# Java编程思想初级

## Java语言介绍

Java是单继承，跨平台的语言，可以在Linux系统中运行，也可以在Windows系统种运行

Java使用Unicode字符集，65535个字符

## 标识符

标识符是指，由字母，数字，下划线及美元符号等组成的一个字符序列

变量、方法名称、类名、接口名称等符合标识符规则

标识符规则：

1. 数字不能开头
2. 第一个字符之后，可以是货币符号、连字符、数字、字母、汉字的任意组合
3. 标识符的字符数量没有限制，一般不要太长
4. Java的关键字（保留字）不能作为标识符
5. Java的标识符是大小写敏感，区分大小写
6. 标识符中不能使用空格，单双引号、分号等

## 变量

变量：保存数据的存储单元（容器），

声明变量语法：类型 变量名称

String str;

## Java数据类型

可分为两大类：基本数据类型、非基本数据类型

基本数据类型：四类八个

整数、浮点数、字符类型、布尔类型（逻辑型）

整型：int、byte、short、long

浮点型:float（单精度浮点数，小数点后面能存7位）、double（双精度浮点数，小数点后面能存15位）、

字符型：char

逻辑型：Boolean

非基本数据类型：由基本数据类型组合成的新类型，例如数组、字符串、类、

## 声明数据类型变量、常量

整型：

十进制整型：Int int10 = 23;

八进制整型：int int8 = 012; //八进制的012代表十进制的10

十六进制整型：int int16 = 0XFF; //十六进制的）0XFF就是十进制的255，十六进制的数0X开头，从10开始依次位A-F,10位A,十一位B,………………15位F

Byte类型：

//一个字节，能存： -128 — 127范围内的数

Short短整型：

//两个字节，能存： -32768 — 32767范围的数

Int整型：

//四个字节，能存：-2147483648 — 2147483647范围的数

Long长整型：

//八个字节，能存：-9223372036854775808 — 9223372036854775807范围的数

上溢：比如最大值 + 1，就会变成最小值、下溢：比如最大值 – 1，就会变成最大值，这是因为计算机使用二进制数存储数据的原因

浮点类型：

Float：单精度浮点小数，四个字节

Double：双精度浮点小数，

布尔类型数据（逻辑型数据）

//true-真 ， false-假 ，占一个字节，包装类Boolean，默认值为false

字符类型：

//char，一个字符两个字节，16个位（比特bit）

## 简单数据类型转换和类包装

各类简单数据类型的转换优先关系

Byte,short,char->int->long->float->double

## 高精度数字

BigInteger:高精度的整数

BigDecimal:高精度的小数

与基本数据类型最主要的区别就在于，高精度数字在进行加、减、乘、除等运算的时候通过实例对象调用方法，而不是跟基本数据类型那样，通过运算符（+、-、\*、/）做运算

## 类与对象

计算机编程语言尽力了三个过程

1. 面向机器：程序 = 数据 + 指令
2. 面向过程：C语言，程序 = 数据结构 + 算法
3. 面向对象：程序 = 对象 + 消息（对象与对象之间的交互方式）

面向对象相关术语

1. 抽象：abstract
2. 对象：每个对象都有自己的状态、行为、名称、方法
3. 消息：对象之间的交互方式
4. 类： 程序由类构成
5. 继承：一个类在现有类的基础上，对父类进行扩充
6. 多态：
7. 封装：私有的数据成员，给公共的接口去操作

类和对象

类是一种或一类对象的设计图纸，

对象是类的实例

类定义格式：[修饰符关键字] class 类名 [extends 父类名] [implements 接口列表] {//类体内容}

所有的类都是Object的子类

This. 没有加前缀thi一般是参数，加了前缀this是类的变量，this一般用于参数和类变量重名时，这时用this指定来区分

## 类和成员的修饰符

访问权限修饰符：



# Java编程思想中级

## 字符串\_正则表达式

正则表达式就是按某种表达式来表达特定的字符串

[\\w](file:///\\w)+:表示非词字符，除了数字和字母外的字符都是非词字符

^:用在不同得地方表示不同得含义，用在中括号里表示非的意思，直接放在字符串里表示开头开始位置，

$ 结尾位置，表示末尾必须匹配啥

|表示或得意思

-表示范围：比如[a-zA-Z0-9] ,相当于\w

\W和\w相反

\s 空白符：

\S 非空白符

\d 表示0-9数字 [0-9]

\D 表示非数字 [^\d]

? 表示指定字符出现的次数，零次或一次

\* 表示某个字符可以出现任意次数

+ 表示某个字符出现一次或多次

{n} n次，刚好n次，n是非负整数

{n,} 至少n次，n是非负整数

{n,m} n至m次， n <= m n和m是非负整数

## 类型信息

Object obj = null;

If(obj != null) {

Obj.xxx方法（）

}这种代码显得枯燥，有很陋，这时可以引入空对象

空对象：可以看作是策略模式的特例，空对象有一种变体叫空迭代器模式

空对象最有用的地方在于它更靠经我们的实际的数据

可以设计空对象，来响应具体实例对象所应具有的所有响应，防止产生空指针等不必要的麻烦