

**项目计划书修订版（C)**

The project plan is the most important indicator plan. The brief introduction reflects the project achievement status of the company. The project plan needs to be formulated in strict accordance with the process

**基础知识**

**Java**

**PROJECT PLANNING PAPER**

**COMPANY NAME**

**JAVA 基础知识**

目录

[第一章:JAVA语言概述 7](#_Toc32608)

[一．Java概述 7](#_Toc8965)

[二、 了解Java集成开发环境 7](#_Toc2396)

[三、 使用MyEclipse开发Java 9](#_Toc7503)

[java的执行机制:先编译，后解释 9](#_Toc24017)

[1、Java开发工具 9](#_Toc15501)

[2、什么是java虚拟机？ 9](#_Toc2303)

[3、JDK(软件开发包) 10](#_Toc19817)

[4、配置JAVA环境变量 10](#_Toc6105)

[四、 Java程序结构组成 11](#_Toc29357)

[1、类的描述 11](#_Toc28724)

[2、包 11](#_Toc22598)

[3、注释 12](#_Toc7691)

[第二章 变量和数据类型 13](#_Toc13053)

[一、 JAVA中的常量和变量 13](#_Toc24788)

[1、 常量 13](#_Toc29434)

[2、 变量 13](#_Toc1145)

[二、 JAVA中的标识符 14](#_Toc27425)

[1、概念 14](#_Toc17224)

[2、命名规则： 15](#_Toc5675)

[3、良好的命名习 15](#_Toc14769)

[4、口诀：字母数字下划线$,开头不能是数字 15](#_Toc12289)

[5、Java标识符命名法 15](#_Toc18965)

[三、 JAVA中的关键字 16](#_Toc26048)

[四、 JAVA中的数据类型 16](#_Toc29244)

[1、Java数据类型 16](#_Toc5818)

[2、6种可用于计算的数据类型 16](#_Toc21019)

[五、 JAVA中的数据类型转换 19](#_Toc20824)

[第三章 运算符和表达式 22](#_Toc4090)

[一、 理解表达式 22](#_Toc14050)

[二、 掌握赋值运算符 22](#_Toc32178)

[1、运算符 22](#_Toc471)

[2、赋值运算符与赋值表达式 23](#_Toc2586)

[三、 掌握算术运算符 23](#_Toc6124)

[1、算术运算符与算术表达式 23](#_Toc3218)

[2、复合赋值运算符 24](#_Toc24187)

[四、 掌握关系运算符 24](#_Toc31395)

[1、关系运算符与关系表达式 25](#_Toc19504)

[2、“==”与“=”的区别： 25](#_Toc4710)

[五、掌握逻辑运算符 25](#_Toc18528)

[1、逻辑运算符 25](#_Toc8466)

[2、条件运算符 26](#_Toc10699)

[3、 运算符的优先级 27](#_Toc3880)

[第四章 分支结构 28](#_Toc30886)

[一、键盘录入(Scanner)（重点） 28](#_Toc26328)

[二、if语句(选择结构)（掌握） 28](#_Toc30193)

[1.基本if语句 28](#_Toc28294)

[2.if...else （如果......否则......） 28](#_Toc14935)

[3.多重if 29](#_Toc165)

[4.if嵌套 30](#_Toc4393)

[5.switch(分支结构)(掌握) 31](#_Toc21313)

[6.if和switch区别? 32](#_Toc6207)

[三、块作用域 32](#_Toc13542)

[第五章 循环结构 34](#_Toc13242)

[第六章 数组 37](#_Toc12755)

[一．了解数组 37](#_Toc7422)

[二．数组的初始化 38](#_Toc25412)

[三、数组的遍历（掌握） 39](#_Toc29086)

[四、数组默认值 40](#_Toc9435)

[五、数组的创建 40](#_Toc26292)

[六、数组扩容（重点） 41](#_Toc27435)

[七、数组地址的替换(掌握) 41](#_Toc10672)

[八、 数组作为形式参数和返回值类型(掌握) 41](#_Toc19603)

[九、 数组排序 41](#_Toc16907)

[1、 可变长参数(背) 41](#_Toc6894)

[2、 冒泡排序(掌握) 42](#_Toc10576)

[3、 选择排序(掌握) 42](#_Toc9274)

[4、 快速排序(掌握) 43](#_Toc11707)

[十、二维数组的定义与使用 43](#_Toc7466)

[1、定义 43](#_Toc31477)

[2、二维数组的创建(了解) 45](#_Toc11650)

[十一．数组总结 46](#_Toc5176)

[第七章 Java中的方法 47](#_Toc31459)

[一、 JAVA的方法 47](#_Toc18856)

[1.方法 47](#_Toc30583)

[2.方法的定义 47](#_Toc30816)

[二、 方法的参数 48](#_Toc10570)

[1、带参数的方法 48](#_Toc11411)

[2、带多个参数的方法 48](#_Toc9801)

[3、参数的顺序 49](#_Toc16231)

[4、方法的返回值 49](#_Toc28282)

[5、形参与实参 50](#_Toc2671)

[6、以数组作为参数的方法 52](#_Toc12724)

[三、变量的作用域 52](#_Toc9827)

[四、可变的参数的方法 53](#_Toc13584)

[五、 类之间方法的相互调用 54](#_Toc32335)

[六．总结 56](#_Toc20586)

[第八章 函数 57](#_Toc26494)

[一、认识函数 57](#_Toc30020)

[1.概念:实现特定功能的一段代码，反复使用。 57](#_Toc8544)

[2.语法:(掌握) 57](#_Toc29043)

[3.函数的调用: 函数名称(); 57](#_Toc13886)

[4.注意: 57](#_Toc21641)

[5、带有参数的函数语法: 57](#_Toc11232)

[二、返回值与返回值类型 58](#_Toc14434)

[三、return的用法 58](#_Toc8121)

[四、函数总结 58](#_Toc12346)

[五、递归 59](#_Toc13529)

**第一章:JAVA语言概述**

**一．Java概述**

1.概念：一种程序设计语言，一个开发环境，一个应用程序的运行环境，一个程序发布的环境，Java是面向对象的编程语言。

2.java历史：

由于这些智能化家电的市场需求没有预期的高，Sun放弃了该项计划。

就在Oak几近失敗之时，随着互聯網的发展，Sun看到了Oak在计算机网络上的广阔应用前景，于是改造了Oak，在1995年5月以“Java”的名称正式发布了。

（java语言属于sun公司,后来被Oracle公司收购。）

Java伴随着互联网的迅猛发展而发展，逐渐成为重要的网络编程语言。

1. **了解Java集成开发环境**
2. **JAVA的特点：**

①跨平台(跨操作系统)。

②简单(语法结构比C、c++简单)。

③纯面向对象语言。

**没有复杂的指针，自动 释放内存**

**一种面向对象的程序设计语言**

**write once, run anywhere**

**提供动态下载程序代码的机制，适合在网络上运行**

**提供代码检验机制以保证安全性**

2.C,C++的特点：

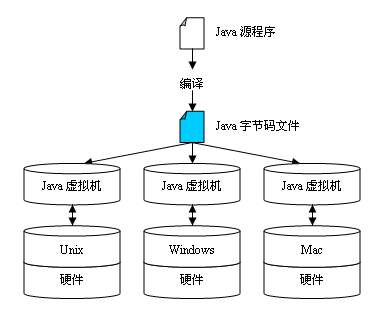
**语法比较复杂**

**主要面向过程的程序设计语言**

**多半是单击版不适合在网络上运行**

**有指针，需要手动释放内存**

3.java流程：



运行方便、安全

一次性编写，到处运行！

1. **使用MyEclipse开发Java**

**java的执行机制:先编译，后解释**

**.java（源文件） --- 编译 --- .class（字节码文件） ---解释器 -- JVM --- os**

**1、Java开发工具**

（1）Eclipse： 是一个开源的、基于Java的可扩展开发平台。Eclipse是一个优秀的集成开发环境，可以用它简化Java程序开发过程。优点：比较干净，不带任何插件

