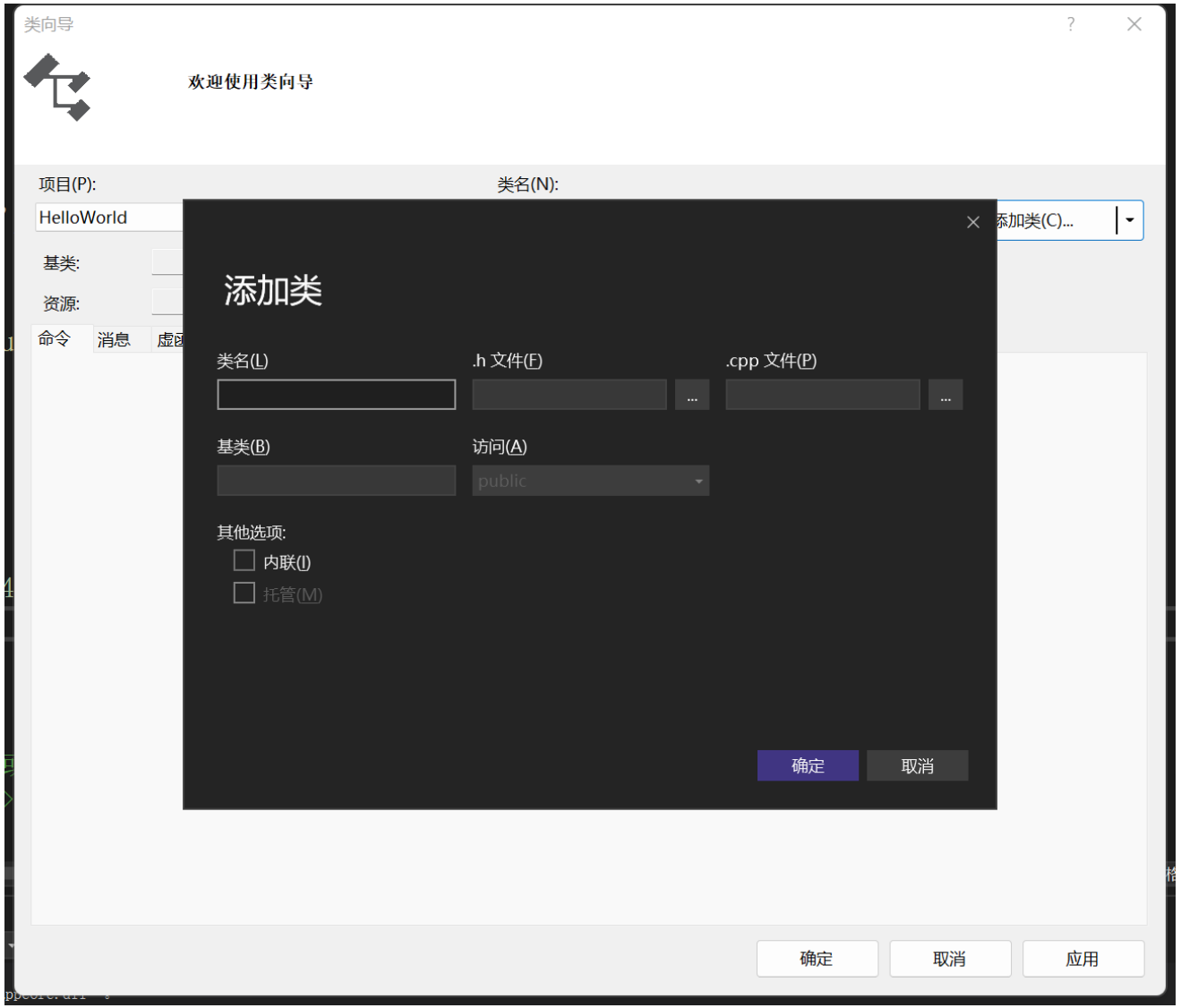
1.我们找到项目文件就能开始使用。

2.然后我们右击就能看到类向导按钮。





VS2022类向导


类向导
欢迎使用类向导
项目(P):
HelloWorld
类名(N):
添加类(C)...
基类:
资源:
命令
消息
添加类
类名(L)
.h 文件(F)
.cpp 文件(P)
基类(B)
访问(A)
public
其他选项:
内联(I)
托管(M)
确定
取消
确定
取消
应用

