Class 7

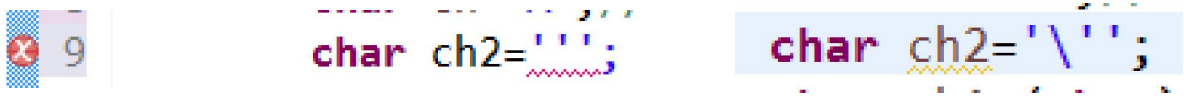
2014-11-06

String

Char用于表示单个字符

- 字符类型的声明和赋值方式
- **char ch='A';//单引号内是字符的值**

- **当字符需要赋值为单引号时，会发生错误。因此，我们用转义符号“\”来标明一下，**



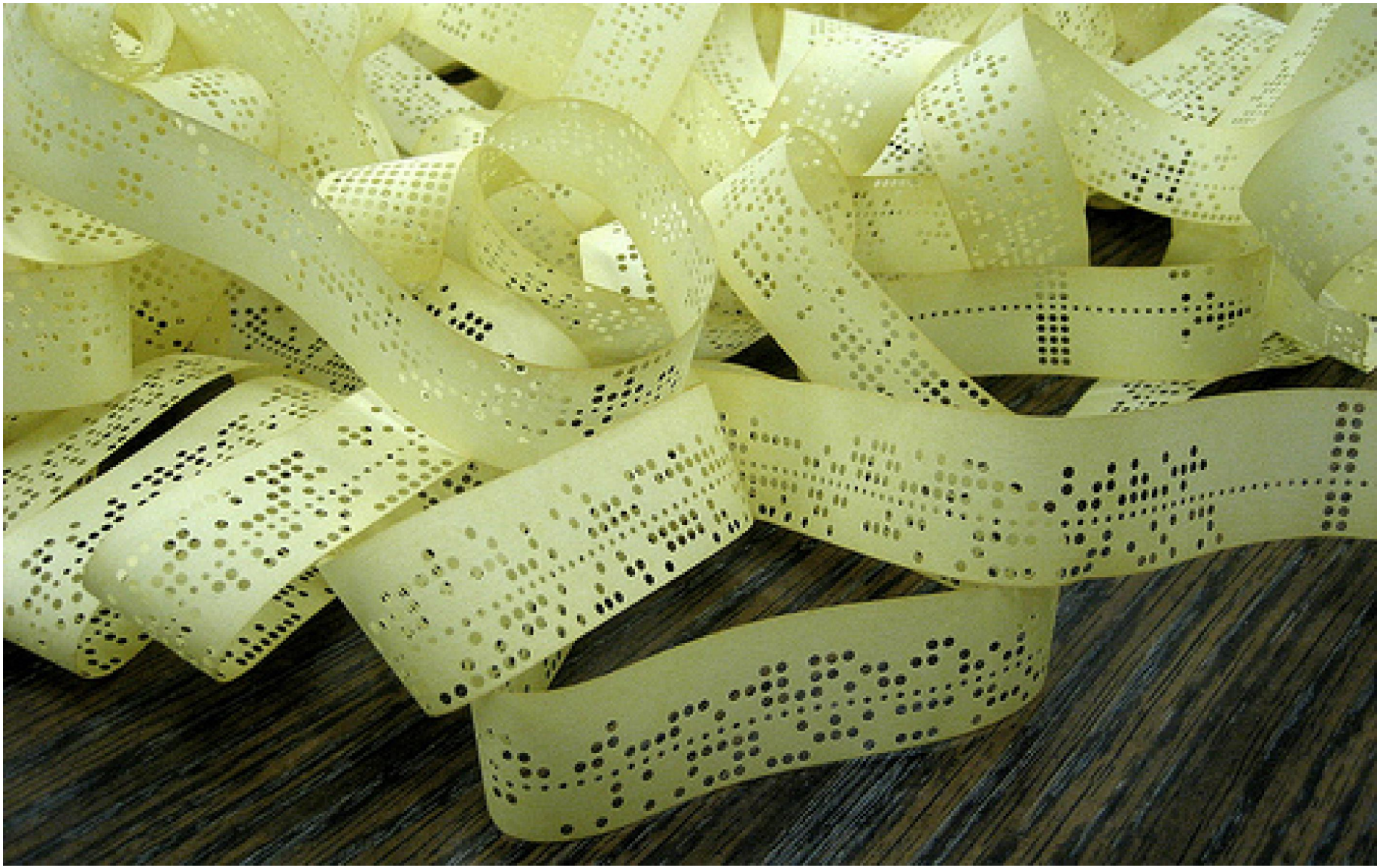
- java中有特殊含义的字符（如：换行符，回车符，单引号，双引号），如果你要用它，必须在前面加一个前缀"\"如换行（"\n"）、回车("\r")、双引号("\"")、反斜杠("\\")等。
- 以"\"符号为例，JAVA中有很多操作，例如文件操作等，需要用到路径这个东西，比如：com\mypackage\xxx.xml，这个路径一般是以字符串形式表示的，但问题来了，JAVA不知道你的\号到底是路径中的下一层的意思，还是字符串"\"的意思。所以正确的写法应该是com\\mypackage\\xxx.xml。
- 如我们熟悉的\n，换行符，如果在字符串中间直接按回车，java是不会对字符串做换行操作的，因为他不知道你想将代码分两行写还是字符串分两行显示，这时候就需要在字符串中间加上\n转义字符来将字符串换行。
- 以输出一段话为例：需要输出的话：I say:“hello world!”。双引号在java中是表示字符串的，如果不转义，则无法输出显示，所以需要用到转义符System.out.print("I say:\"hello world\"");这样就能输出双引号了。

