基本输出

1：echo 命令

echo是Shell的一个内部指令，用于在屏幕上打印出字符串，每条echo语句会自动换行。命令格式：

```
1. echo arg
```

显示转义字符

```
1. echo "\"It is a test\""
```

结果将是：

"It is a test"

双引号也可以省略。

显示变量

```
1. name="OK"
2. echo "$name It is a test"
```

结果将是：

OK It is a test

同样双引号也可以省略。

如果变量与其它字符相连的话，需要使用大括号（{ }）：

```
1. mouth=8
2. echo "${mouth}-1-2009"
```

结果将是：

8-1-2009

显示换行

```
1. echo "OK!\n"
2. echo "It is a test"
```

输出：

OK!

```
It is a test
```

原样输出字符串

若需要原样输出字符串（不进行转义），请使用单引号。例如：

```
1. echo '$name\"'
```

显示命令执行结果

```
1. echo `date`
```

结果将显示当前日期

2：printf 命令

printf 命令用于格式化输出， 是echo命令的增强版。它是C语言printf()库函数的一个有限的变形，并且在语法上有些不同。

注意：printf 由 POSIX 标准所定义，移植性要比 echo 好。

printf 不像 echo 那样会自动换行，必须显式添加换行符(\n)。

printf 命令的语法：

```
printf  format-string  [arguments...]
```

format-string 为格式控制字符串，arguments 为参数列表。

printf()在C语言入门教程中已经讲到，功能和用法与 printf 命令类似

这里仅说明与C语言printf()函数的不同：

- printf 命令不用加括号
- format-string 可以没有引号，但最好加上，单引号双引号均可。
- 参数多于格式控制符(%)时，format-string 可以重用，可以将所有参数都转换。
- arguments 使用空格分隔，不用逗号。

请看下面的例子：

```
1. # format-string为双引号
2. $ printf "%d %s\n" 1 "abc"
3. 1 abc
4. # 单引号与双引号效果一样
5. $ printf '%d %s\n' 1 "abc"
6. 1 abc
```

```
# 没有引号也可以输出
$ printf %s abcdef
abcdef
# 格式只指定了一个参数，但多出的参数仍然会按照该格式输出，format-string 被重用
$ printf %s abc def
abcdef
$ printf "%s\n" abc def
abc
def
$ printf "%s %s %s\n" a b c d e f g h i j
a b c
d e f
g h i
j
# 如果没有 arguments，那么 %s 用NULL代替，%d 用 0 代替
$ printf "%s and %d \n"
and 0
# 如果以 %d 的格式来显示字符串，那么会有警告，提示无效的数字，此时默认置为 0
$ printf "The first program always prints'%s,%d\n'" Hello Shell
-bash: printf: Shell: invalid number
The first program always prints 'Hello,0'
$
```

注意，根据POSIX标准，浮点格式%e、%E、%f、%g与%G是“不需要被支持”。这是因为awk支持浮点预算，且有它自己的printf语句。这样Shell程序中需要将浮点数值进行格式化的打印时，可使用小型的awk程序实现。然而，内建于bash、ksh93和zsh中的printf命令都支持浮点格式。