



第四章 数组

南京农业大学

谢忠红



数组（是引用类型）

Java数组与C/C++的不同之处：

C/C++： 数组是基本类型

例：`int a[5];`

Java： 除了基本类型外，其他类型的数据全部是对象。

例：`int a[];`

`a=new int[5];`

数组的定义

数组是相同类型数据的有序集合。

举例：

```
■ double [] sheep;
```

```
■ sheep = new double [7]; //为数组分配空间
```

■ 或者


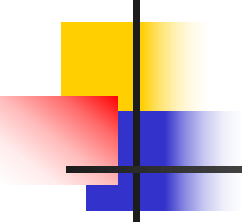
```
■ double []sheep=new double [7];
```

例1 将两个一维数组中的对应元素相加后显示出来。

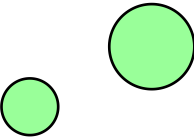

```
import java.io.*;
public class cal2{
    public static void main(String[] a) {
        int i;
        int[] x={10,9,8,7,6}; int[] y={1,2,3,4,5};
        int[] z = new int[5];
        for(i=0; i<x.length ; i++){
            z[i]=x[i]+y[i];
            System.out.println(z[i]);}
    }}
```

数组元素个数为:

数组名.length



数组元素
可以是对象吗?



```
■ class Sheep{  
■     int ID;  
■     double weight;  
■     String type;  
■     Sheep(int ID,double weight,String type){}  
■ }
```

创建对象数组



创建对象数组

```
Sheep [] MrLi=new Sheep[3];
```

初始化:

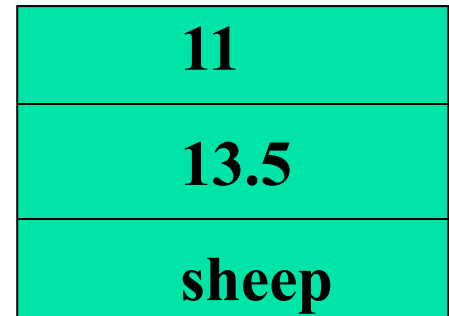
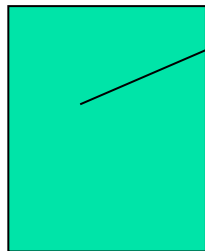
```
MrLi[0]=new Sheep( 2,23,"goat");
```

```
MrLi[1]=new Sheep(11,13.5,"sheep);
```



■ 内存布局如下:

MrLi



- 举例：
- 采用面向对象的方法编程：求任意四边形的面积，已知该四边形四个点分别是
- **P1(20,30) p2(80,40)**
- **p3(90,100) p4(10,70)**
- 要求：创建**point**类和**quadrangle**类
-

举例

```
Class Point {
```

```
    int x, y;
```

```
    public point (int x1,int y1){  
        x=x1;y=y1;}
```

```
    public double distance(Point other){
```

```
        double dis= Math.sqrt( Math.pow(this.x-other.x,2)  
                +Math.pow(this.y-other.y,2) );
```

```
        return dis;
```

```
    }
```

```
}
```

```
class quadangle {
```

```
    Point[] points=new Poinit[4]; //定义数组对象
```

```
    public quadangle(Point[] pointArray){  
        points=pointArray;    }
```

```
    double area(){
```

```
        double dis0=points[0].distance(points[1]);
```

```
        double dis1=points[1].distance(points[2]);
```

```
        double dis2=points[2].distance(points[3]);
```

```
        double dis3=points[3].distance(points[0]);
```

```
        double dis4=points[0].distance(points[2]);
```

```
        double p1=0.5*(dis0+dis1+dis4), p2=0.5*(dis2+dis3+dis4);
```

```
        double area1=Math.sqrt(p1*(p1-dis0)*(p1-dis1)*(p1-dis4));
```

```
        double area2=Math.sqrt(p2*(p2-dis2)*(p2-dis3)*(p2-dis4));
```

```
        return area1+area2;    }
```


```
}
```

4.2 二维数组



工资项 员工号	基本工资	职务津贴	房贴	扣税
01	1200	900	250	-35
02	1700	1200	300	-110
03	1500	1000	290	-75
04	2200	1800	400	-210
05	1200			-35

怎样保存呢？



```
int [][] wage=new int[5][4];
```

二维不等长数组

定义：

```
int [][] wage=new int[5][];
```

```
wage[0]=new int[2];
```

```
wage[1]=new int[4];
```

```
wage[2]=new int[3];
```

```
wage[3]=new int[4];
```

```
wage[4]=new int[1];
```