（2）MyEclipse：是在eclipse 基础上加上自己的插件开发而成的功能强大的企业级集成开发环境，主要用于Java、Java EE以及移动应用的开发。**优点：**做web项目比较好，因为集成了很多的网页插件。



1. Jbuilder：大型的软件，写起来比较麻烦，收费软件。

2、什么是java虚拟机？

（1）Java虚拟机（Java Virtual Machine) 简称JVM， Java虚拟机是一个想象中的机器，在实际的计算机上通过软件模拟来实现。

（2）一般的高级语言如果要在不同的平台上运行，至少需要编译成不同的目标代码。而引入Java语言虚拟机后，Java语言在不同平台上运行时不需要重新编译。Java语言使用模式Java虚拟机屏蔽了与具体平台相关的信息，使得Java语言编译程序只需生成在Java虚拟机上运行的目标代码**（字节码）**，就可以在多种平台上不加修改地运行。Java虚拟机在执行字节码时，把字节码解释成具体平台上的机器指令执行。

3、JDK(软件开发包)

（1）适于Windows98和WindowsNT/2000,Linux,Unix等OS**，Java的JDK版本是免费的，可以从java.sun.com免费下载。**

（2）JDK是Java语言最基本的开发工具包，包括：

①javac（Java编译器）；

②java（Java解释器）；

③jdb（Java调试器）；

④javah（C语言头文件和存根文件产生器）；

⑤javap（Java反汇编器）；

⑥javadoc（Java文档管理器）；

⑦appletviewer（Java Applet浏览器） 。

1. JRE:(包含解释器+JVM)运行环境

JDK:(包含JRE+类库+编译器+调试工具)开发工具

4、配置JAVA环境变量

（1）右键计算机-属性-高级系统设置-环境变量

（2）**path:JDK的安装目录/bin**

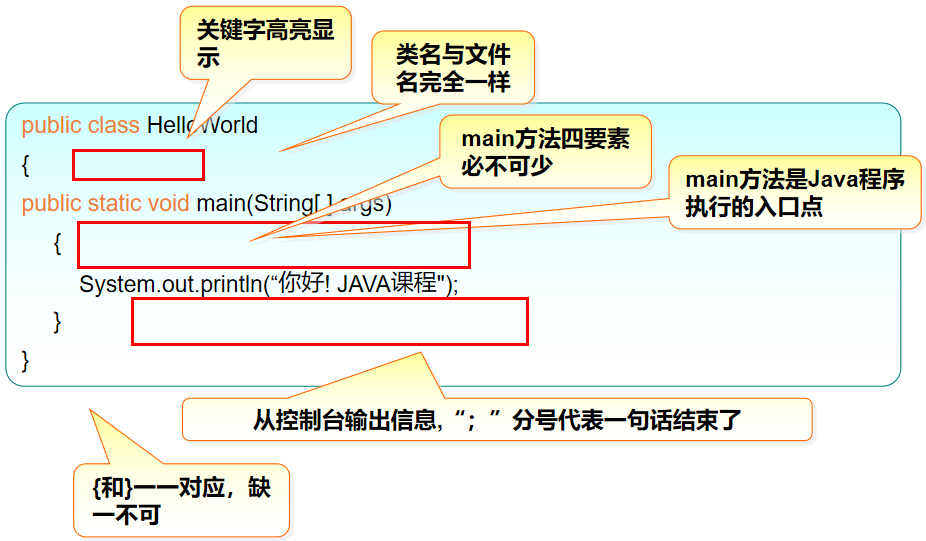
**JAVA\_HOME:JDK的安装目录**

**CLASSPATH:.**

（3）验证JDK是否配置成功

win+R-输入cmd- 输入java -version或输入java和javac命令，只要没有出现，java或javac不是内部或外部命令，说明安装和配置都正确

1. **Java程序结构组成**



1、类的描述

（1）.一个源文件中可以有多个类，在编译之后每个类都会生成对应的.class文件。

（2）.一个类中只能有一个主函数。

（3）.一个源文件中只能有一个公开类，文件中如果有公开类，文件的名称必须和类名一致(包括大小写)。

2、包

作用:方便管理.class文件

语法:package 包名;

包的命名采用域名倒置:举例:www.baidu.com.cn -- cn.com.baidu.xxx

**3、注释**

单行注释: //

多行注释: /\*\*/

文档注释: /\*\*\*/

1. **变量和数据类型**
2. **JAVA中的常量和变量**
3. **常量**

（1）什么是常量?

值永恒不变的量叫常量。如圆周率(3.14)、

光速(299792.458km/s)、水的密度是1g/mL等。

（2）Java约定常量的标识符全部使用大写英文字母表示。

声明常量使用关键字final来修饰。

格式：final 类型名 标识符=值;

如：final float PI = 3.14;

常量的类型和取值范围和变量都是一样的。

1. **变量**
2. 什么是变量？

时间， 星期

**概念**:计算机内存中的一块存储空间。

（2）Java变量是程序中最基本的存储单元。

**三要素：变量类型、变量名、作用域**

### **（3）变量的定义流程（重点）**

方式1:先声明，再赋值。

语法:数据类型 变量名; 变量名 = 值;

举例:int money; money = 100;

方式2:声明并赋值

语法:数据类型 变量名 = 值;

举例:int age = 20;

方式3:同时声明多个变量并赋值（了解）

语法:数据类型 变量名1,变量名2 = 值,变量名3 = 值...

举例:int a,b,c = 30;

a = 10;

b = 20;

可以在程序中的任何一个地方做变量的声明，只要在使用之前声明即可。

**注意：**

1.变量必须先赋值再使用。

2.java是强类型语言变量的类型必须和数据的类型保持一致。

1. **JAVA中的标识符**

1、概念

用来标识类名、变量名、数组名、方法名等的有效字符称为标识符。简单的说，标识符就是一个名字。由字母、数字、\_、$组成。但是数字不能开头。不能和java中的关键字或保留字(goto、null、false、true)重名。

2、命名规则：

（1）由合法字符（大小写字母、汉字、希腊字母等）、数字、下划线组成。

（2）由合法字符、下划线（\_）或美元符号($)开头，不能以数字开头。

（3）区分大小写，如name和Name就是两个不同的标识符。

（4）长度无限制。

3、良好的命名习

（1）望文生义，见名知意。

（2）类名如果由多个单词组成，每个单词的首字母大写，其余都小写。

（3）变量/函数名如果由多个单词组成，首单词首字母小写，拼接词首字母大写。

（4）常量如果由多个单词组成，全部大写，单词之间用\_连接。

（5）包名全小写，层级关系之间用"."隔开。

4、口诀：字母数字下划线$,开头不能是数字

5、Java标识符命名法

（1）**匈牙利（Hungarian）命名法：**

标识符的名字以一个或者多个小写字母开头作为前缀；前缀之后的是首字母

例如：int iAge=18;

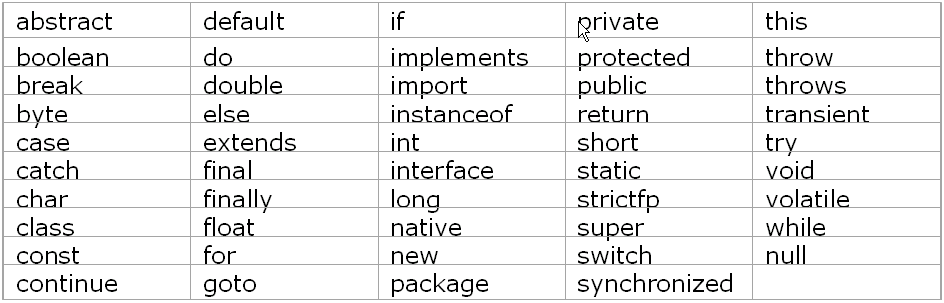
（2）**骆驼（Camel）命名法:**

骆驼式命令法，正如它的名称所表示的那样，是指混合使用大小写字母来构成变量和函数的名字。一般都是首字母小写。

例如： String userName = “李华”;