转义序列	字符
\b	退格
\f	走纸换页
\n	换行
\r	回车
\t	横向跳格 (Ctrl-I)
\'	单引号
\"	双引号
\\	反斜杠

从一段字符串中获取一个字符

- String s=“hello”;
- char ch=s.charAt(0);

h	e	l	l	o
0	1	2	3	4



字符在计算机中是以数值的形式存储，因此字符是可以进行加减计算的。
下面是常用字符ASCII码的对应表

$D_3D_2D_1D_0$ \ $D_6D_5D_4$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	、	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

实例

```
import java.util.*;
import acm.program.*;

public class charTemplate extends ConsoleProgram{
    public void run(){
        setFont("TimesRoman-50");
        char ch='A';//单引号内是字符的值
        char ch2='\'';
        int ch1='a'-'A';
        println(ch1);
    }
}
```

问题

- 如何将小写字母转化成为大写字母呢？

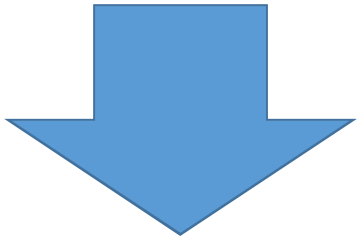
```
import acm.program.*;
public class LowerCase  extends ConsoleProgram{
public void run(){
        String s=readLine("请输入一个小写字符");
        char ch=s.charAt(0);
        if(ch >= 'a' && ch <= 'z'){
            char r=(char) (ch-32);
            println(r);
        }
        else println("输入错误");
    }
}
```

Character类

- **char ch = Character.toUpperCase(c);**

static boolean	**isDigit**(char ch) 确定指定字符是否为数字。
static boolean	**isLetter**(char ch) 确定指定字符是否为字母。
static boolean	**isLowerCase**(char ch) 确定指定字符是否为小写字母。
static boolean	**isUpperCase**(char ch) 确定指定字符是否为大写字母。
static boolean	**isLetterOrDigit**(char ch) 确定指定字符是否为字母或数字。
static char	**toLowerCase**(char ch) 使用取自 UnicodeData 文件的大小写映射信息将字符参数转换为小写。
static char	**toUpperCase**(char ch) 使用取自 UnicodeData 文件的大小写映射信息将字符参数转换为大写。

```
if(ch >= 'a' && ch <= 'z'){
   char r=(char) (ch-32);
   println(r);
}
```



```
if(Character.isLowerCase(ch)){
      println(Character.toUpperCase(ch));
}
```

凯撒密码

明文字母表	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
密文字母表	D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

```
import acm.program.*;
public class CaesarCipher extends ConsoleProgram{
        public void run(){
            setFont("TimesRoman-50");
            int n=readInt("请输入秘钥： ");
            String password=readLine("请输入密码： ");
            Coding(password,n);
        }
        private void Coding(String p,int key){
            String result="";
            for(int i=0;i<p.length();i++){
            char ch=p.charAt(i);
            char r=(char)((ch-'a'+key)%26+'a');
            result+=r;
            }
            println("加密后的密码为： "+result+" 秘钥为： "+key);
        }
}
```

