Java

面向对象程序设计

软件学院 贾伟峰

关于本课程

■开设理由

Java本身的流行度(Java只是载体)
大数据技术的要求
面向对象程序设计思想

■基本信息

32学时：24理论学时+8实验学时
1-16周
后续项目课：面向对象编程课程设计

■学习途径

1.面对面建微信群
2.学习通
3.课程门户网址
https://mooc1-1.chaoxing.com/course/204596590.html

课下的学习非常重要！带着疑问上课堂。

1 面向对象的概念

2 类与对象

3 构造方法、this，垃圾回收

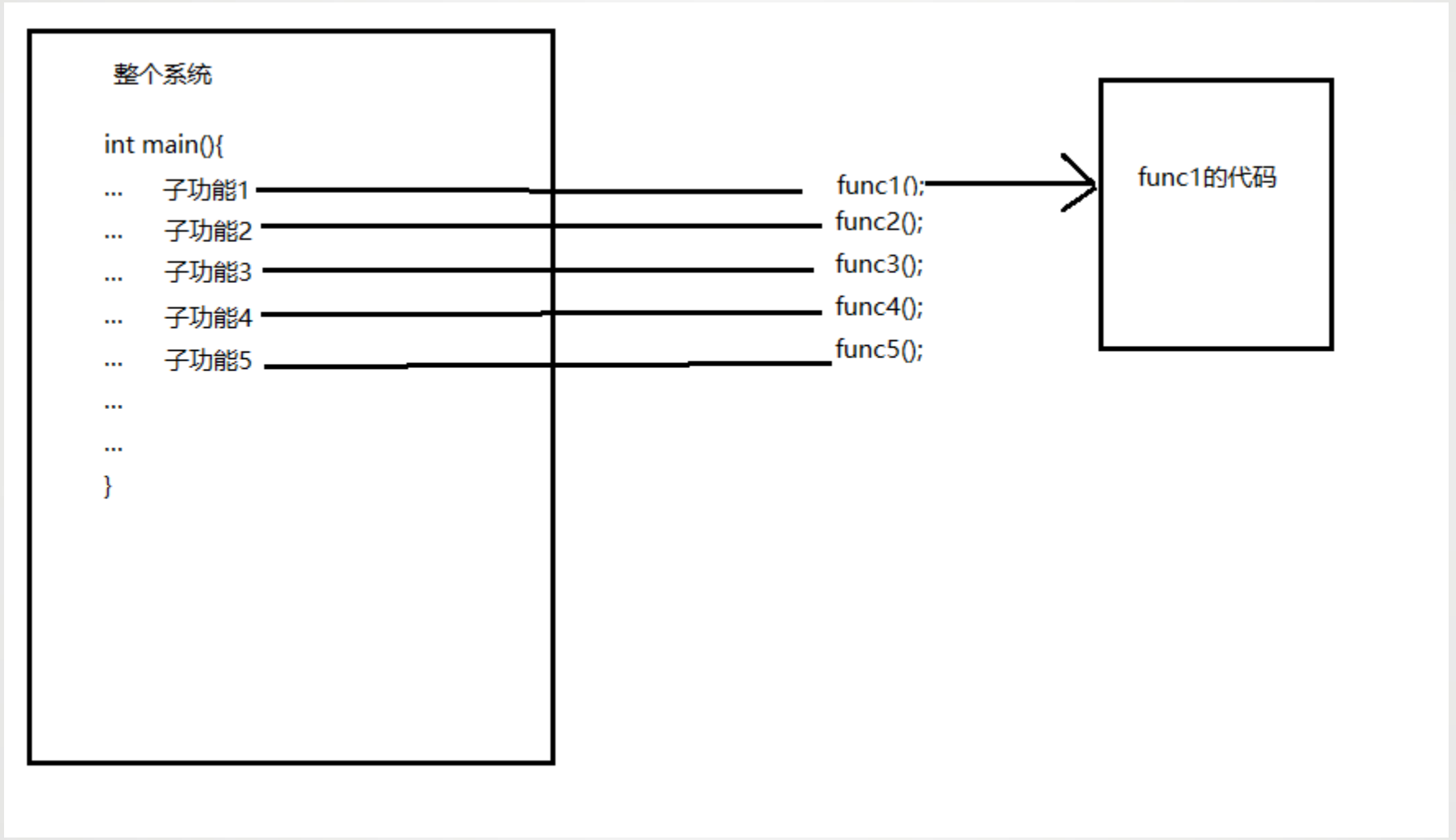
