Chapter 4 数学函数、字符和字符串

4.1 常用数学函数

Java中有一个Math类，是一个final类，再java.lang.Math包中，所有数学函数都是静态方法。

- Math类中定义了常用的数学常量
  - PI：圆周率
  - E：自然对数
- Math类中定义了常用的数学方法
  - 三角函数：sin, cos, tan...
  - 指数：exp, log, log10, pow, sqrt...
  - 取整：ceil, floor, round
  - 其他：min, max, abs, random([0.0,1.0))

    通过Math.random可以实现随机生成字符，代码参见Coding/.../ch4/RandomCharacter。

4.2 字符串数据类型和操作

4.2.1 Unicode 和 ASCII

Unicode包括ASCII码， 从'\u0000' 到 '\u007f' 对应128个ASCII字符，java中的ASCII字符也可以用Unicode表示，例如 `char a = '\u0041'` 等价于 `char a = 'A'`。

4.2.2 字符型数据和数值类型数据之间的转换

char类型数据可以转换成任意一种数值类型，反之亦然。将整数转换成char类型数据时，只用到该数据的低16位，其余被忽略。如 `char ch = (char)0xAB0041`,其中高位AB被忽略掉，只将0041进行转化，ch值为A。将浮点数转成char，先将浮点数转为int型，再转为char。如果转化不产生精度损失，可以采用隐式转换，否则必须强制类型转换。

4.2.3 Character类

Character类时char的包装类，它的作用在于：将char类型的数据封装成对象；包含处理字符的方法和常量。该类有对字符进行类型判断和对字符进行大小写转换的常用方法。

4.3 字符串类型

4.3.1 String类

String类是一个final类，不能被继承，表示一个固定长度的字符序列，实例化后其内容不能更改。不能更改不是真正意义上的不能更改，而是一旦更改了，原来的字符串引用次数为0，那就被垃圾回收了。

String类的实例化：

```java
// 从字面值创建字符串
String message = new String("Welcome to Java");
// 简写形式
String message2 = "Welcome to Java";
```

4.3.2 规范字符串和常量池

```java
String m1 = "Welcome";
String m2 = "Welcome";
String m3 = "Wel" + "come";
String m4 = "Wel" + new String("come");
m2 == m1;       // true
m1 == m3;       // true
m1 == m4;       // false
```

通过上例可以感觉到java的String存储机制有悖我们的直觉，是因为java中存在常量池。为了提高效率和节省内存，Java中的字符串字面值维护在字符串的常量池中。

实际上上面的m1和m2和m3指向的都是常量池中的“Welcome”，因此它们都指向的是相同的地址，而m4中new String("come")不是一个字面值，因此m4指向的时堆中的一个地址。

4.3.3 字符串常用方法

- String.equals(String)：两个字符串比较
- String.equalsIgnoreCase(String)：两个字符串比较，忽略大小写
- String.regionMatch(int, String, int, int)：比较两个字符串的一部分
- String.startsWith(String)：判断是否以某个字符串开头
- String.endsWith(String)：判断是否以某个字符串结尾
- String.length()：获取字符串长度
- String.charAt(int)：获取字符串在某个位置的字符
- String.substring(int, [int])：字符串截取，返回新字符串
- String.toLowerCase()：转成小写，返回新字符串
- String.trim()：删除首尾空格，返回新字符串
- String.replace(String, String)：字符串替换，返回新字符串
- String.indexOf(char)：查找某个字符的首次出现位置，返回-1表示未找到

4.3.4 StringBuilder和StringBuffer

String类一旦初始化完成，字符串就不可修改，但是StringBuilder和StringBuffer初始化后还可以修改字符串，StingBuffer修改缓冲区的方法时同步的，更适合多线程环境。StringBuilder线程不安全，与StringBuffer工作机制类似。StringBuilder和StringBuffer还拓展了字符串的方法，例如append，insert，reverse等。

4.4 格式化控制台输出

JDK1.5之后支持`System.out.printf();`格式化输出字符串。举例如下：

```
    public static void main(String[] args) {
        System.out.printf("boolean:%6b\n", false);
        System.out.printf("boolean:%6b\n", true);
        System.out.printf("character:%4c\n", 'a');
        System.out.printf("integer:%6d,%6d\n", 100,200);
        System.out.printf("double:%7.2f\n", 12.23);
        System.out.printf("String:%7s\n", "JAVA");
    }
```