这里要写的依旧是Java的类型转换

**Int—》String**

（1）String s = Integer.toString(Int i);

（2）String s = String.valueOf(int i);

**String—》Int**

（1）Integer I = Integer.valueOf(String s);//Integer.valueOf(s) 相当于 new Integer(Integer.parseInt(s))，也会抛异常，但会多产生一个Integer对象

（2）Int I = Integer.parseInt(string s); //直接使用静态方法，不会产生多余的对象，但会抛出异常

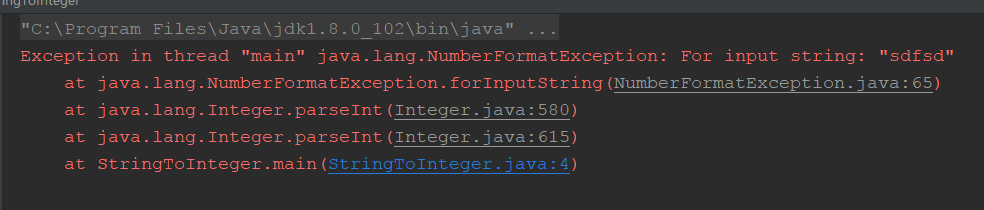
注意：Integer.valueof返回的是Integer对象，Integer.paseInt返回的是int

（3）Int I =12345

String s = “”

S=i+”” //会产生两个String对象,一个是开始的空字符串，一个是后来int+空

那么当字符串为字符或者为空的时候，方法还能用吗？亲自做了测试，报了这样的错误



那么如何将为字母的字符串转化成数字呢？

String s = "ab";  
for(byte b:s.getBytes()){ //将字母转换成为数字  
 System.*out*.println(b-96);  
}

Char – 96 也可以实现字母转化成为数字

当一个字符串中既包含整数，又包含字母的时候怎么办，不能用数字减去96直接添加数字就好了。需要用正则表达式，进行匹配。Reg = [A-Za-z],然后用String.matchs(reg)匹配。如何做到一个一个字符匹配呢？需要把String，转化成为char的数组，用string。ToCharArray（）方法，然后用foreach遍历。注意，遍历的是char类型，进行匹配的时候需要把char类型转化成为string，用String.ofValue(char c)来完成。最后将结果逐个加入stringbuffer中。

代码如下：

String reg = "[a-zA-Z]";  
String s ="213 sdf 123";  
StringBuffer stringBuffer=new StringBuffer();  
if(s==""){  
 System.*out*.println("String is empty!");  
}  
for(char c : s.toCharArray()){ //将字母转换成为数字  
 if(String.*valueOf*(c).matches(reg)){  
 stringBuffer.append(c-96);  
 }  
 else{  
 stringBuffer.append(c);  
 }  
}  
System.*out*.print(stringBuffer);