4 static关键词

5 内部类

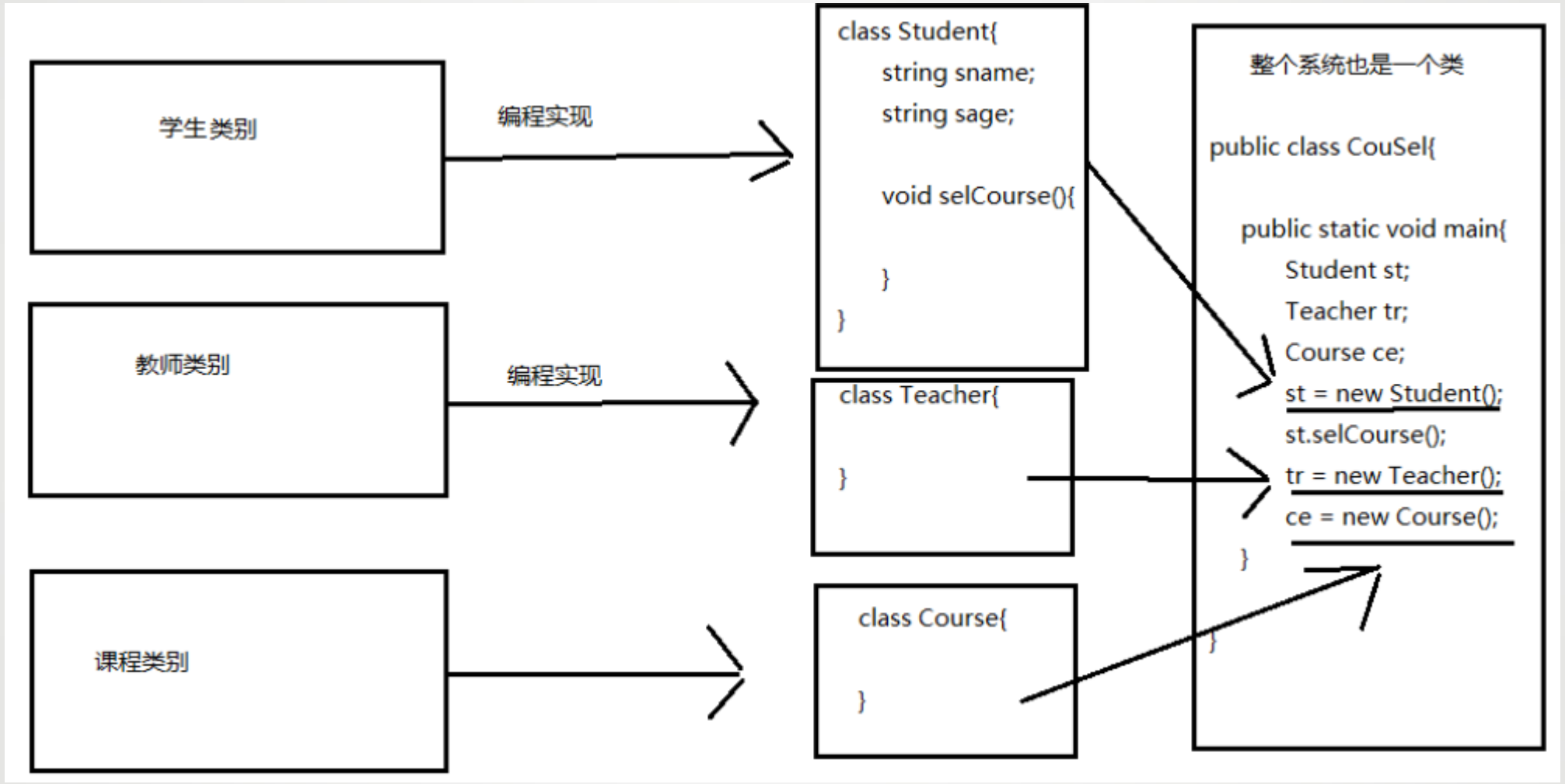
一、面向对象的概念

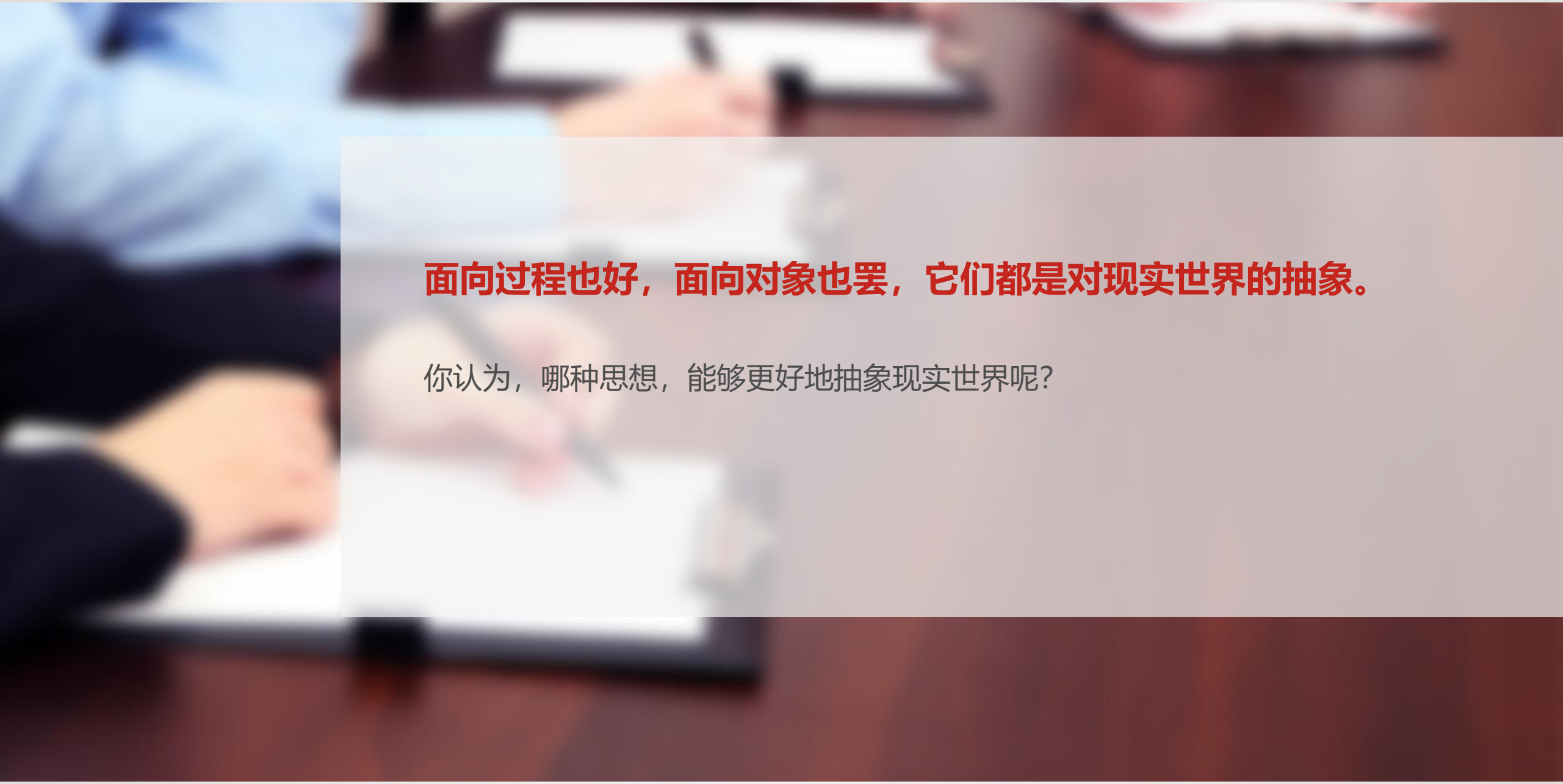
- 编程思想
- C语言与面向过程
- Java与面向对象
- 面向对象重要特征：封装、继承、多态

C语言面向过程编程


整个系统
int main(){
... 子功能1
... 子功能2
... 子功能3
... 子功能4
... 子功能5
...
...
}
func1();
func2();
func3();
func4();
func5();
func1的代码

Java面向对象编程


学生类别
编程实现
教师类别
编程实现
课程类别
class Student{
string sname;
string sage;
void selCourse(){
}
}
class Teacher{
}
class Course{
}
整个系统也是一个类
public class CouSel{
public static void main{
Student st;
Teacher tr;
Course ce;
st = new Student();
st.selCourse();
tr = new Teacher();
ce = new Course();
}
}