VS2022类向导实现point类

头文件1：point.h

```
#ifndef POINT_H
#define POINT_H
#include <iostream>
using namespace std;
class Point
{
        public:
        Point(float =0,float =0);
        void setPoint(float,float);
        float getX() const{return x;}
        float getY() const{return y;}
        friend ostream &operator<<(ostream &,const
Point &);
        protected:
                float x,y;
};
#endif
```

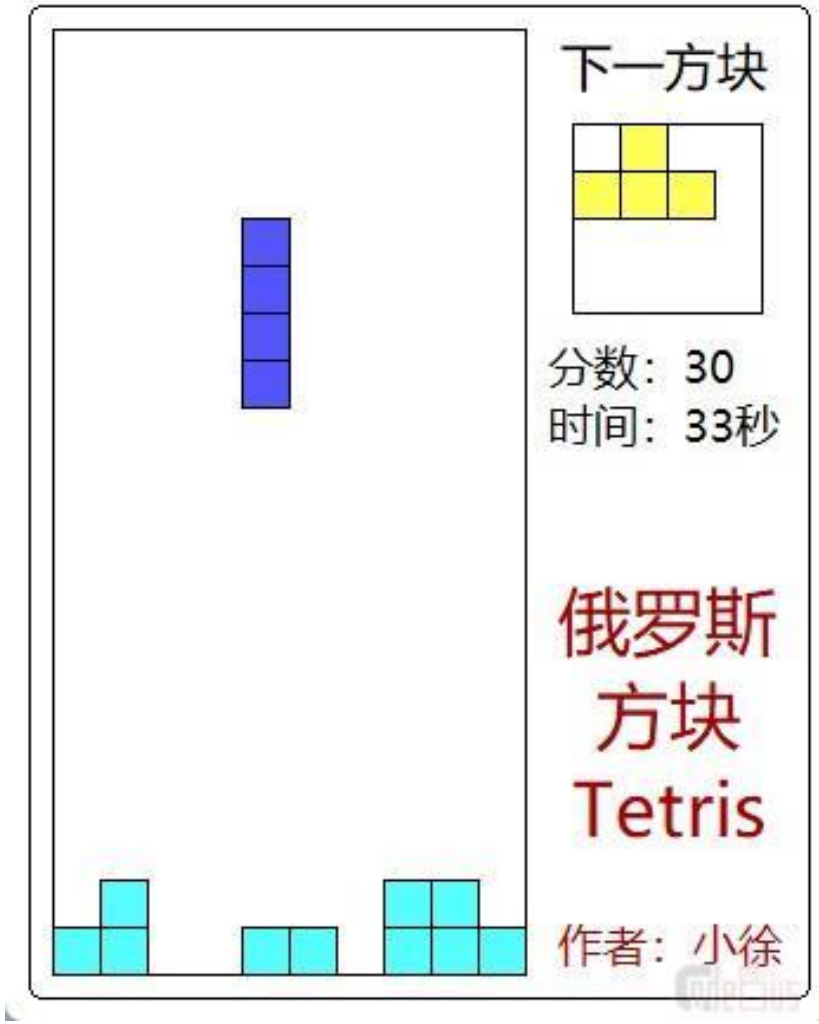
源文件1：point.cpp

```
#include"point.h"
Point::Point(float a,float b)
{
        x=a;
        y=b;
}
void Point::setPoint(float a,float b)
{
        x=a;
        y=b;
}
ostream &operator<<(ostream &output,const Point &p)
{
        output<<"["<<p.x<<","<<p.y<<"]"<<endl;
        return output;
}
```