说明：Java的二维数组不一定是矩形可以是任意形状。

数组程序举例

1.输出 数组a

0

5

6

10

11

12

15

16

17

18

20

21

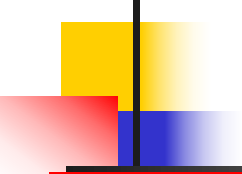
22

23

24

- **final int n=10;**
- **int [][]a=new int[n][];**
- **int i,j;**
- **for(i=0;i<a.length;i++)**
-

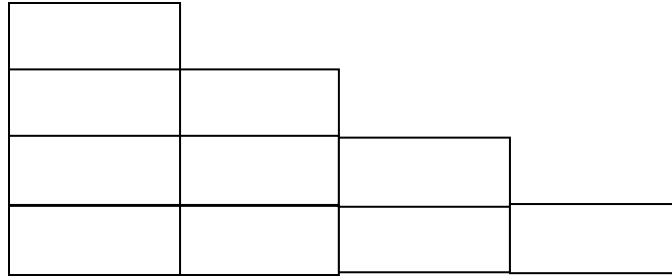
二维行数



```
■ for(i=0;i<a.length;i++){  
■   for(j=0;j<a[i].length;j++){  
■  
■       System.out.print(" "+a[i][j]);  
■   }  
■   System.out.println();  
■ }
```

每一列元素个数

- **举例：**通过下列程序建立一个如下图所示的二维数组。




```
class Student{  
    int  numbert ,age ;String name  
    Student (int number,int age,String name) {  
        this.number= number;this.age=age;  
        this.name=name;  }  
}
```

```
■ class useStudent {  
■     public static void main(String args[]) {  
■         Student [][] classTwo=new                   
■         for(int i=0;i<4;i++) {  
  
■                           
  
■     }  
  
■ }  
■ }
```




字符串处理

主要内容

-
- **1、字符Character 类**
- **2、字符串**
 - **String类**
 - **String类的常用方法**
 - **StringBuffer类；**
 - **StringBuffer类的常用方法**
- **3、字符串应用**



字符Character 类

(Java.lang.Character)

- 定义字符

- 方法1:

- ***char a;*** //Java的基本类型

- 方法2:

- ***Character b=new Character('a');***

- **Character**类的构造函数

- **public Character (char)**

Character类提供的常用方法

int compareTo(Character)

作用： 比较两个 **Character** 对象包含的字符，返回一个整数表明当前对象的值大于（**>0**）、等于（**=0**）、小于（**<0**）给定的参数的值。

注意： 参数必须为一个**Character**对象，而不能是一个**char**类型的数据 。

例：

```
Character c1=new Characer('b');  
Character c2=new Characer('y');  
int val=c1.compareTo(c2);
```

Boolean equals(Character obj)

作用： 比较两个对象所容纳的数值是否相等。

相等返回true、不相等返回false;

例： `Character c1=new Characer('a');`

`Character c2=new Characer('a');`

`boolean val=c1.equals(c2);`

String toString()

作用： 将对象转换为字符串;

例： `Character c1=new Characer('a');`

`String s=c1.toString();`//结果是字符串 “a”

char charValue()