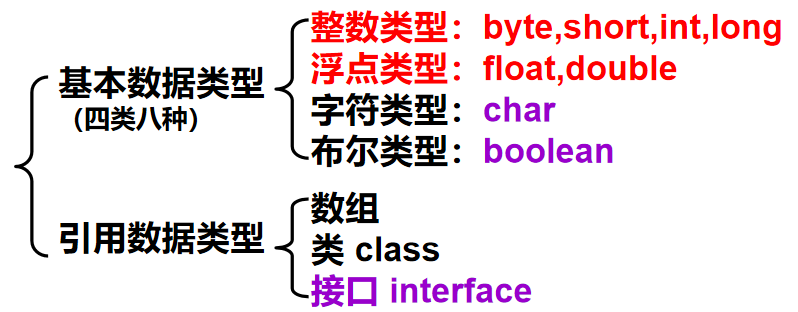
1. **JAVA中的关键字**

**一些赋予特定的含义，用做专门用途的字符串（大多数编辑器都会以特殊颜色标出）**



1. **JAVA中的数据类型**

**1、Java数据类型**



Java各种数据类型有固定的字段长度，其不受具体操作系统的影响，以保证Java程序的可移植性。

**2、6种可用于计算的数据类型**

整数类型：byte（字节）,short（短整型）,int（整型）,long（长整型）

浮点类型：float（单精度浮点型）,double（双精度浮点型）

|  |  |  |
| --- | --- | --- |
| **数据类型** | **存值范围** | **占用存储空间** |
| byte | 127 至 -128 | 1字节 |
| short | 32 767 至 -32 768 | 2字节 |
| int | 2 147 483 647 至 -2 147 483 648 | 4字节 |
| long | 9 223 372 036 854 775 807 至  -9 223 372 036 854 775 808 | 8字节 |
| float | 3.402 823e+38 至 1.401 298e-45 |  |
| double | 1.797 693e+308 至 4.900 000e-324 |  |

数据类型范围大小（从低到高的顺序）数值类型

byte < short < int < long < float < double

**注意:**

1.声明long类型的数据，数值后面加"L"或"l"。

2.声明float类型的数据，数值后面加"F"或"f"。

3.整数字面值默认int类型，小数字面值默认double类型。

4.float和double存储的数值是近似值。

（1）整数类型

所有整数都是有符号数。

整型常量有三种表示形式：十进制、八进制(前面加0)、十六进制（前面加0x或0X）

整型变量默认类型为int类型。

long类型数据的值则需在数值后加’L’或’l’ 。

（2）字符类型char 2个字节

用单引号括起来的单个字符。

字符涵盖所有语言涉及的字符，包括汉字、日文和韩文等。也可用十六进制编码形式表示。

字符分为普通字符和特殊字符(如转义字符\n,\r,\t)

所有字符可以采用统一的书写方式表示:\u+字符编码

如：char eChar = ‘a’; char c1 = ‘\u0062’;

char cChar = ‘中’;

char c2 = ‘\n’; ‘\n’代表换行符

转义字符：（了解）

\n 换行

\t 制表符

\\ \

\" "

\' '

（3）布尔类型boolean　 1个字节

适于逻辑运算，一般用于程序流程控制。

boolean类型数据只允许取值true或false。

**注 意：**

true和false这两个值不能加单引号

只能小写

不可以用0或非0的整数替代。

1. **JAVA中的数据类型转换**

1、数据类型的自行转换和强行转换

**自转：**自动转换由 JAVA编译器自动转换的。只能发生在数据类型的范围由低向高转换。

**强转：**强制类型转换需要人为转换。发生在数据类型的范围由高向低转换。和特殊条件下的转换。

从高到低 强制转换

从低到高 自动转换

数据类型范围大小（从低到高的顺序）数值类型

byte->short->int->long->float->double

2、基本数据类型转换

boolean类型不可以与其他数据类型进行转换。

整型、浮点型、字符型之间是可混合运算的！！！！

3、类型转换

自动类型提升：目标类型大于源类型。(掌握)

举例:byte b = 20;

int i = b;

int i:目标类型 byte b:源类型

​

强制类型转换:目标类型小于源类型。(掌握)

举例:short s = 128;

byte b = (byte)s;//-128

**注 意:**

（1）.如果长度足够数据完整。

（2）.如果长度不够数据截断。

（3）.小数转整数，小数舍掉，保留整数部分。

（4）.char类型转int无需强转。

### **4、自动类型提升**

两个操作数其中一个为double，最终结果为double

两个操作数没有double类型，其中一个为float，最终结果为float

两个操作数没有float类型，其中一个为long，最终结果为long

两个操作数没有long类型，其中一个为int，最终结果为int

两个操作数没有int类型，均为short、byte、char最终结果提升为int

特殊:String和任何类型相加(+)，实际是拼接。

**5、使用Scanner录入数据**

//得到一个从键输入对象

Scanner input =new Scanner(System.in);

//接收一个输入的整数

Int a=input.nextInt();

**第三章 运算符和表达式**

1. **理解表达式**

a+1

a/2+a\*3

(a+1)/(b-2)

a>3

1. **掌握赋值运算符**

1、运算符

表达式由操作数、运算符等元素组成，最常见的运算符及对应表达式有：

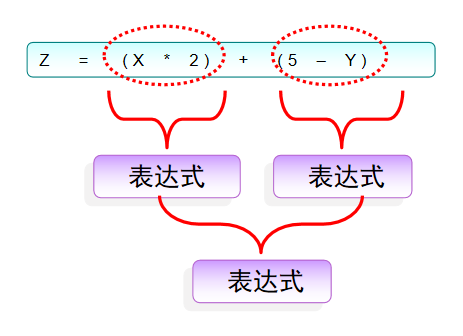
赋值运算符-赋值表达式

算术运算符-算术表达式

关系运算符-关系表达式

逻辑运算符-逻辑表达式

条件运算符-条件表达式



2、赋值运算符与赋值表达式

在所有运算符中，最简单的就是赋值运算符。它的通用形式为：variable = expression;

variable是任何有效的标识符，expression是常量、变量或表达式

例如：int age; age = 19;

注意“=”和数学中的“等于”，在java中“=”是把右边的值赋值给左边

1. **掌握算术运算符**

1、算术运算符与算术表达式

+、-、\*、/、%、++、--

分为**一元运算符和二元运算符**

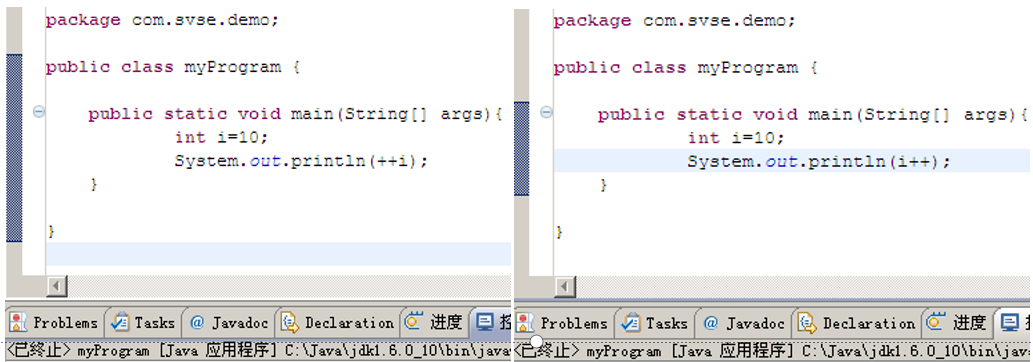
**（1）一元运算符**

①指只处理一个操作数的运算符

②++： num++等同于 num = num + 1

③--： num—等同于 num = num – 1

注意区分 num++ 与 ++num、num--与--num的区别



**（2）二元运算符**

①二元运算符是指处理两个操作数的运算符。

②+、-、\*、/、%

③“%”运算符用来求余数，即两个数相除获得整数商以后的余数，该运算符只作用于两个整数

2、复合赋值运算符

**在赋值运算符“=”之前加上二元算术运算符可构成复合赋值运算符**

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 表达式 | 计算 | 结果（假设a=10） |
| += | a += 5 | a = a + 5 | 15 |
| -= | a -= 5 | a = a - 5 | 5 |
| \*= | a \*= 5 | a = a \* 5 | 50 |
| /= | a /= 5 | a = a / 5 | 2 |
| %= | a %= 5 | a = a % 5 | 0 |