多读多写

https://codebus.cn/qtlittlexu/tetris

使用 C++ 和 EasyX 写的一个
俄罗斯方块小游戏



学习的四重境界

知学、好学、会学、乐学

祝大家实验顺利！

OOD 设计实例

问题： OOD 思想设计两人投掷硬币的游戏。

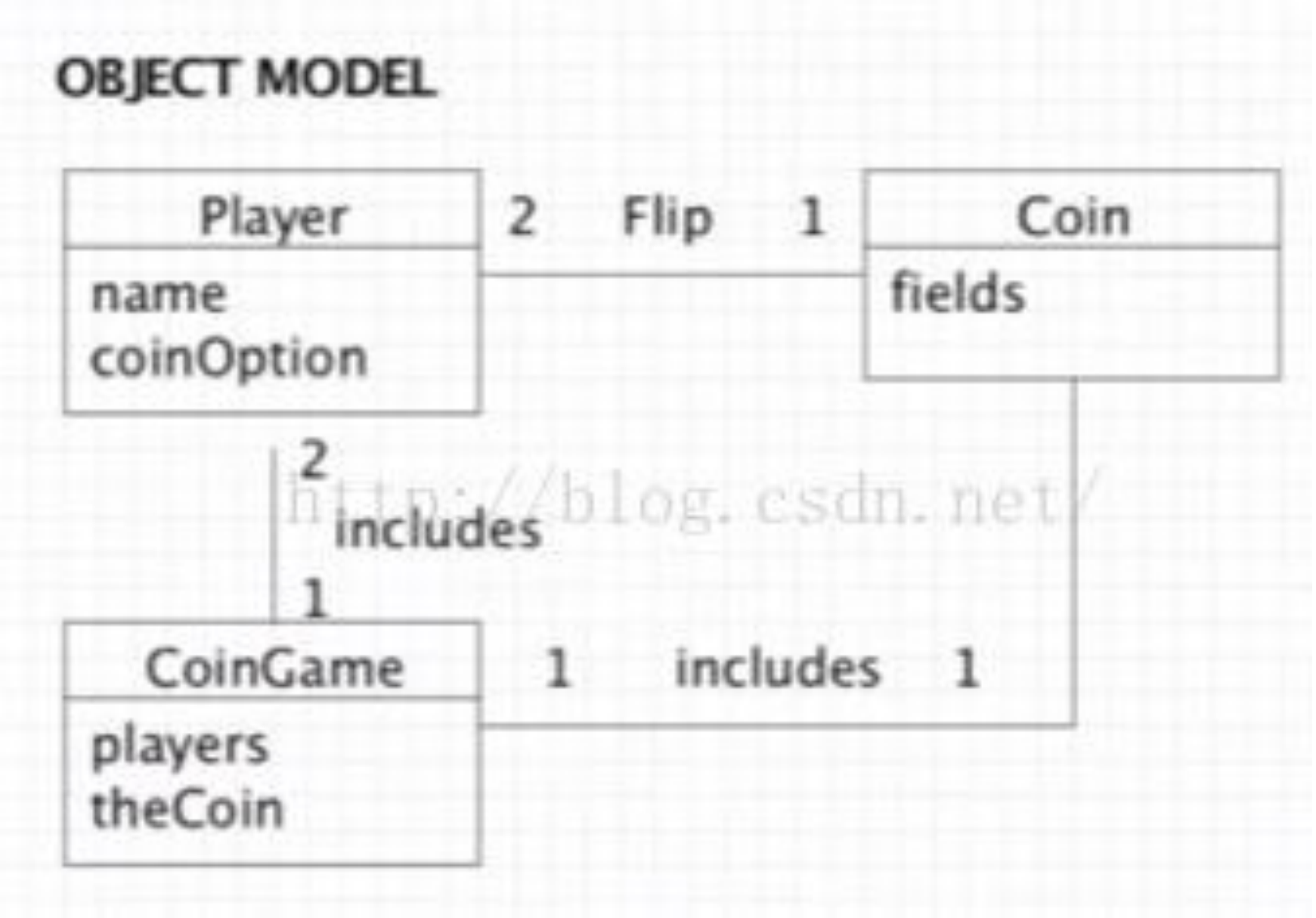
步骤一： 准备documentation游戏介绍：随机选择一个玩家，令该玩家预测硬币的正反面，另一个玩家获得和第一个玩家相反的选项。开始投掷硬币，预测正确的赢。