作用： 返回**Character**对象的字符值

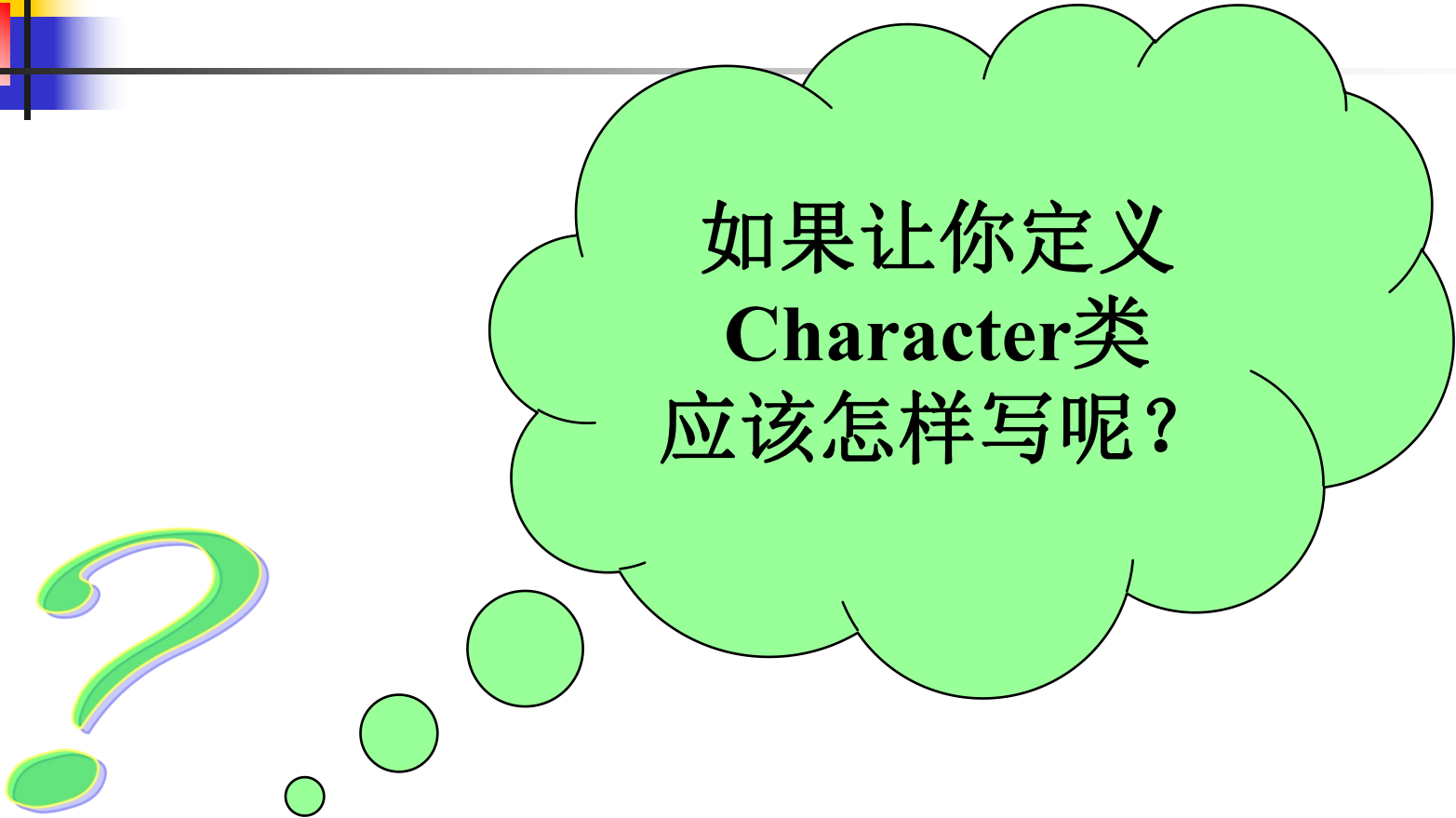
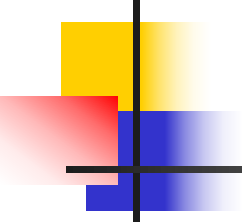
例： **Character c1=new Characer('a');**
char x=c1.charValue();

static Boolean isUpperCase(char)

作用： 判断参数字符是否是大写

注意： 参数是字符型数据，不是**Character**对象

Boolean x=Character.isUpperCase('a')



如果让你定义
Character类
应该怎样写呢？

- 例程
- 自定义一个**Character1**类(按照上课所讲的**java**提供的**Character**)形式
- **class Character1{**
- **char c;**
- **Character(char c){this.c=c;}**
- **int compareTo (Character x){**
- **}**
- **char charValue()**
- **String toString(){}
static boolean isUpperCase(char c)**
- **.....**
- **}**



String类(java.lang.String)

定义字符串:

方法1: **String str1="china";**

方法2: **char [] b1="Hello" ;**

String的构造函数

1 public String(String value)

例如: **String str1="Java";**

String str2=new String(str1);

System.out.println(str2);

2. public String(char value[])

将字符数组的内容转换为字符串，并赋予新建的对象。

例如:

char a[]={'J','a','v','a'};

String str=new String(a);

String类常用方法

1. public int length():

作用：求串长。

例如：String str="Java" ;

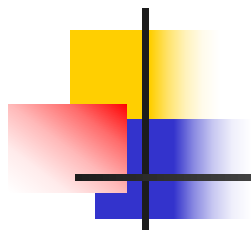
```
System.out.println(str.length( ));
```

2. public char charAt(int index)

作用：提取指定位置上的字符。

3. public int compareTo(String anotherString)

