



# 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  
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：Java面向对象课程设计

### ■学习要求

微信群  
蓝墨云班课  
课下的学习非常重要！带着疑问上课堂。



Contents  
第三章

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

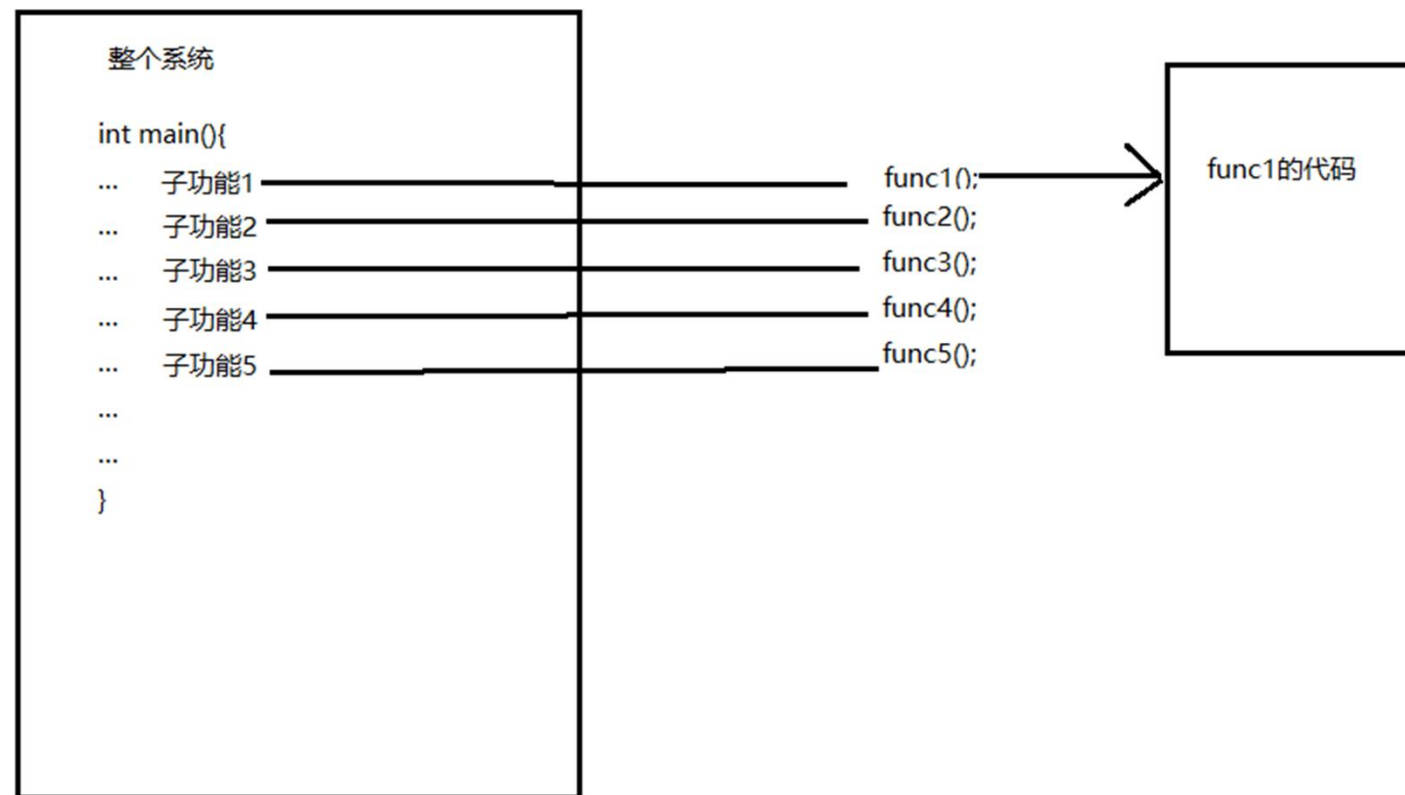




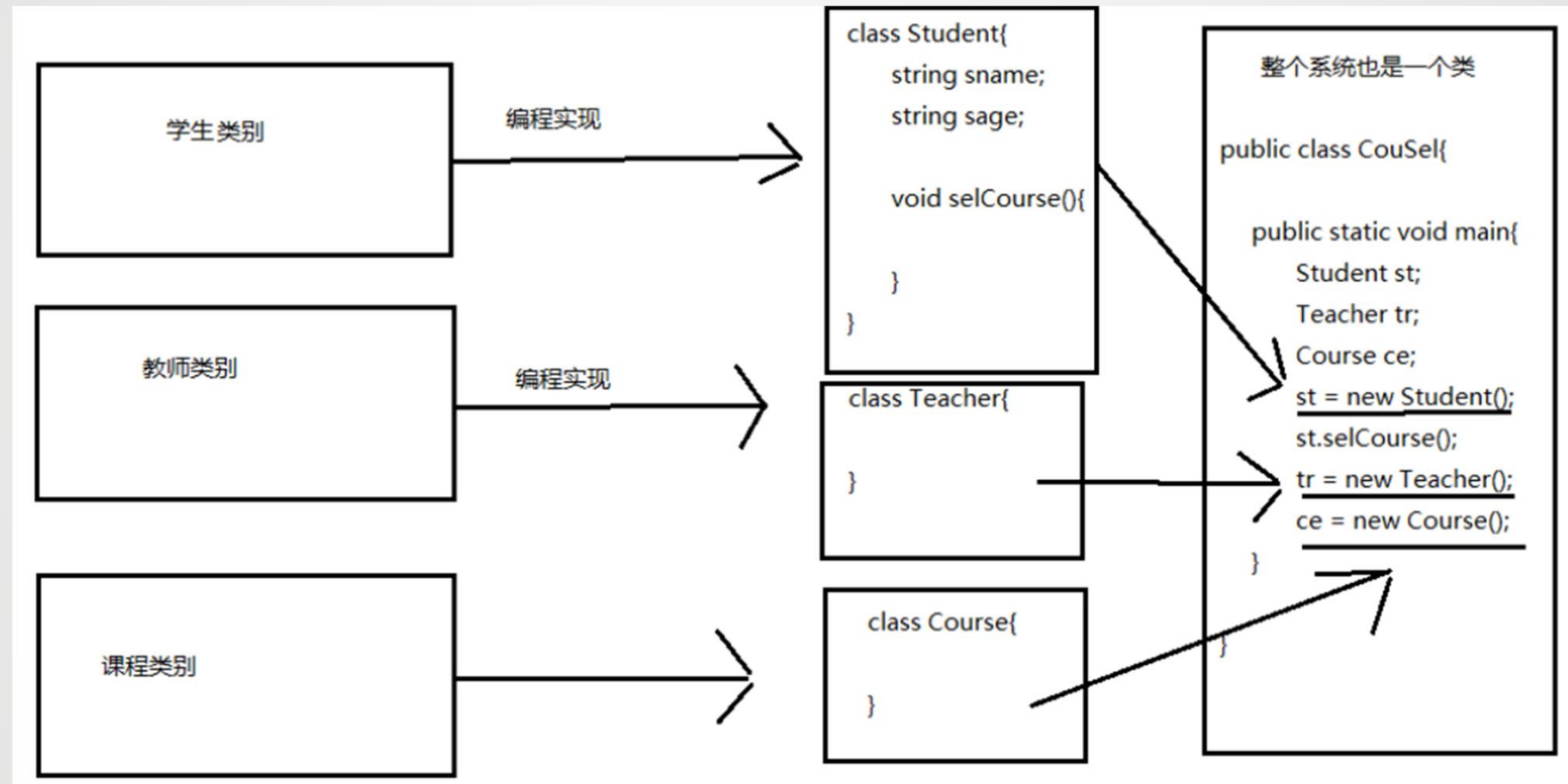
## 一、面向对象的概念

- ④ 编程思想
- ④ C语言与面向过程
- ④ Java与面向对象
- ④ 面向对象重要特征：封装、继承、多态

## C语言面向过程编程



# Java面向对象编程



A blurred background image showing several people sitting at a table, writing on papers with pens. The image is out of focus, emphasizing the text overlay.

**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

```
1 #include <stdio.h>
2
3 void add(int x, int y);
4 void minus(int x, int y);
5 void multiply(int x, int y);
6 void divide(int x, int y);
7
8 int main(void){
9     add(10, 5);
10    minus(10, 5);
11    multiply(10, 5);
12    divide(10, 5);
13    return 0;
14 }
15 void add(int x, int y){
16     printf("x + y = %d\n", x + y);
17 }
18 void minus(int x, int y){
19     printf("x - y = %d\n", x - y);
20 }
21 void multiply(int x, int y){
22     printf("x * y = %d\n", x * y);
23 }
24 void divide(int x, int y){
25     printf("x / y = %d\n", x / y);
26 }
27
```