Actors： 1. 玩家A 2. 玩家B 3. 硬币 4. 掷硬币游戏

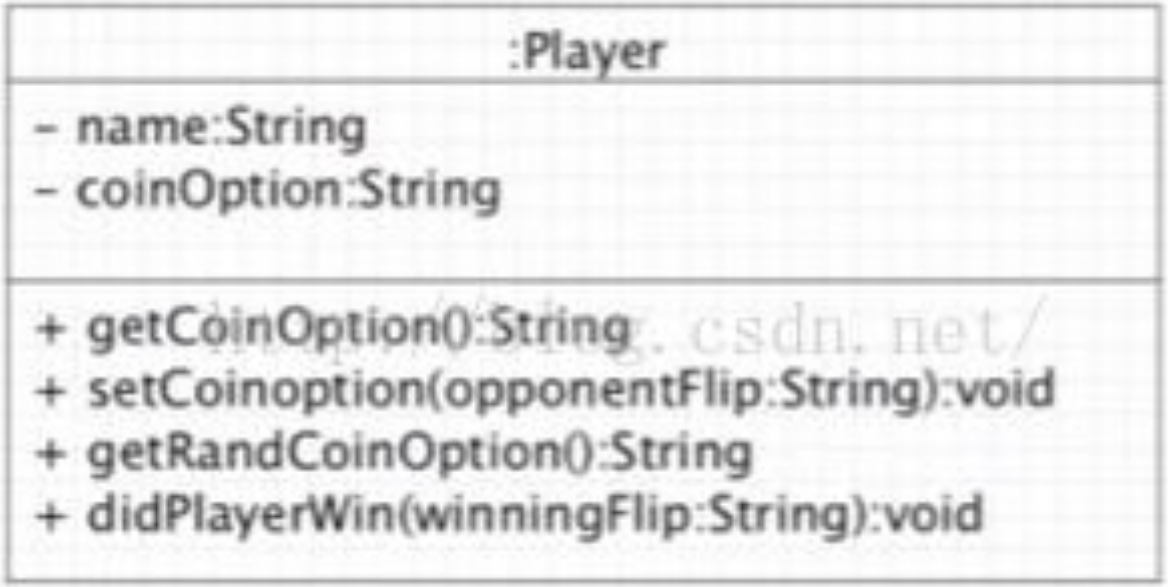
根据以上信息我们可以画出object model. 这个决定我们要创建多少个实体类。