1. **掌握关系运算符**

1、关系运算符与关系表达式

关系运算符就是用于测试两个操作数之间关系的符号，其中操作数可以是变量、常量或表达式，结果返回布尔值（true或false）

使用关系运算符连接的表达式叫做关系表达式

> 、< 、>=、<=、==、!=

2、“==”与“=”的区别：

= 代表赋值 如 int num=10;

== 代表比较连个表达式是否相等 如 int num1=10,num2=20;

**五、掌握逻辑运算符**

1、逻辑运算符

逻辑运算符用于测试两个操作数之间的逻辑关系，且这两个操作数必须是布尔类型的（例如关系表达式），得到的结果也是布尔类型。

通过逻辑运算符连接的结果为boolean型的变量或表达式叫做**逻辑表达式**

|  |  |  |
| --- | --- | --- |
| 逻辑运算符 | 描述 | 表达式 |
| ! （逻辑非） | 将操作数的值改变，真反转为假，假反转为真 | !a |
| &&（短路与） | 只有两个条件都为真才返回真，否则返回假 | a && b |
| ||  （短路或） | 两个条件任意一个为真就返回真，两个均为假则返回假 | a || b |

总结：与运算 有假为假 或运算 有真为真

2、条件运算符

条件运算符又称**三元运算符**，是“?”和“:”符号的组合，根据条件执行两个语句中的其中一个，它的一般形式如下：

**test ? 语句1 : 语句2**

布尔类型表达式为true取结果1，为false取结果2

**int** num = 15;

String str;

str = (num%2==0) ? "num是偶数！" : "num是奇数！";

System.out.println (str);

1. 运算符的优先级

|  |  |  |
| --- | --- | --- |
| 次序 | 运算符 | 结合性 |
| 1. | 括号，如( )和[ ] | 从左到右 |
| 2. | 一元运算符，如+(正)，-(负)，++，--和! | 从右到左 |
| 3. | 乘除算术运算符，如\*，/和% | 从左到右 |
| 4. | 加减算术运算符，如+（加）和-（减） | 从左到右 |
| 5. | 大小关系运算符，如>，<，>=和<= | 从左到右 |
| 6. | 相等关系运算符，如==和! = | 从左到右 |
| 7. | 与运算符 ，如& 和 && | 从左到右 |
| 8. | 异或运算符，如 ^ | 从左到右 |
| 9. | 或运算符 ，如| 和 || | 从左到右 |
| 10. | 条件运算符 如，? : | 从左到右 |
| 11. | 赋值运算符，如=，+=，-=，\*=，/=和%= | 从右到左 |

**第四章 分支结构**

**一、键盘录入(Scanner)（重点）**

1.导包 import java.util.Scanner;

2.创建Scanner变量 Scanner sc = new Scanner(System.in);

3.接收数据

**接收字符串 ：**

next():遇到空格和回车结束

nextLine():遇到回车结束

nextInt():接收int类型整数

nextDouble():接收double类型的小数

next().charAt(0):接收字符串中的第一个字符

**二、if语句(选择结构)（掌握）**

1.基本if语句

**if(布尔类型表达式 ,条件为真){**

**代码块 //如果条件为真，就执行这里！**

**}**

**执行流程:**

判断布尔类型表达式，结果为true执行代码块，然后继续执行后续代码

结果为false，跳过代码块执行后续代码。

2.if...else （如果......否则......）

**if(布尔类型表达式){**

**代码块1 //如果条件为真，就执行这里！**

**}else{**

**代码块2 //条件不成立，就执行这里！**

**}**

**执行流程:**

布尔类型表达式为true，执行代码块1

布尔类型表达式为false，执行代码块2

3.多重if

**if(布尔类型表达式1){**

**代码块1**

**}else if(布尔类型表达式2){**

**代码块2**

**}else if(布尔类型表达式3){**

**代码块3**

**}...**

**else{**

**代码块4**

**}**

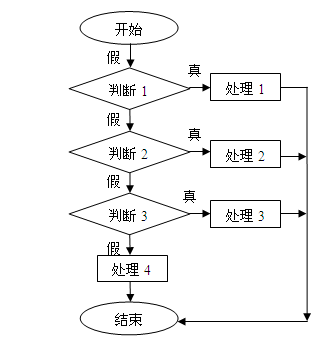
**执行流程:**

判断布尔类型表达式1，为true，执行代码块1，否则，

判断布尔类型表达式2，为true，执行代码块2，否则，

判断布尔类型表达式3，为true，执行代码块3，以上判断

均为false执行代码块4



4.if嵌套

概念:在一个完整的if结构中嵌套另一个。

**语法:**

**if(){**

**//如果条件为真，就执行这里！**

**if(){**

**//如果条件为真，就执行这里！**

**}else{**

**//条件不成立，就执行这里！**

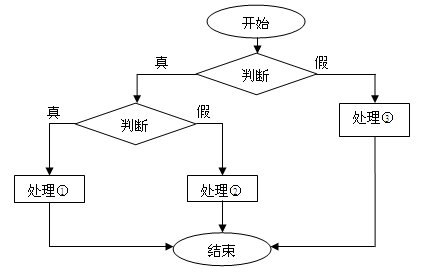
**}**

**}else{**

**//条件不成立，就执行这里！**

**}**

嵌套if语句可以在条件内，针对真或假的情况，再指定条件进行判断，从而拥有执行更多分支的功能



5.switch(分支结构)(掌握)

if-else：只有一个条件分支时使用。

嵌套if：多个条件时使用。

多重if：多个分支时使用。

switch语句可以替代多重if语句,语法:

//变量可以是byte、short、int、char、[String（JDK7+）时可使用] 其中变量|表达式不能是long String boolean类型！

**语法：**

**switch(变量|表达式){**

**case 值1:**

**代码块1**

**break;**

**case 值2:**

**代码块2**

**break;**

**case 值3:**

**代码块3**

**break;**

**...**

**default:**

**为满足条件**

**}**

**执行流程:**

先判断变量与值1是否相等，相等执行代码块1，否则

判断是否与值2相等，相等执行代码块2，否则

判断是否与值3相等，相等执行代码块3，直到

所有case都不相等执行default

**注 意:**

1.每一个case中需要加break。

2.default可以写在switch任意位置，写在switch末尾不需要加break。

3.每一个case的值不能重复。

6.if和switch区别?