面向过程也好，面向对象也罢，它们都是对现实世界的抽象。

你认为，哪种思想，能够更好地抽象现实世界呢？

C与Java示例：设计一个计算器

```c
#include <stdio.h>

void add(int x, int y);
void minus(int x, int y);
void multiply(int x, int y);
void divide(int x, int y);

int main(void){
    add(10, 5);
    minus(10, 5);
    multiply(10, 5);
    divide(10, 5);
    return 0;
}
void add(int x, int y){
    printf("x + y = %d\n", x + y);
}
void minus(int x, int y){
    printf("x - y = %d\n", x - y);
}
void multiply(int x, int y){
    printf("x * y = %d\n", x * y);
}
void divide(int x, int y){
    printf("x / y = %d\n", x / y);
}
```

VS

```java
class Calculator{ /*Calculator类*/
    /*属性*/
    static String version = "1.0";

    //方法，是不是很像c语言中的函数？面向对象中，我们不再提函
    void add(int x, int y){
        System.out.println("x - y = " + (x + y));
    }
    void minus(int x, int y){
        System.out.println("x - y = " + (x - y));
    }
    void multiply(int x, int y){
        System.out.println("x * y = " + (x * y));
    }
    void divide(int x, int y){
        System.out.println("x / y = " + (x / y));
    }
}

public class OOPExample{//程序入口类，又称测试类，顾名思义，
    public static void main(String[] args){//不管程序写得多么

        //类实例化为对象c。面向对象编程，我们现在面向了一个对
        Calculator c = new Calculator();

        //通过对象c，调用不同的方法。
        c.add(10, 5);
        c.minus(10, 5);
        c.multiply(10,5);
        c.divide(10,5);
    }
}
```

程序世界的“世界观”

过程论：数据和逻辑是分离的、独立的，各自形成程序世界的一个方面（Aspect）。所谓世界的演变，是在逻辑作用下，数据做改变的一个过程。……

对象论：数据和逻辑不是分离的，而是相互依存的。相关的数据和逻辑形成个体，这些个体叫做对象（Object），世界就是由一个个对象组成的。……

前者衍生出“**面向过程**”的方法，后者衍生出“**面向对象**”的方法。

https://www.cnblogs.com/feng9exe/p/6782945.html

孰优孰劣？

实践是检验真理的唯一标准。

面向计算

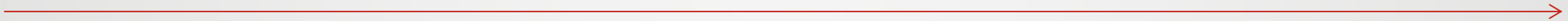
单纯科学计算

面向过程

业务逻辑计算

面向对象

复杂现实世界



编程思想发展路线

二、类与对象

- 类的定义
- 对象的创建和使用
- 类的设计
- 类的封装

类的定义

什么是“类”？

对现实是世界中的同一类“对象”的抽象；描述了他们共同特征和行为。

01

02

对象是“具体的”

现实世界中具体的事物，客观存在的个体。

类是“抽象”的

面向“对象”，抽象出它们的共同特征和行为。以类的形式表达出来。

03

04

如何定义“类”？

```
class Person{
    //特征描述;
    int age;
    //行为描述;
    void speak(){
        ...;
    }
}
```

示例代码：关于类的定义和使用

```java
class Calculator{ /*Calculator类*/
    /*属性*/
    static String version = "1.0";

    //方法，是不是很像c语言中的函数？面向对象中，我们不再提函数，而是说方法。大家可以看到，方法是写在类（class）中的。
    void add(int x, int y){
        System.out.println("x - y = " + (x + y));
    }
    void minus(int x, int y){
        System.out.println("x - y = " + (x - y));
    }
    void multiply(int x, int y){
        System.out.println("x * y = " + (x * y));
    }
    void divide(int x, int y){
        System.out.println("x / y = " + (x / y));
    }
}

public class OOPExample{//程序入口类，又称测试类，顾名思义，是为了帮助测试我们写的Calculator类的类。很拗口吧。
    public static void main(String[] args){//不管程序写得多么复杂，入口都在main这个地方。C语言也是从main开始执行的！

        //类实例化为对象c。面向对象编程，我们现在面向了一个对象，对象名叫c  ^_^
        Calculator c = new Calculator();

        //通过对象c，调用不同的方法。
        c.add(10, 5);
        c.minus(10, 5);
        c.multiply(10,5);
        c.divide(10,5);
    }
}
```

对象的创建和使用

01 让“类”工作

“类”中的普通代码（静态的除外）是无法直接工作的。要让他们工作，得将它实例化为一个“对象”，让“对象”去工作。

02 实例化一个“类”

```
Person p = new Person();
p.speak();
```

Person是编写好的类；
p是变量，通过它，我们可以访问 new Person()生成的对象。

03 还记得malloc么?