原文链接：https://blog.csdn.net/YtdxYHZ/article/details/51432620

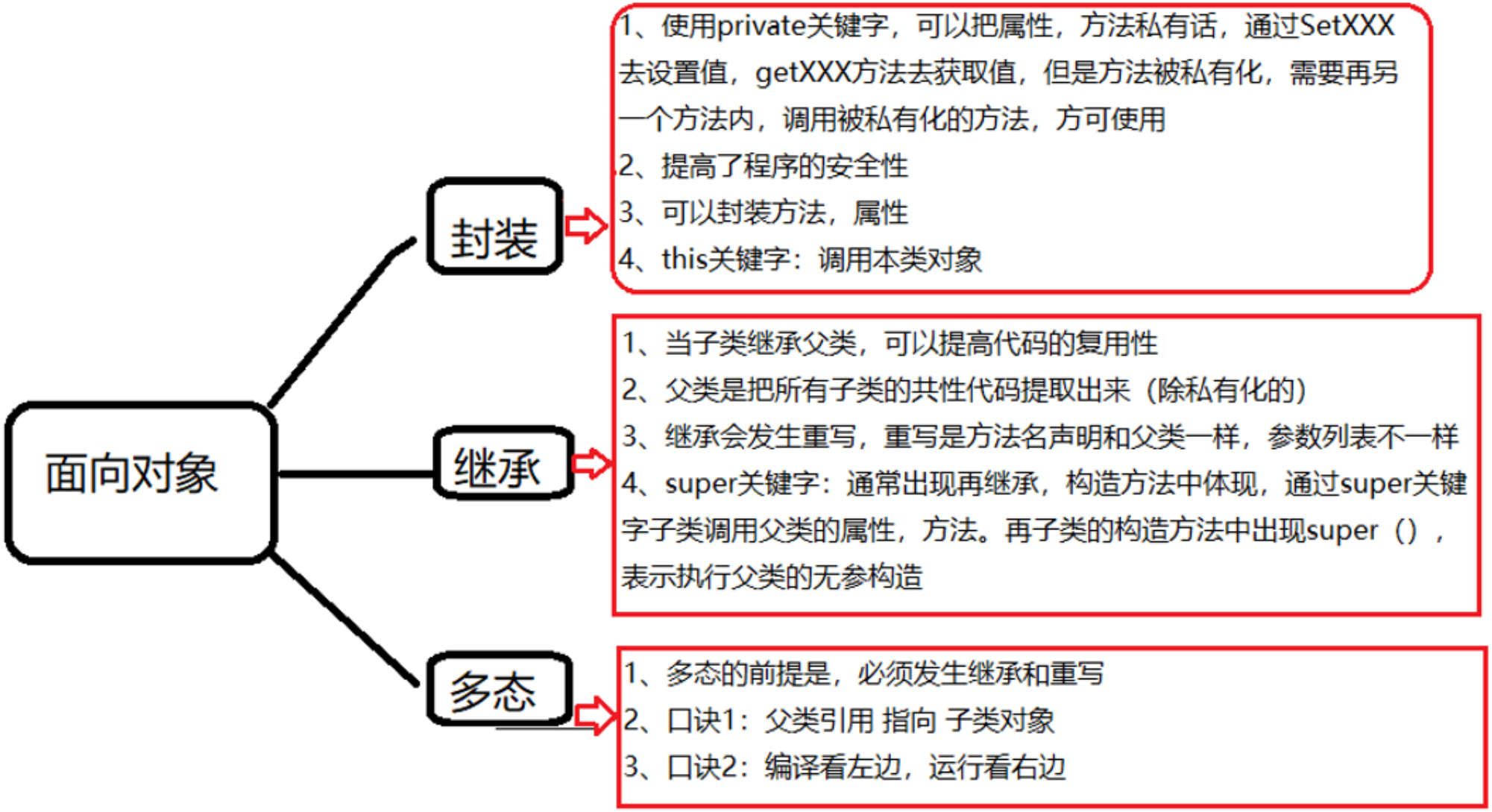
根据以上信息我们可以画出object model.



根据游戏规则以及流程，我们可以画出类图。



面向对象程序设计


面向对象
封装
1、使用private关键字，可以把属性，方法私有话，通过SetXXX去设置值，getXXX方法去获取值，但是方法被私有化，需要再另一个方法内，调用被私有化的方法，方可使用
2、提高了程序的安全性
3、可以封装方法，属性
4、this关键字：调用本类对象
继承
1、当子类继承父类，可以提高代码的复用性
2、父类是把所有子类的共性代码提取出来（除私有化的）
3、继承会发生重写，重写是方法名声明和父类一样，参数列表不一样
4、super关键字：通常出现再继承，构造方法中体现，通过super关键字子类调用父类的属性，方法。再子类的构造方法中出现super（），表示执行父类的无参构造
多态
1、多态的前提是，必须发生继承和重写
2、口诀1：父类引用 指向 子类对象
3、口诀2：编译看左边，运行看右边