作用：对字符串内容按字典序进行大小比较。



- 编程题：
- 将字符串数组中的所有字符串按照字母顺序排序（选择排序法）

```
String str="";  
String s[ ]={"Computer","CHINA","world","U.S.A"};  
int i,j,k;  
System.out.print("排序之前: ");
```

```
for(i=0; i<s.length-1; i++) {  
    for(k=i, j=i+1; j<s.length; j++) {  
        if( s[k]. compareTo (s[j]) >0)  
            k=j;  
    }  
    str=s[i];  s[i]=s[k];  s[k]=str;  
}
```

```
System.out.print("\n排序之后: ");  
for(i=0;i<s.length;i++)  
    System.out.print("\t"+s[i]);
```

4. **char[] toCharArray()**

作用：将String对象转换到一个字符数组中。

例如：

```
String s="Hello,Java!";  
char a[ ];    a=s.toCharArray( );  
for (int i=0;i<a.length;i++)  
    System.out.print(a[i]);
```

5. **public boolean equals (String anString)**

作用：比较两个字符串对象的内容是否相等。

7. int indexOf (char ch / String str)

作用： 在字符串中搜索**字符或子串**，返回字符或子串在String对象中从左边起首次出现的位置。如果没有出现，返回-1。**例：** String s="abcdefge9e";

```
int a=s.indexOf('e'); a=4;
```

```
int b=s.indexOf("fg");b=5;
```

8 String substring (int begin, int end)

作用： 提取string对象中**从begin开始，到end-1结束的子串**，返回提取的子串。

子串长度：**end - begin**

例： String s2=s.substring(3,5); "de"



9. public String concat (**String str**)

作用： 将str对象接到调用对象的后面，返回新串。

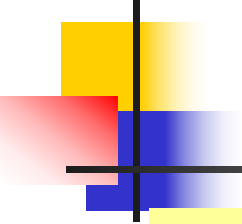
例如： String s1="Hello ", s2="Java" ;

String s3;

s3=s1.concat(s2);

System.out.println(s3);

注意： 如果参数str为空（null），则concat方法不创建新串，而仅仅返回当前串，

- 
- 编程题：//4周
 - 将以下的字符串
 - **S="8, 48, 19, 36"**
 - 转换为数组
 - **int a={8,48,19,36}**


```
class test{
```

```
    public static void main(String args[]){
```

```
        String s="8,48,19,36";
```

```
        int arr[]=new int[4];  int i=0;
```

```
        int pos=s.indexOf(",");
```

```
        while( pos!=-1){
```

```
            arr[i] =Integer.parseInt(s.substring(0,pos));
```

```
            s=s.substring(pos+1);  i++;
```

```
            pos=s.indexOf(",");
```

```
        }
```

```
        arr[i]=Integer.parseInt(s);
```

```
    } }
```

10.String replace(char oldChar, char newChar)

作用： 将String对象中所有的oldChar字符替换为newChar，返回替换后的新串。

例如：

```
String path="d:/myjava/documents";  
System.out.println(path.replace( '/' , '\\' ));
```

11.public static String valueOf(各种类型 f)

作用： 将各种数据类型转换成一个相应的字符串表示，该方法是一个**static方法**

举例

```
public class Stringtest{  
    public static void main(String[] a) throws IOException{  
        float f=3.14159f;  
        System.out.println(String.valueOf(f));  
        s1=in.readLine();           //从键盘输入  
        System.out.println(s1.length());  
        System.out.println(s1.toUpperCase());}  
        System.out.println(s1.substring(3)); //取出从3开始到最后的字符串  
        System.out.println(s1.toLowerCase());  
        System.out.println(s1.charAt(2));  
        String s2=s1.replace('a', 'o');  
        System.out.println(s1); }  
}
```

课堂练习

1下面哪些语句是创建数组的正确语句？

A `float f[][]=new float[6][6];`

B `float f[][]=new float[][6];`

C `float[]f[]= newfloat[6][6];`

D `float [][]f=new float[6][];`

■ 2阅读下列程序选出不正确的选项（ ）。

■ `int [][] arr=new int[5][];`

■ `for(int i=0;i<arr.length;i++)`

■ `arr[i]=new int [5-i]`

■ A) arr数组5行 B) `arr[0].length=5`

■ C) `arr.length=5` D) arr共有12个元素

StringBuffer类

- **String**类使用的情况：程序中不改变字符串的情况。

例：**String str1="china" str2="Great"**

String str3=str1.concat(str2);