C语言中的malloc用于申请内存，使用完之后，还需要free掉这些内存，否则会产生“垃圾”。

04 “垃圾”的产生

new关键词和malloc类似，也会产生“垃圾”：

```
Person p = new Person();
P=null;
```

示例代码：对象的创建和使用

```
1  class Calculator{ /*Calculator类*/
2      /*属性*/
3      static String version = "1.0";
4
5      //方法，是不是很像c语言中的函数？面向对象中，我们不再提函数，而是说方法。大家可以看到，方法是写在类（class）中的。
6      void add(int x, int y){
7          System.out.println("x - y = " + (x + y));
8      }
9      void minus(int x, int y){
10         System.out.println("x - y = " + (x - y));
11     }
12     void multiply(int x, int y){
13         System.out.println("x * y = " + (x * y));
14     }
15     void divide(int x, int y){
16         System.out.println("x / y = " + (x / y));
17     }
18 }
19
20 public class OOPExample{//程序入口类，又称测试类，顾名思义，是为了帮助测试我们写的Calculator类的类。很拗口吧。
21     public static void main(String[] args){//不管程序写得多么复杂，入口都在main这个地方。c语言也是从main开始执行的！
22
23         //类实例化为对象c。面向对象编程，我们现在面向了一个对象，对象名叫c  ^_^
24         Calculator c = new Calculator();
25
26         //通过对象c，调用不同的方法。
27         c.add(10, 5);
28         c.minus(10, 5);
29         c.multiply(10,5);
30         c.divide(10,5);
31     }
32 }
```

没有对象，一切都显得那么静止。我们写的这些非静态方法（无static修饰），均无法运行。

类的设计

■类名

明确是对哪些“对象”进行抽象而得到的类。

■特征

属性的设计

■行为

方法的设计

■关键词的选择

public, protect, private, static, void…

类的封装

封装：面向对象重要特性之一。

隐藏一些直接访问会带来问题的信息，比如一些私有变量。

为什么要封装?
为什么要强调封装?

安全、独立性、解耦

setAge();
getAge();