（1）if用于判断区间值

（2）switch用于等值判断

**三、块作用域**

大括号（｛｝）括起来的内容称为“块”。

作用域：全局和局部

全局变量和局部变量：

全局变量又称外部变量，作用域是从定义点到程序结束。

局部变量又称内部变量，作用域仅限于代码快内部使用，代码快外部使用是非法的。

局部变量：概念:定义在函数中，必须先赋值再使用。

作为范围:定义行到所在代码块结束。

命名冲突:多个变量，作用范围重合的情况下，不能重名。

**第五章 循环结构**

**一、while循环**

**语法：**

**while(布尔类型表达式){**

**循环操作(循环体)**

**}**

**执行流程：**

判断布尔类型表达式如果为true,执行循环体，然后回到判断条件，为true继续执行循环体，直到为false循环结束。

特点：首次有入口条件，先判断，再执行，适用于循环次数明确的情况。

**二、do...while...**

**语法:**

**do{**

**循环操作(循环体)**

**}while(条件判断);**

**执行流程:**

先执行循环操作，然后做判断，结果为true则再次执行循环操作，直到结果为false循环结束。

特点:首次执行没有入口条件，先执行，再判断，适用于循环次数不明确情况。

**三、for循环**

**语法:**

**for(初始部分; 条件判断; 迭代部分){**

**循环操作(循环体)**

**for (初始化语句①; 条件语句②; 控制语句③)**

**{ 循环体： 若干语句④ }**

**}**

**执行流程:**

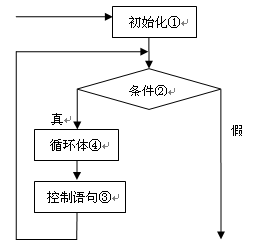
1.执行初始部分(仅执行一次)。

2.执行条件判断，结果为true。

3.执行循环操作

4.执行迭代部分，再次回到判断条件，为true继续执行循环操作，直到为false循环结束，执行后续代码。

特点:首次有入口条件，先判断，再执行，适用于循环次数明确的情况。



**初始化语句负责完成变量的初始化（initialization）**

**条件语句是值为 boolean型的表达式，称为循环条件（condition）**

**控制语句用来修整变量，改变循环条件（iteration）**

**①②③之间一定要使用分号隔开**

采用了for循环后，程序要比采用while循环更加简洁一些，初始化、递增以及条件语句都在for循环语法内部，循环体内只保留需要做的事情

**四、java中三个循环的区别?(重点)** IMG_256IMG_256IMG_256IMG_256IMG_256

while、for:首次有入口条件，先判断，再执行，适用于循环次数明确的情况。

do...while:没有入口条件，先执行，再判断，适用于循环次数不明确的情况。

for比while语法更清晰，可读性更高。

**五、break、continue关键字**

break:用在switch和循环中，表示结束switch和结束当前循环。

continue:用在循环中，表示跳出本次循环，继续执行下次循环。

**六、循环的嵌套应用(掌握)**

总结: 外层循环控制行

内层循环控制列

1. **数组**

**一．了解数组**

1.概念：

（1）数组是为了解决同类数据整合摆放而提出的，可以理解为一组具有相同名称的变量的集合，它的每个元素具有相同的数据类型。

一组连续的存储空间，存储多个相同数据类型的值。

(2)在内存中连续存放

(3)数组分为一维数组和多维数组，可以用一个统一的数组名和下标来唯一地确定其中的元素。

2.数组的特点:（背）

1.类型相同 2.长度固定

3.数组的使用:

语法:  **数组名[下标]:访问数组元素**

下标:从0开始，依次+1，最大下标是数组长度-1.

4..数组的声明

**<data\_type>[] <array\_name>; 或 <data\_type> <array\_name>[];**

5.注意：

①．数组的类型实际上是指数组元素的取值类型。对于同一个数组，其所有元素的数据类型都是相同的。

②．数组名的书写规则应符合标识符的书写规定

③．数组名不能与其它变量名相同

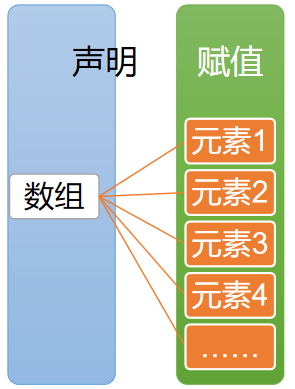
④．在数组声明中包含数组长度永远是不合法的！

**二．数组的初始化**

1.数组声明后，并不能直接使用;必须要给它分配内存，初始化以后才可以使用

2.数组的初始化有两种方式：

**静态初始化**  **动态初始化**



1. **静态初始化**

静态初始化的方式是在声明数组变量的同时进行的。

这种方式不仅定义了数组当中包含的元素的数量，而且指定了每个元素的值



1. **动态初始化**

静态初始化的方式在声明数组时就必须定义数组的大小，以及每个元素的初始值。

如果要定义的数组长度或数组数据只有在运行才能决定，就要使用动态初始化

声明与初始化语句分开写时，两条语句中的数组名、类型标识符必须一致



动态初始化数组时也可以使用变量的来确定义的数组大小（只能是int、short等整数

）



**三、数组的遍历（掌握）**

1.概念:依次访问数组中的每一个元素。

**代码:**

for(int i = 0; i<数组名.length; i++){

System.out.println(数组名[i]);

}

**四、数组默认值**

整数(byte short int long):0

小数(float double):0.0

字符(char):空格

布尔(boolean):false

引用类型(String...):null

**五、数组的创建**

方式1：先声明，再分配空间（掌握）

语法:数据类型[] 数组名; 数组名 = new 数据类型[数组长度];

举例：byte[] b;//声明

b = new byte[10];//分配空间

​

方式2:声明并分配空间 （掌握）

语法:数据类型[] 数组名 = new 数据类型[数组长度];

举例:int[] a = new int[5];

​

方式3:声明并赋值（复杂）（了解）

语法:数据类型[]

语法:数组名 = new 数据类型[]{value1,value2...};

举例:int[] a2 = new int[]{1,2,3,6};

​

方式4:声明并赋值(简单) （掌握）

语法:数据类型[] 数组名 = {value1,value2...};

举例:int[] a3 = {1,2,3,4,5,6};

**六、数组扩容（重点）**

1.使用循环依次复制。

2.System.arraycopy(原数组名称， 原数组起始， 新数组名称， 新数组起始， 复制的长度);

3.java.util.Arrays.copyOf(原数组名称，新数组长度);该函数会返回一个新数组，新数组的数据类型和原数组是一致的

### 七、**数组地址的替换(掌握)**

举例: int[] arr = {1,2,3};//数组arr保存的是数组的地址0x01

int[] array = java.util.Arrays.copyOf(arr, arr.length\*2);

//得到一个新数组的地址0x02

arr = array;//地址的替换，arr指向新数组保存的是0x02

### **数组作为形式参数和返回值类型(掌握)**

数组作为形式参数接收的是地址。

数组作为返回值类型返回的是地址。

1. **数组排序**
2. 可变长参数(背)

（1）概念:定义在函数上，可以接收多个相同类型的数据，使用方和数组一样。

（2）语法:数据类型... 形参名

注意:一个函数中只能有一个可变长参数并且必须在形参列表的最后。

1. 冒泡排序(掌握)

（1）**概念:**相邻两个元素，两两比较，互换位置。

（2）代码:

for(int j = 0; j<a.length-1; j++){

for(int i = 0; i<a.length-1-j; i++){

if(a[i] > a[i+1]){

int temp = a[i];

a[i] = a[i+1];

a[i+1] = temp;

}

}

}

1. 选择排序(掌握)

**概念:**固定值与每一个值做比较，互换位置。

for(int j = 0; j<a.length-1; j++){

for(int i = j; i<a.length-1; i++){

if(a[j] > a[i+1]){

int temp = a[j];

a[j] = a[i+1];

a[i+1] = temp;