- **StringBuffer**类对象是一个内容可以改变的字符串，修改后，仍存放在原对象。

StringBuffer str1=new StringBuffer("china");
Str1.append("Great")

- **原因：****StringBuffer**类在分配空间是预留了一定的空间，而**String**类没有



StringBuffer类对象的缓冲区

- **StringBuffer**类对象有一块缓冲区，字符串被存放在缓冲区中，缓冲区的大小可以随程序的需要进行调整。
- 缓冲区的大小称为对象的容量。

创建StringBuffer类对象

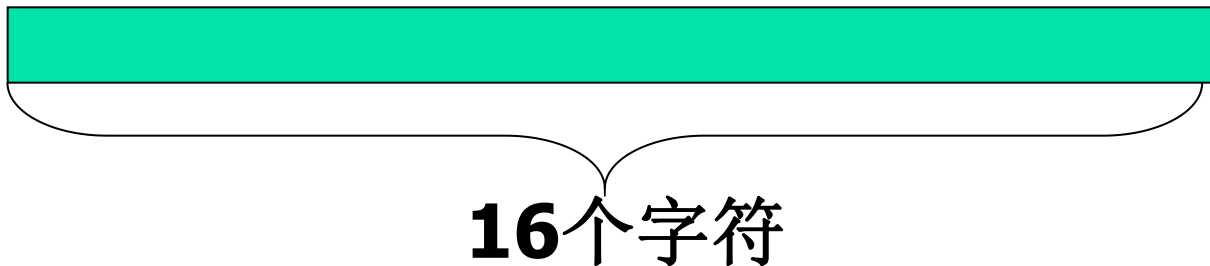
- StringBuffer类的构造函数有三种方式:

1. public StringBuffer()