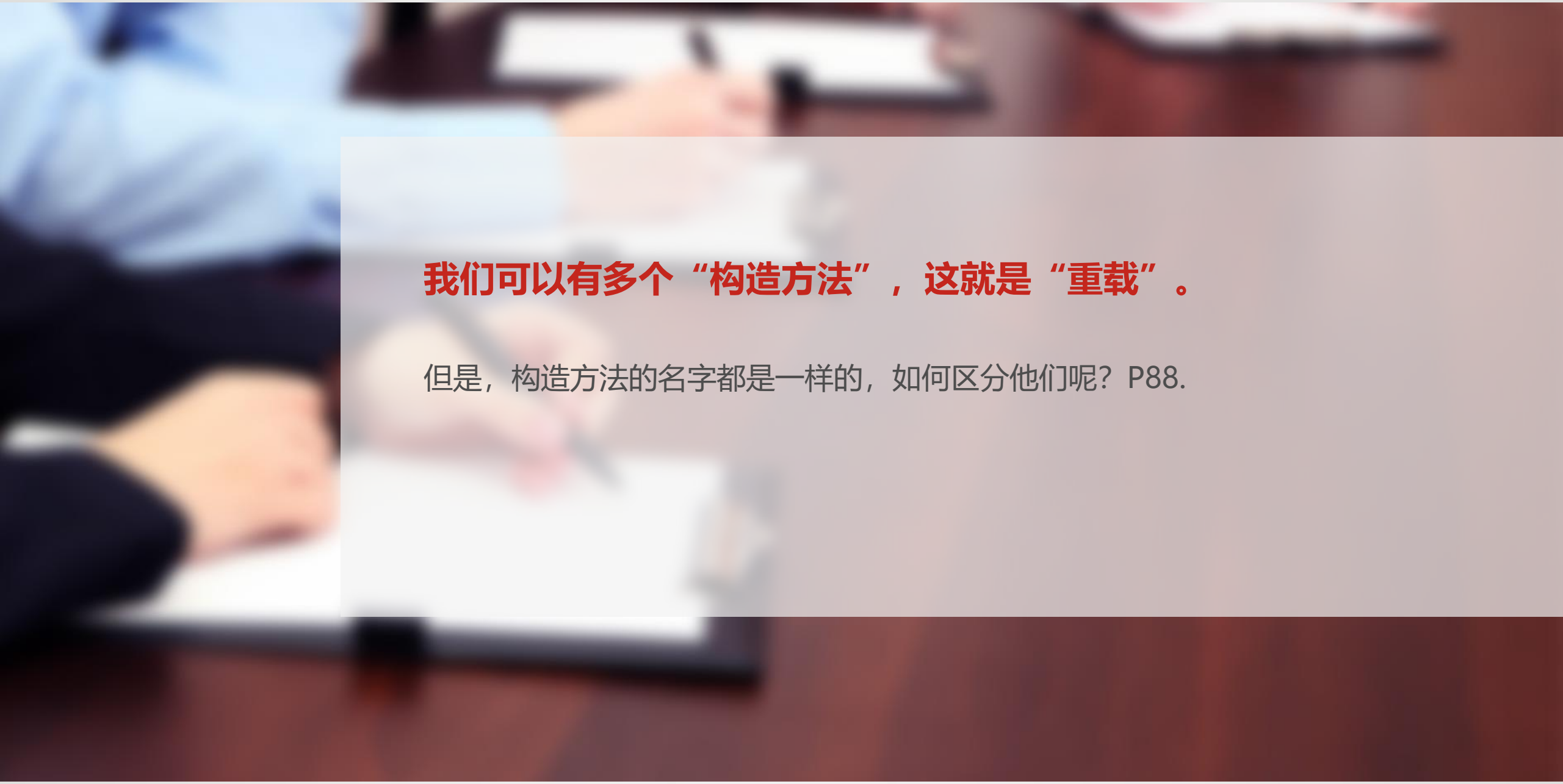
P85例3-6

三、构造方法、this

- 定义
- 重载
- this

构造方法


方法
构造
方法
Person p = new Person();
这里的Person()即构造方法。
疑问：怎么没有在Person类定义中，看到Person()的影子?
思考：如何对默认的构造方法进行重写？以便按照程序员的思路进行构造？比如实例化Person的时候，设置其age的值。请看p86, p87.
按要求修改后，可以这样使用Person:
Person p = new Person(20);



我们可以有多个“构造方法”，这就是“重载”。

但是，构造方法的名字都是一样的，如何区分他们呢？P88.

this关键词

访问成员变量

```
1  class Person{
2      int age;
3      public Person(int age){
4          this.age = age;
5      }
6      public int getAge(){
7          return this.age;
8      }
9  }
```

访问成员方法

```
1  class Person{
2      int age;
3      public void openMouth(){
4          //...
5      }
6      public void speak(){
7          this.openMouth();
8      }
9  }
```

在构造方法中访问其他构造方法：this();

垃圾回收(p94)

new的后果

对象可能成为垃圾，必须想办法处理。

finalize()

同默认的构造方法一样，默认的finalize()方法在垃圾回收时调用。

该过程**自动进行**。

System.gc()

等不及系统的自动化回收，可以通过System.gc()直接启动垃圾回收。

小结


第一次课的内容
C语言
面向过程
Java语言
面向对象
对象无处不在
面向对象
类
class
对象
new
抽象、具体
类与对象
构造方法
new之后
默认就有
this
无参、有参
垃圾回收
new之后，可能会有垃圾
自动回收
强制回收
System.gc()
finalize()

课下任务

一、组建作业小组

7人一组。小组成员要通力合作，保证每位同学提交作业，以免影响小组成绩。下次上课前每小组推荐1名代表，参与最佳作业的评选。

二、作业

1.参见学习通相关视频，安装配置Java环境，编写第一个Java程序；

2.参见学习通相关视频，结合教材内容编写Java代码，并使用github或者gitee托管；

3.撰写本次课程学习心得，写在word文档中，内容可以但不限于以上工作的小组协作情况、代码展示、步骤总结、体会、发现的问题及解决办法等。严禁抄袭！