数组下标从 0 开始。
示例 1：讲 seq 生成的数字序列循环放到数组里面

```
#!/bin/bash
for i in $(seq 1 10); do
      array[a]=$i
      let a++
done
echo ${array[*]}
# bash test.sh
1 2 3 4 5 6 7 8 9 10
```

示例 2：遍历数组元素

```
方法 1：
#!/bin/bash
IP=(192.168.1.1 192.168.1.2 192.168.1.3)
for ((i=0;i<${#IP[*]};i++)); do
      echo ${IP[$i]}
done
# bash test.sh
192.168.1.1
192.168.1.2
192.168.1.3
方法 2：
#!/bin/bash
IP=(192.168.1.1 192.168.1.2 192.168.1.3)
for   IP   in ${IP[*]}; do
        echo $IP
done
```

第六章　Shell 正则表达式

正则表达式在每种语言中都会有，功能就是匹配符合你预期要求的字符串。
Shell 正则表达式分为两种：
基础正则表达式：BRE（basic regular express）
扩展正则表达式：ERE（extend regular express），扩展的表达式有+、?、|和()
下面是一些常用的正则表达式符号，我们先拿 grep 工具举例说明。

符号	描述	示例
.	匹配除换行符(\n)之外的任意单个字符	匹配 123： echo -e "123\n456" |grep '1.3'
^	匹配前面字符串开头	匹配以 abc 开头的行： echo -e "abc\nxyz" |grep ^abc
$	匹配前面字符串结尾	匹配以 xyz 结尾的行：

		echo -e "abc\nxyz" |grep xyz$
*	匹配前一个字符零个或多个	匹配 x、xo 和 xoo： echo -e "x\nxo\nxoo\no\noo" |grep "xo*" x 是必须的，批量了 0 零个或多个
+	匹配前面字符 1 个或多个	匹配 abc 和 abcc： echo -e "abc\nabcc\nadd" |grep -E 'ab+' 匹配单个数字：echo "113" |grep -o '[0-9]' 连续匹配多个数字：echo "113" |grep -E -o '[0-9]+'
?	匹配前面字符 0 个或 1 个	匹配 ac 或 abc： echo -e "ac\nabc\nadd" |grep -E 'a?c'
[ ]	匹配中括号之中的任意一个字符	匹配 a 或 c： echo -e "a\nb\nc" |grep '[ac]'
[.-.]	匹配中括号中范围内的任意一个字符	匹配所有字母： echo -e "a\nb\nc" |grep '[a-z]'
[^]	匹配[^字符]之外的任意一个字符	匹配 a 或 b： echo -e "a\nb\nc" |grep '[^c-z]' 匹配末尾数字：echo "abc:cde;123" |grep -E '[^;]+$'
^[^]	匹配不是中括号内任意一个字符开头的行	匹配不是#开头的行： grep '^[^#]' /etc/httpd/conf/httpd.conf
{n}或{n,}	匹配花括号前面字符至少 n 个字符	匹配 abc 字符串（至少三个字符以上字符串）： echo -e "a\nabc\nc" |grep -E '[a-z]{3}'
{n,m}	匹配花括号前面字符至少 n 个字符，最多 m 个字符	匹配 12 和 123（不加边界符会匹配单个字符）： echo -e "1\n12\n123\n1234" |grep -E -w -o '[0-9]{2,3}'
\<	边界符，匹配字符串开始	匹配开始是 123 和 1234： echo -e "1\n12\n123\n1234" |grep '\<123'
\>	边界符，匹配字符串结束	匹配结束是 1234： echo -e "1\n12\n123\n1234" |grep '4\>'
( )	单元或组合：将小括号里面作为一个组合 分组：匹配小括号中正则表达式或字符。\n 反向引用，n 是数字，从 1 开始编	单元：匹配 123a 字符串 echo "123abc" |grep -E -o '([0-9a-z]){4}' 分组：匹配 11 echo "113abc" |grep -E -o '(1)\1' 匹配出现 xo 出现零次或多次： echo -e "x\nxo\nxoo\no\noo" |egrep "(xo)*"

	号，表示引用第 n 个分组匹配的内容	
|	匹配竖杠两边的任意一个	匹配 12 和 123： echo -e "1\n12\n123\n1234" |grep -E '12\>|123\>'
\	转义符，将特殊符号转成原有意义	1.2，匹配 1.2：1\.2，否则 112 也会匹配到

Posix 字符	描述
[:alnum:]	等效[a-zA-Z0-9]
[:alpha:]	等效[a-zA-Z]
[:lower:]	等效[a-z]
[:upper:]	等效[A-Z]
[:digit:]	等效[0-9]
[:space:]	匹配任意空白字符，等效[\t\n\r\f\v]
[:graph:]	非空白字符
[:blank:]	空格与定位字符
[:cntrl:]	控制字符
[:print:]	可显示的字符
[:punct:]	标点符号字符
[:xdigit:]	十六进制

示例：

```
echo -e "1\n12\n123\n1234a" |grep '[[:digit:]]'
```

在 Shell 下使用这些正则表达式处理文本最多的命令有下面几个工具：

命令	描述
grep	默认不支持扩展表达式，加-E 选项开启 ERE。如果不加-E 使用花括号要加转义符\{\}

egrep	支持基础和扩展表达式
awk	支持 egrep 所有的正则表达式
sed	默认不支持扩展表达式，加-r 选项开启 ERE。如果不加-r 使用花括号要加转义符\{\}

支持的特殊字符	描述
\w	匹配任意数字和字母，等效[a-zA-Z0-9_]
\W	与\w 相反，等效[^a-zA-Z0-9_]
\b	匹配字符串开始或结束，等效\<和\>
\s	匹配任意的空白字符
\S	匹配非空白字符

空白符	描述
\n	换行符
\r	回车符
\t	水平制表符
\v	垂直制表符
\0	空值符
\b	退后一格

第七章　Shell 文本处理三剑客

7.1 grep

过滤来自一个文件或标准输入匹配模式内容。
除了 grep 外，还有 egrep、fgrep。egrep 是 grep 的扩展，相当于 grep -E。fgrep 相当于 grep -f，用的少。
Usage: grep [OPTION]... PATTERN [FILE]...

支持的正则	描述