}

}

}

1. 快速排序(掌握)

java.util.Arrays.sort(数组名);//只能升序

**十、二维数组的定义与使用**

1、定义

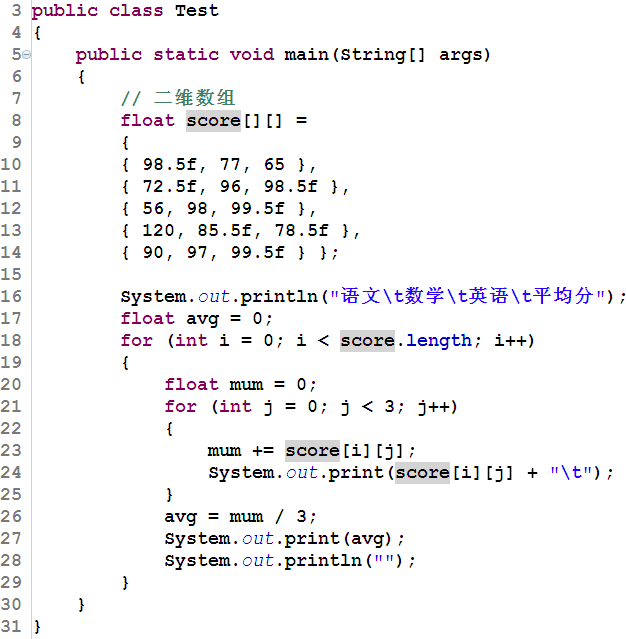
在实际问题中有很多数据是**二维的或多维的和一维数组相同**，二维数组也是有序数据的集合，数组中的每个元素具有相同的数据类型。我们可以把二维数组理解为一维数组的集合

**<datatype> [][] <array\_name>;**

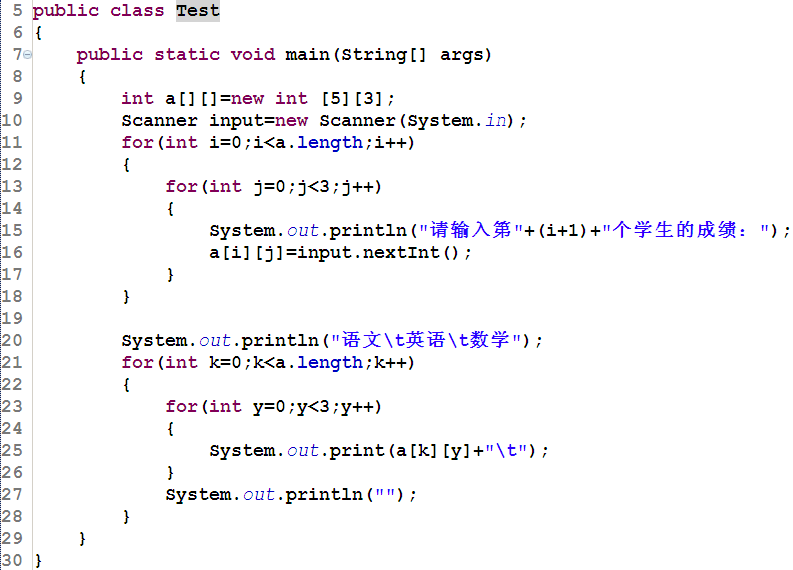
**如： float [][]stu\_scores ;**

需求：某个小组有5位学员，每位学员有3门课程的成绩，编写程序统计平均成绩。这时要定义多个一维数组。

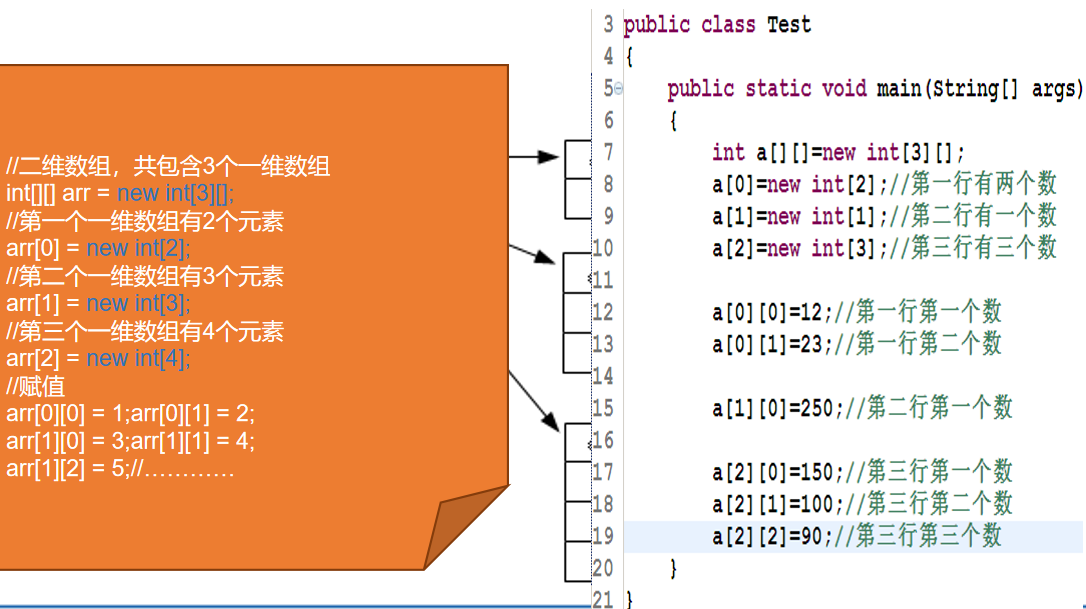
（1）多维数组静态初始化



（2）多维数组动态态初始化



（3）不规则的二维数组



2、二维数组的创建(了解)

**方式1:先声明，再分配空间**

语法:数据类型[][] 数组名;

数组名 = new 数据类型[行][列];

**方式2:声明并分配空间。**

语法:数据类型[][] 数组名 = new 数据类型[行][列];

**方式3:声明并赋值(麻烦)。**

语法:数据类型[][] 数组名 = new 数据类型[行][];

数组名[行] = new 数据类型[列];

**方式4:声明并赋值(简单)。**

语法:数据类型[][] 数组名 = { {1,2,3},{4,4,4,4},{4,1,1,23} };

​

**遍 历:**

for(int i = 0; i<数组名.length; i++){

for(int j = 0; j<数组名[i].length; j++){

System.out.print(数组名[i][j]);

}

System.out.println();