面向对象设计三大特征

- **encapsulation** 封装：Combining a number of items, such as variables and functions, into a single package, such as an object of some class, is called encapsulation.将许多项（如变量和函数）组合到一个单独的包中，如某个类的对象，称为封装。

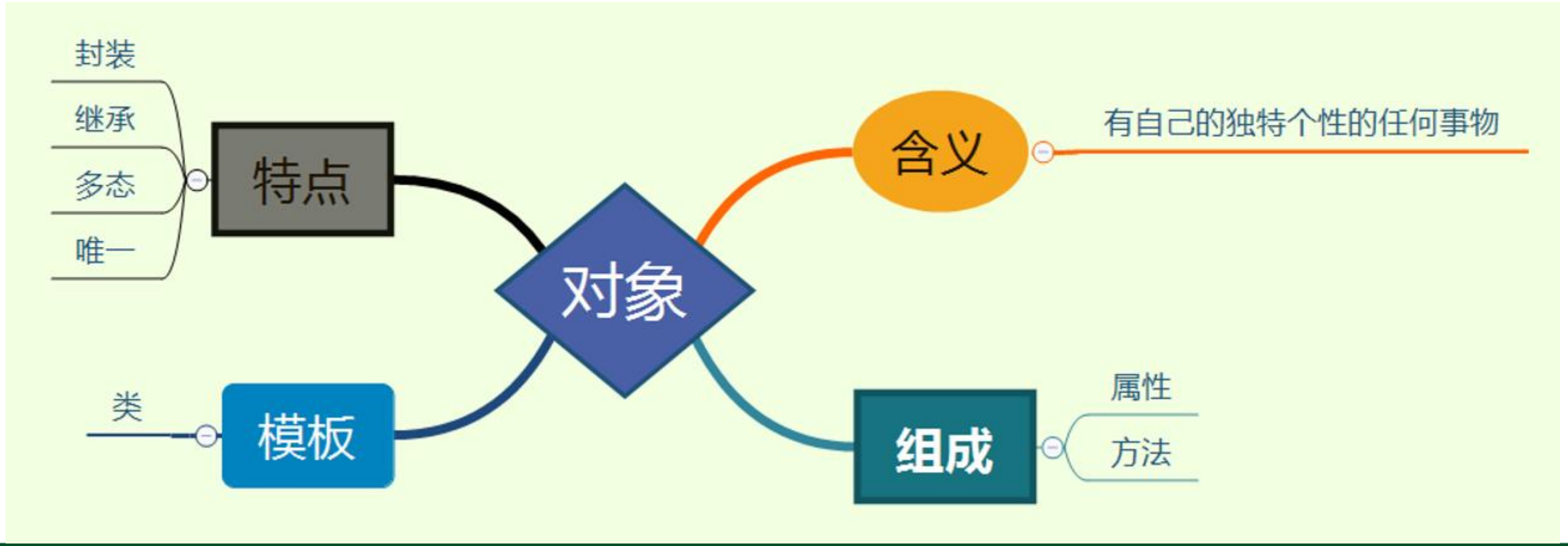
类和对象

技术简介

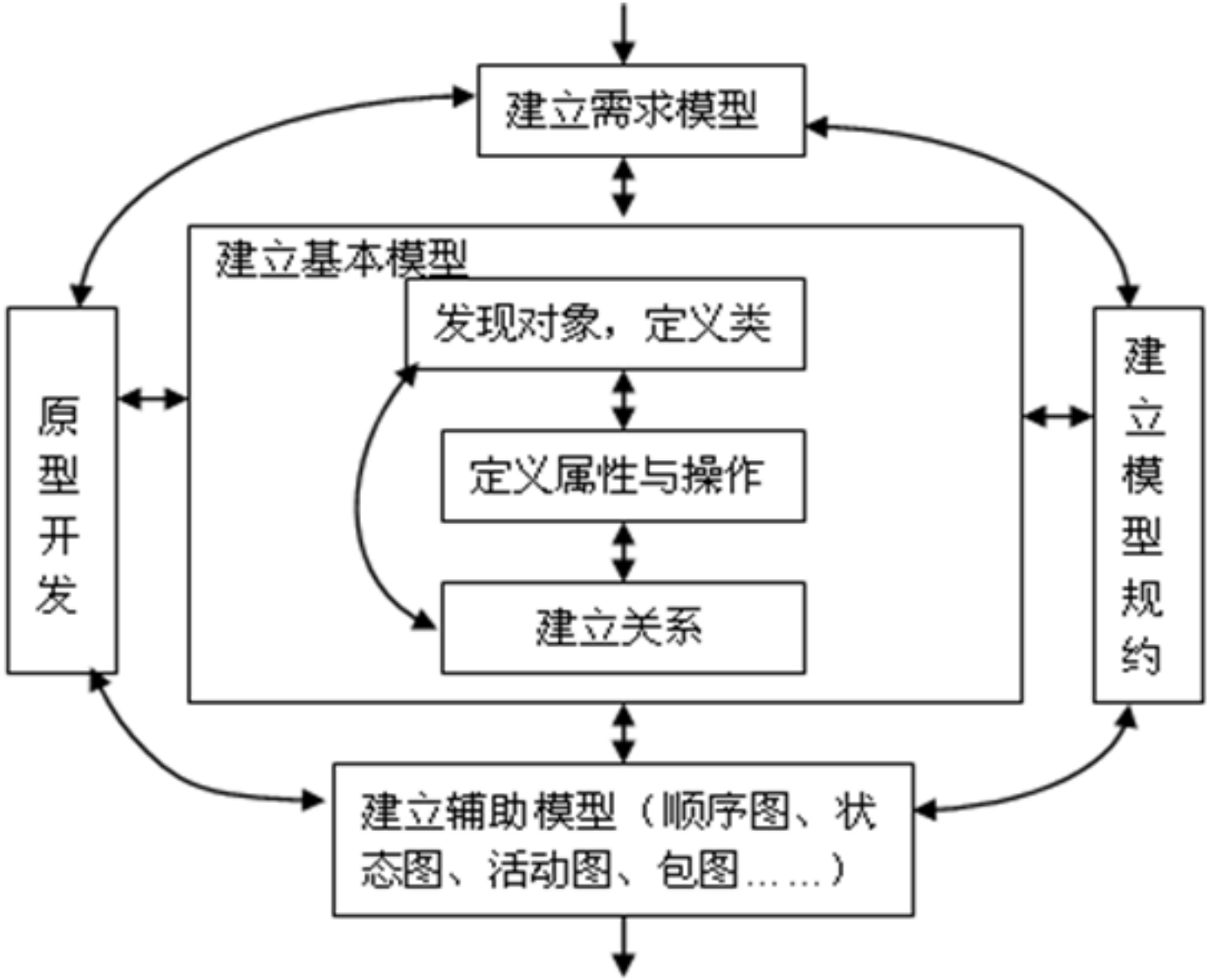
- 类(Class): 用来描述具有相同的属性和方法的对象的集合。它定义了该集合中每个对象所共有的属性和方法。对象是类的实例。
- 方法：类中定义的函数。
- 类变量：类变量在整个实例化的对象中是公用的。类变量定义在类中且在函数体之外。类变量通常不作为实例变量使用。
- 数据成员：类变量或者实例变量用于处理类及其实例对象的相关的数据。
- 方法重写：如果从父类继承的方法不能满足子类的需求，可以对其进行改写，这个过程叫方法的覆盖（override），也称为方法的重写。
- 局部变量：定义在方法中的变量，只作用于当前实例的类。
- 实例变量：在类的声明中，属性是用变量来表示的，这种变量就称为实例变量，实例变量就是一个用 self 修饰的变量。
- 继承：即一个派生类（derived class）继承基类（base class）的字段和方法。继承也允许把一个派生类的对象作为一个基类对象对待。例如，有这样一个设计：一个Dog类型的对象派生自Animal类，这是模拟"是一个（is-a）"关系（例图，Dog是一个Animal）。
- 实例化：创建一个类的实例，类的具体对象。
- 对象：通过类定义的数据结构实例。对象包括两个数据成员（类变量和实例变量）和方法。