VS

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





## 二、类与对象

④ 类的定义

④ 类的设计

④ 对象的创建和使用

④ 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

```
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 }
```

### 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。

01

### 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”。

03

### 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。
```

02

### “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;
```

04

## 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

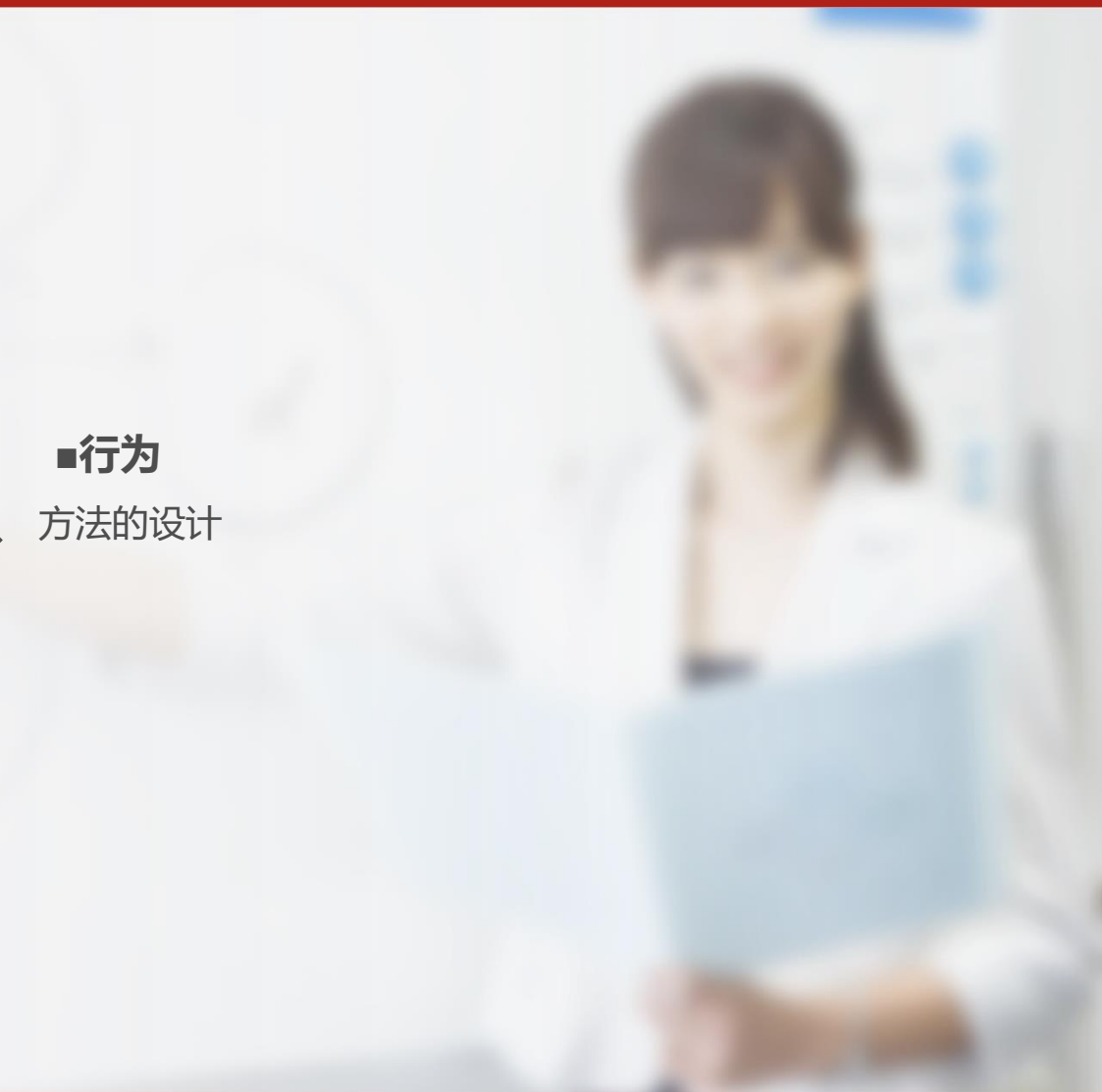
### ■行为

方法的设计

类的设计

### ■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。

What



Why



为什么要封装？  
为什么要强调封装？

安全、独立性、解耦

```
setAge();  
getAge();
```

How





### 三、构造方法、this

▶ 定义

▶ 重载

▶ this



## 构造方法





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();

### 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