}

**十一．数组总结**

（1）数组是内存中有序数据的集合，数组中每个元素具有相同的数据类型，且在内存中的顺序是相邻的。

（2）数组必须先声明，对其初始化以后才能使用。对数组初始化分为静态初始化和动态初始化两种方式。

（3）通过下标来访问数组中的每个元素，注意下标是从0开始的。

（4）数组可以分为一维数组和多维数组。二维数组可以理解为一维数组的集合。

（5）数组结合循环可以实现很多有用的功能，例如求最大值、最小值、平均值、搜索数据、排序等等。

1. **Java中的方法**
2. **JAVA的方法**

1.方法

1. 概念：

来源于生活，指解决问题的处理途径。Java中，main方法是解决问题的入口

（2）把所有代码都放入main方法中有什么缺陷？

代码庞大 难以阅读 难以维护 代码难以重用 不利于合作开发

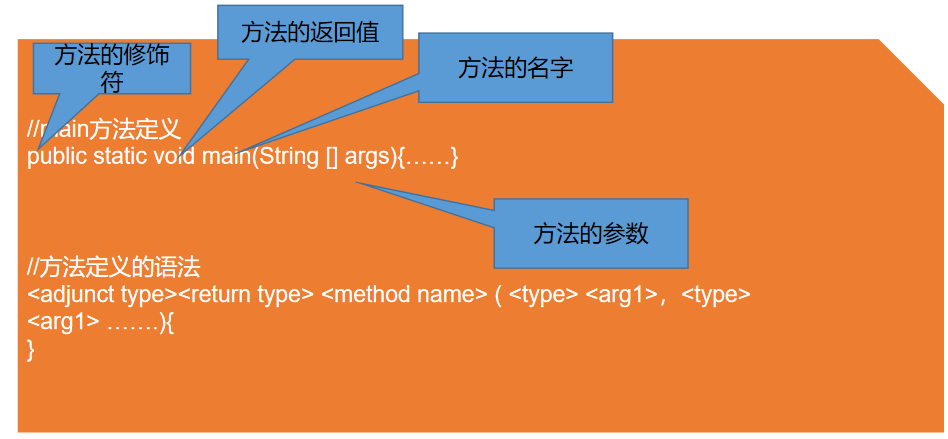
（3）如何解决？

将庞大的程序划分为不同的功能小块，逐个突破

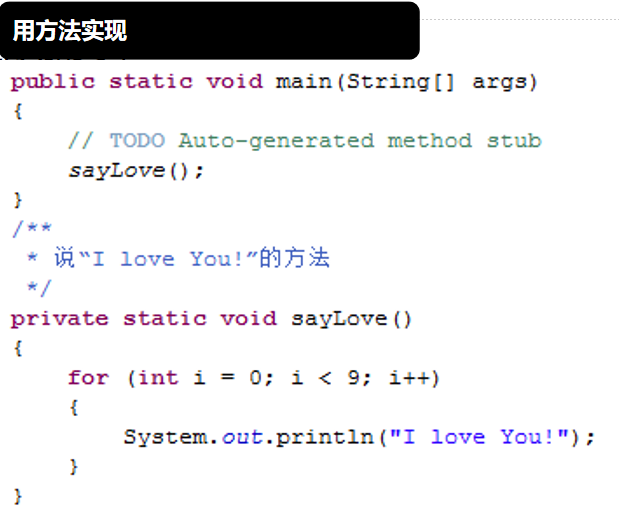
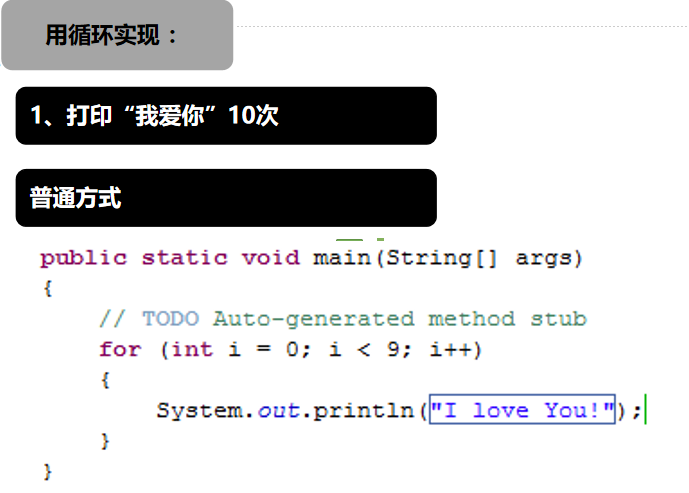
在main方法中组织各个功能块，组合成整个功能体

