1. 自动装箱

list.add(3);会进行自动装箱->list.add(Integer.*valueOf*(3));

2.自动拆箱

list.get(0); -> list.get(0).intValue();

1. Integer n = 3; 编译器会自动装箱

n++; 编译器会先自动拆箱再装箱

1. 数值提升，如Integer \* Double，也会进行拆箱和装箱

Integer a = 1; //a先自动装箱  
Double b = 2.0; //b先自动装箱  
System.*out*.println(a \* b); //a先自动拆箱，提升为double，b自动拆箱，相乘，再把结果装箱成Double

1. 拆装箱是编译器做的，不是虚拟机做的
2. 包装器里面的value是final类型的，所以值是不能改变的，要想改变值要用IntHolder等类型。
3. 用于转换

//解析成int类型,参数用10进制表示  
int b = Integer.*parseInt*("1");  
//解析n进制字符串成10进制的int类型  
int c = Integer.*parseInt*("100", n);

//将其解析成Integer类型，参数为10进制  
Integer c = Integer.*valueOf*("100");  
//将其解析成Integer类型，参数为n进制  
Integer d = Integer.*valueOf*("100", n);

1. 自动装箱类型比较要用equals或者compareTo,因为用==等可能出现错误

如boolean、byte、char<=127, short、int 介于-128~127才能==有效，因为它们会被包装到固定的对象中去。

CompareTo 如果为Boolean，true.compareTo(false)返回1，false.compareTo(true)返回0

a.equals(b);  
a.compareTo(b);

1. 包装器类型Long，Integer，Double，Float，Short，Byte都继承自Number抽象类