作用：创建一个内容为空的StringBuffer对象，容量为16。

例如：

```
StringBuffer s=new StringBuffer();
```

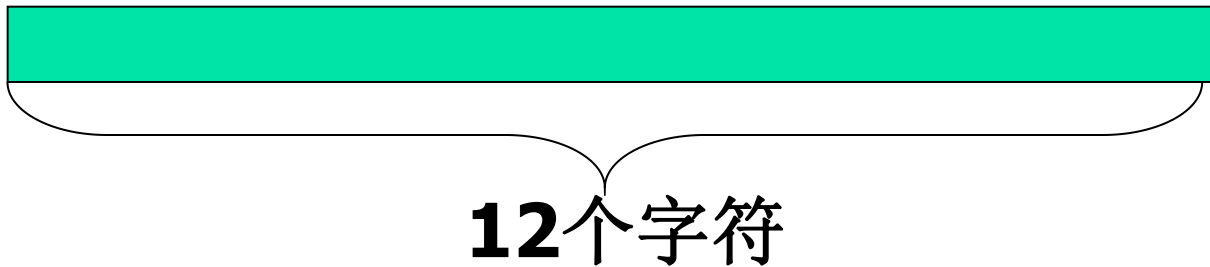


2. `public StringBuffer(int length)`

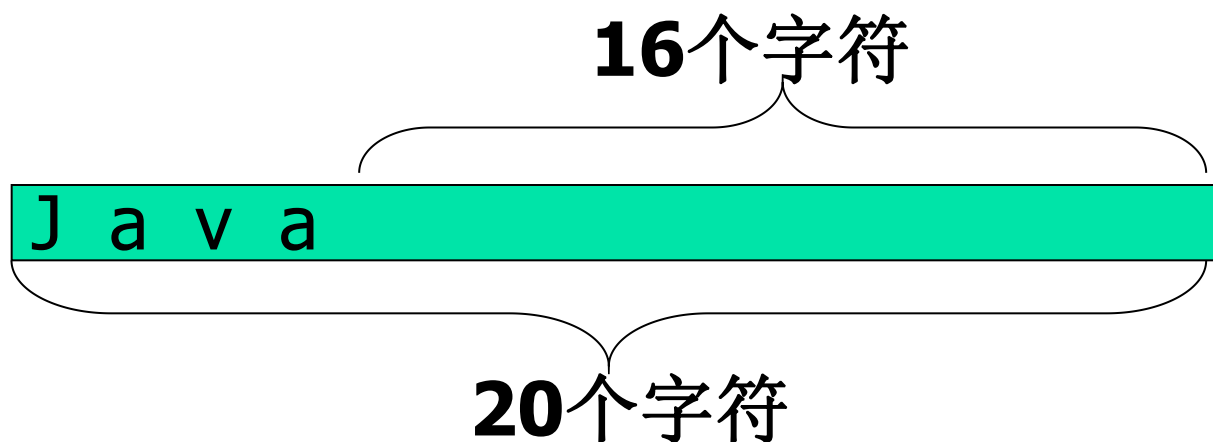
作用：内容为空,容量为length指定的大小。

例如：

```
StringBuffer s2=new StringBuffer(12);
```



- **3. public StringBuffer (String str)**
- **作用：**初始内容和参数**str**的内容相同，容量为参数**str**的长度加上**16**。
- **例如：**
- **String s1="Java";**
- **StringBuffer s2=new StringBuffer(s1);**
- **则s2的容量是20，内容是"Java"。**





StringBuffer类常用方法

1. `public int length()` 返回字符串的长度
2. `public int capacity()` 返回缓冲区大小
3. `public void setLength(int newLength)`

指定对象的长度，

如果参数小于对象的长度，则将对象截断；
如果参数大于等于对象的长度，则填充空字符('\u0000')扩充新增加的部分。



4. public void ensureCapacity (int NewCapacity)

作用： 设定对象的缓冲区的大小，若参数 小于对象的容量，则新的设置将不起作用，也就是说容量只能扩大而不能缩小。

```
public class TestCapacity{
    public static void main(String args[ ]) {
        String s="Java";
        StringBuffer str=new StringBuffer(s);
        System.out.println("Length="+str.length());//4
        System.out.println("Capacity="+str.capacity()); //20
        str.setLength(8);           // 设置长度为8
        str.ensureCapacity(80);     // 设置容量为80
        System.out.println("\nstr= "+str);
        System.out.println("Length = "+str.length( )); //8
        System.out.println ("capacity="+str.capacity()); //80
    }
}
```



5. `public void setCharAt(int index, char ch)`

作用：将参数index指定位置上的字符，设置成参数ch指定的字符。

例如：

```
StringBuffer str=new StringBuffer("Hello,Java");  
str.setCharAt(3,'L');
```

则str的内容变成了"HelLo,Java"



6. `public StringBuffer append(多种数据类型)`

作用： 将其它类型的数据添加到
`StringBuffer`对象的尾部，返回修改后的
`StringBuffer`对象。

例如：

```
StringBuffer s1,s2= new StringBuffer( );
```

```
s1=s2;    // s1和s2代表同一个对象
```

```
s2.append(3.14).append(' ').append("Java");
```

```
System.out.println(s1==s2);    // 输出true
```

```
System.out.println(s1);        // 输出3.14 Java
```

■ 7. public String toString()

作用：该方法把**StringBuffer**对象的内容复制到一个新的**String**类对象中，返回这个新的**String**类对象。

```
String s1;
```

```
StringBuffer s2=new StringBuffer("Hello ")
```

```
s1=s2.append("Java!").toString( );
```

```
System.out.println("s1="+s1+" "+s2=" "+s2);
```

程序输出结果：

s1= Hello Java! s2= Hello Java!

8. `public StringBuffer insert(int offset, 多种类型 b)`

作用：该方法是将一个其它类型的对象**b**插入到**offset**指定的位置前。

例如：

```
StringBuffer str=new StringBuffer(" Java!");  
str.insert(0,"Hello");  
System.out.println("str= "+str);
```

输出结果为： `str= Hello Java!`



应用举例

从键盘上读入一行文本，对其中的字母进行加密。加密原则是：
将每个字母向后移动**2**位，

例如：'**a**'变成'**c**'，'**X**'变成'**Z**'，
'**Y**'变成'**A**'，'**z**'变成'**b**'，非字母
不改变。

import java.io.*; //为进行输入和输出引入I/O类

public class encipher{

static char cipher (int c){

if(Character.isLetter((char)c)){//处理字母

c=c+2;

if(c>'Z' || c>'z')

c=c-26;

}

return (char)c; // 将int转换为char

}

```
public static void main(String args[])
                                throws IOException {
    BufferedReader in=new BufferedReader(
        new InputStreamReader(System.in));
    String line;
    StringBuffer buf;
    for(;;) {
        line=in.readLine();
        if( (line==null) || line.equals("quit") )
            break;
        buf=new StringBuffer( line);
        for( int i=0;i<buf.length();i++ )
            buf.setCharAt(i,cipher (buf.charAt(i)));
        System.out.println(buf);
    }
}
```



课后练习

- 从键盘上读入一行文本，识别其中单词的个数并分别输出它们。
- 算法分析：
- Flag=false