2.方法的定义

**和main方法类似：**

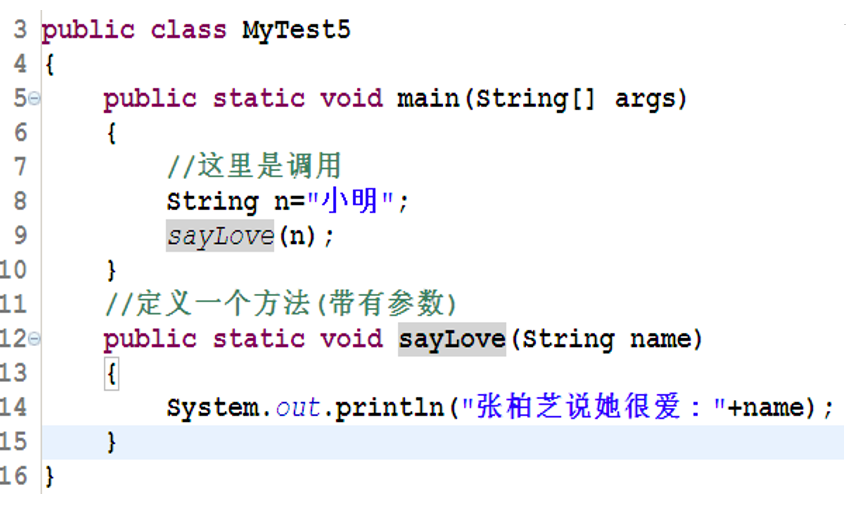


**案例：**

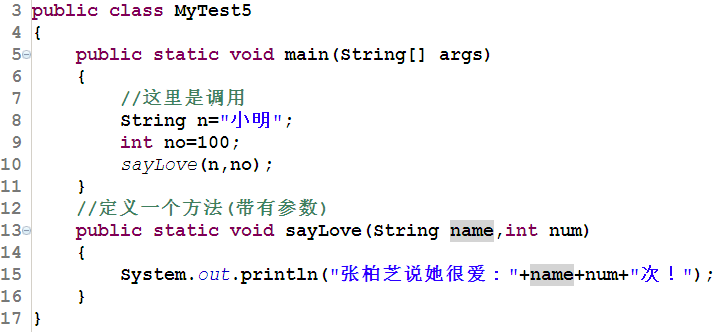


1. **方法的参数**

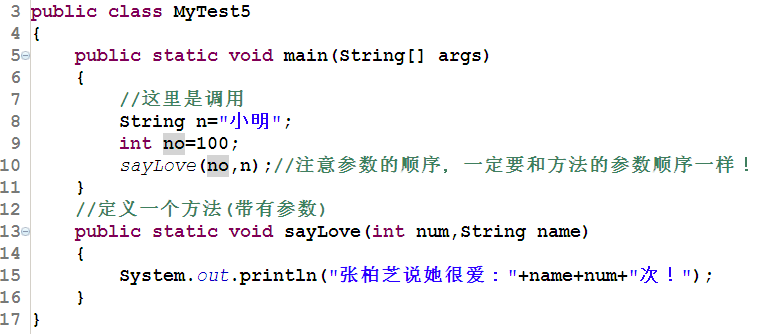
1、带参数的方法



2、带多个参数的方法



3、参数的顺序

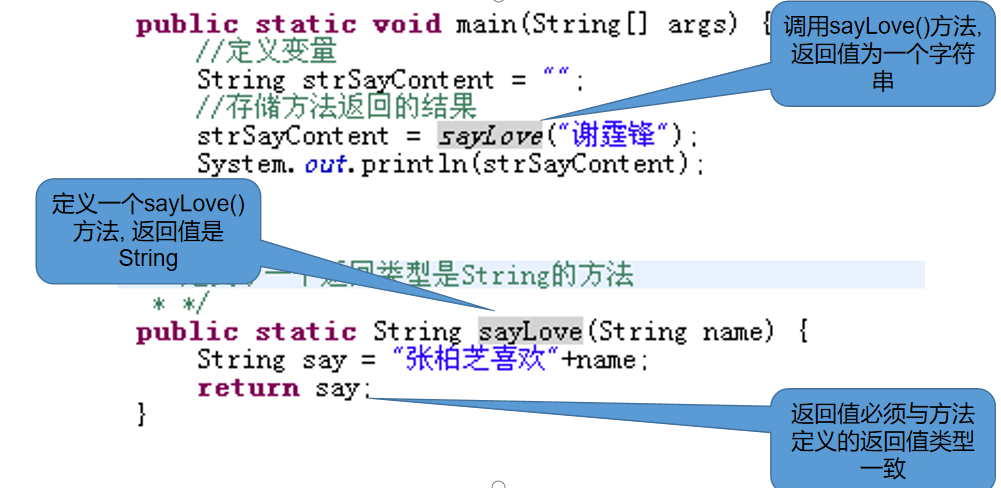


4、方法的返回值

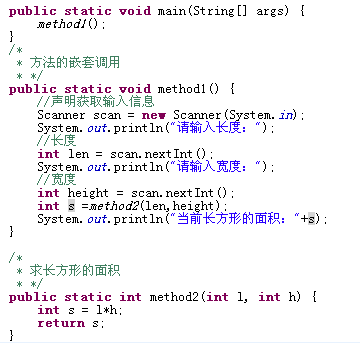
·方法可以完成一定的功能，也可以在执行完毕后返回一定的结果。

·每个方法是否需要返回值，返回值的类型，都是我们在定义方法前需要考虑清楚的问题。

·如果一个方法有返回结果，需要在方法体中使用return语句。返回出的目标数据类型必须和方法定义的返回类型一致。



案例：修改一个刚刚求长方形面积的方法，试试，能不能改成带返回类型的方法？



5、形参与实参

·方法的参数分为形式参数和实际参数，简称为形参和实参

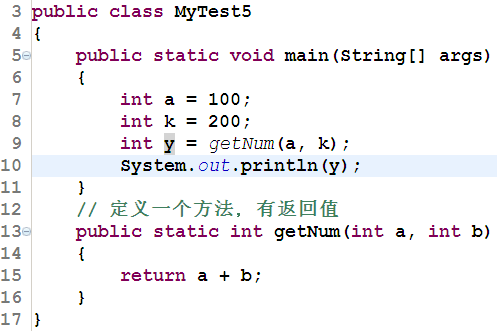
·形参是指定义方法时方法列表中的参数（beign和end）

·实参指的是方法调用时传递的参数

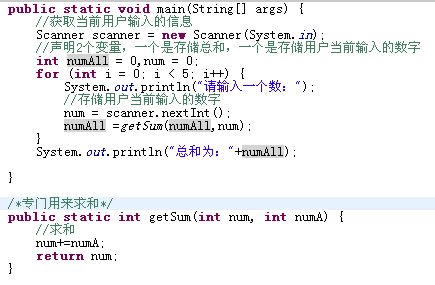
·定义一个方法时，形参的值是不确定的，它的值是由实参传递的

·形参、实参的个数、类型、顺序必须是匹配的。方法需要什么类型的参数列表，在调用时，就要传递什么类型的参数

·使用普通变量作为参数，形参改动不会影响实参，数组作为参数，形参数组的值改变，实参数组中的值也会改变



案例：用户连续输入几个数，求和。使用带有参数的方法完成



6、以数组作为参数的方法

数组形参与普通形参的区别

案例：如何存储用户连续输入的10个学生的名字，并再打印出来？

使用普通变量作为 参数，形参改动不会影响实参，数组作为参数，形参数组的值改变，实参数组中的值也会改变。

**三、变量的作用域**

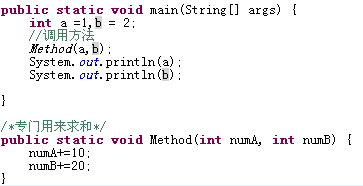
·块作用域，块由左右两个大括号包含，块作用域内定义的变量，只能在本块内使用

·在方法内定义的变量，只能在本方法内使用

·一个方法如果有参数，那么该参数也被包含于方法的作用域内

·作用域可以嵌套，如果我们定义了一个方法，在方法内定义了一个变量，那么在该方法中的任意块中（如方法内的if语句块），该变量均可以使用；反之，如果我们在内部块中定义变量，那么在块的外部是不可以使用的。

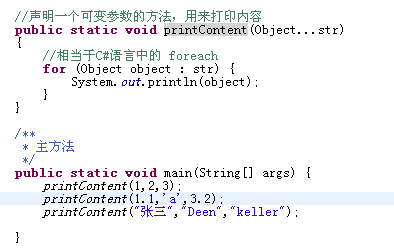
**案 例：**



**四、可变的参数的方法**

1·Java语言在JDK1.5中推出Java可变参数，简称varargs。这个新语言特征给软件开发人员在编写方法时提供了方便性和灵活性。

2·它主要适用于参数个数不确定，类型确定的情况。



**·注意的地方：**

1、可变参数的方法只有在必须的时候才会调用，如果有确切匹配的不可变参数的方法，会优先选择不可变参数的方法。

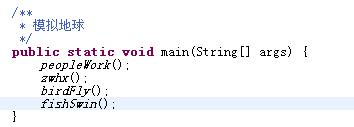
2、如果两个方法都是可变参数，都能匹配时，会编译通不过；编译器不知道调用哪个。

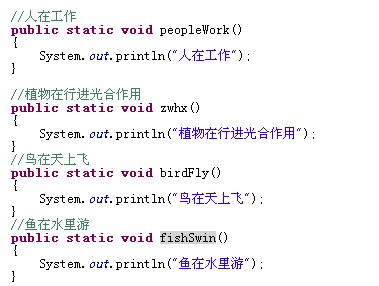
3、一个方法只能有一个可变参数，并且这个可变参数必须是该方法的最后一个参数。

1. **类之间方法的相互调用**

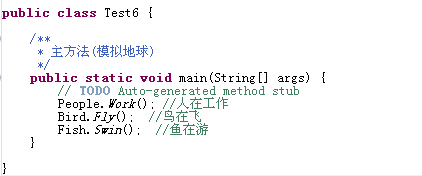
1.为了把功能块拆分出来，我们采取了对一个功能块编写一个方法的方式，但是如果一个类里面的方法太多，也会显得类太乱，而且方法太多，之间又没有什么联系，会造成类的功能不明晰。

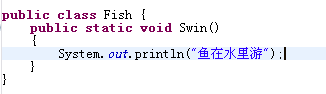
2.为了解决这个问题，可以编写多个类，类中只定义相关方法，类与类之间相互调用，完成整个功能。

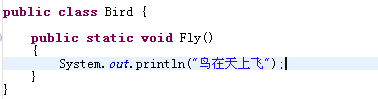


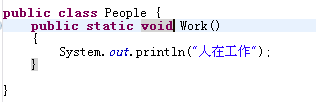


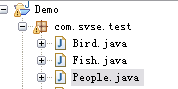
**案例：**











**六．总结**

使用普通变量作为 参数，形参改动不会影响实参，数组作为参数，形参数组的值改变，实参数组中的值也会改变。

可变参数适用于参数个数不确定，类型确定的情况，Java把可变参数当做数组处理。

对于public static 类型的方法，可以直接通过类的名称带上点操作符来调用。

1. **函数**

**一、认识函数**

1.概念:实现特定功能的一段代码，反复使用。

2.语法:(掌握)

**public static void 函数的名称(){**

**函数主体**

**}**

函数主体:程序中出现多次重复的代码应该定义在函数内部。

3.函数的调用: 函数名称();

4.注意:

当遇到函数调用，会优先执行函数内部代码，执行完毕后程序流程回到调用位置。

5、带有参数的函数语法:

**public static void 函数名称(形式参数){//形式参数等同于局部变量的声明**

**函数主体**

**}**

调用语法:函数名称(实际参数);

注意:调用函数时要保证实际参数列表与形式参数列表顺序，个数，类型一致。(背)

形式参数(形参)

实际参数(实参)

多个形式参数:形式参数列表(形参列表)

多个实际参数:实际参数列表(实参列表)

**二、返回值与返回值类型**

**public static 返回值类型 函数名称(形式参数){**

**函数主体**

**return 返回值;**

**}**

调用:

变量 = 函数();

注意:返回值类型必须和返回值一致。

**三、return的用法**

**两种情况:**

1.应用在返回值类型为void函数中。

语法:return;//表示结束当前函数，直接回到调用位置

2.应用在返回值类型除void以外函数中。

语法:return value;//表示结束当前函数，并伴有一个结果返回到调用位置

**四、函数总结**

注意:

1.函数之间并列关系。

2.函数不能嵌套。

3.一个函数只完成一个功能。

​

好处:

增强可读性

提高复用性

减少代码冗余

提高可维护性

方便分工合作

**五、递归**

概念:函数中调用本身的现象。

**注意:**递归必须有出口条件，否则就是无穷递归(死递归)。

递归次数不能太多，否则就会内存溢出。