类和对象

面向对象是把构成问题事务分解成各个对象，建立对象的目的不是为了完成一个步骤，而是为了描叙某个事物在整个解决问题的步骤中的行为。



面向对象程序设计

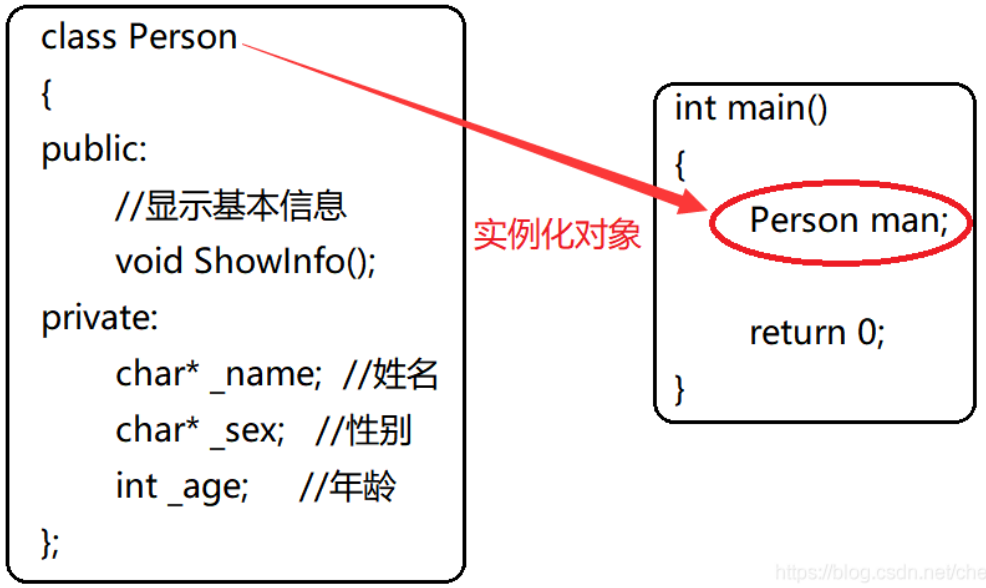

建立需求模型
建立基本模型
发现对象，定义类
定义属性与操作
建立关系
原型开发
建立模型规约
建立辅助模型（顺序图、状态图、活动图、包图……）

类和对象

class	objects
Fruit	Apple Banana Mango

Another example:

class	objects
Car	Volvo Audi Toyota



数据抽象和封装

抽象是通过特定的实例抽取共同特征以后形成概念的过程。一个对象是现实世界中一个实体的抽象，一个类是一组对象的抽象。

封装是将相关的概念组成一个单元，然后通过一个名称来引用它。面向对象封装是将数据和基于数据的操作封装成一个整体对象，对数据的访问或修改只能通过对象对外提供的接口进行。

类的构造函数成员函数

(1) 构造函数

构造函数是一个特殊的、与类同名的成员函数，用于给每个数据成员设置适当的初始值。

(2) 成员函数

成员函数必须在类内部声明，可以在类内部定义，也可以在类外部定义。如果在类内部定义，就默认是内联函数。

（3） 成员函数可被重载

可以有多个重载成员函数，个数不限。

（4）内联函数

有三种：

1）直接在类内部定义。

2）在类内部声明，加上inline关键字，在类外部定义。

3）在类内部声明，在类外部定义，同时加上inline关键字。注意：此种情况下，内联函数的定义通常应该放在类定义的同一头文件中，而不是在源文件中。这是为了保证内联函数的定义在调用该函数的每个源文件中是可见的。

（5） 访问限制

public，private，protected 为属性/方法限制的关键字。

（6） 类的数据成员中不能使用 auto、extern和register等进行修饰, 也不能在定义时进行初始化

如 **int xPos = 0; //错;**