String 字符串

- 字符串就是字符序列。

- String s="";//空字符串
- String h="hello";

字符串子串

- substring方法
- 从一个较大的字符串提取一个子串
  - String h="hello";
  - String h1=h.substring(a,b);
  - String h1=h.substring(0,3);
- 第二个参数，表示不想复制的第一个位置，即包含0,1,2的字符，不包含3
- 字符串的开始位置是从0开始
- h1的值为hel，长度为3

拼接，使用+号拼接两个字符串

- String h1="hello";
- String h2="world";
- String h3=h1+h2;

- 非字符串类型与字符串类型拼接时，自动转换成字符串。
  - **int age=12;**
  - String name="xiao ming";
  - String s=name+age;
- 常用语输出语句中：
- println(“hello ”+name);//注意字符串中的空格

String类型是不可变字符串

- “hello”->“help”
- 没有直接的修改方法，字符串作为一个整体，本身是不能修改的
- 可以通过子串和拼接实现：
- String s=h.substring(0,3)+"p";

检测字符串是否相等

- equal方法检测
  - s.equals(t)
  - 如果相等，返回true，不相等返回false
  - 注意：s可以使字符串变量，也可以是字符串常量。
  - 例如，s.equals(t); 或者“Times”.equals(t)都是正确的。
- 不区分大小写，检测是否相等的方法：
- "hello".equalsIgnoreCase("Hello");

- 注意，一定不能用==检测两个字符串是否相等！

空串和null串

- 空串“”是长度为0的字符串可以通过以下代码来判断是否为空串
  - **if(s.equals("")); or if(s.length()==0);**
- 空串有自己的内容（空）和长度（**0**），但**null**表示什么都没有。

- 有时需要检查一个字符串既不是空串也不是**null**
- 我们使用以下代码
  - **if(str!=null && str.length()!=0)**

String类的常用方法

StringTokenizer

- StringTokenizer类是字符串分隔解析类型。 **import java.util.*;**
- 根据自定义字符为分界符进行拆分。

构造方法包括以下几种：

- StringTokenizer（String str）：构造一个用来解析str的StringTokenizer对象。java默认的分隔符是“空格”、“制表符（‘\t’）”、“换行符(‘\n’）”、“回车符（‘\r’）”。
- StringTokenizer（String str，String delim）：构造一个用来解析str的StringTokenizer对象，并提供一个指定的分隔符。
- StringTokenizer（String str，String delim，boolean returnDelims）：构造一个用来解析str的StringTokenizer对象，并提供一个指定的分隔符，同时，指定是否返回分隔符。

主要方法：

- int countTokens（）：返回nextToken方法被调用的次数。
- boolean hasMoreTokens（）：返回是否还有分隔符。
- boolean hasMoreElements（）：返回是否还有分隔符。
- String nextToken（）：返回从当前位置到下一个分隔符的字符串。
- Object nextElement（）：返回从当前位置到下一个分隔符的字符串。
- String nextToken（String delim）：与4类似，以指定的分隔符返回结果。

例：

```
import java.util.*;
import acm.program.ConsoleProgram;
public class StringTokenizerTemplate extends ConsoleProgram{
    public void run(){
        String s=new String("good morning everyone");
        StringTokenizer st=new StringTokenizer(s);
        println("Token Total:"+st.countTokens());
        while ( st.hasMoreElements() ){
                println(st.nextToken());
        }

        String s2=new String("good=morning=everyone! Nice to see=you");
        StringTokenizer st2=new StringTokenizer(s2,"=",true);
        println("Token Total:"+st2.countTokens());
        while ( st2.hasMoreElements() ){
                println(st2.nextToken());
        }
    }
}
```

homework

- 英语姓名的一般结构为：教名 自取名 姓。如 William Jefferson Clinton。但在很多场合中间名往往略去不写，如 William Clinton。
- 在期刊论文的参考文献中，一般将英文作者名进行缩写，如 Kate Smith Stafilidis可写为Stafilidis KS

- 输入英文名，自动去掉中间名，并输出显示。
- 输入英文名，自动改为参考文献格式。

See you next week!