

1. 字符串型

用于表示一串字符

两种风格：

- c 风格字符串： `char 变量名[] = “字符串值”;`
- c++ 风格字符串： `string 变量名 = “字符串值”;`

但是注意， c 风格字符串变量名后一定要加中括号，c++ 风格字符串一定要添加 `<string>` 头文件。且两者必须都用双引号。

The image shows a C++ IDE window titled '4.cpp' with the following code:

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4
5
6 using namespace std;
7
8 int main()
9 {
10     char str1[] = "hello_world";
11     cout << str1 << endl;
12
13     string str2 = "hello_world";
14     cout << str2 << endl;
15
16
17
18
19
20
21
22
23
24
25
26     system("pause");
27     return 0;
28 }
```

Annotations in the image:

- An arrow points to line 2 (`#include <string>`) with the text: "c++ 风格，需要头文件 string".
- An arrow points to line 10 (`char str1[] = "hello_world";`) with the text: "C语言风格，不要忘记 []".

Below the code editor is a terminal window showing the output of the program:

```
C:\Users\19504\Desktop\mygithub\c++学习笔记 (3) \4\Debug\4.exe
hello_world
hello_world
请按任意键继续. . .
```

2. 布尔类型 bool

作用：布尔数据类型代表真或假的值

bool 类型只有两个值：

- true --- 真（本质是1）
- false --- 假（本质是0）

bool 类型占1个字节大小

```
bool flag = true; //此时给flag指定为true, 实际flag为1
cout << flag << endl;

flag = false; //此时给flag重新赋值
cout << flag << endl;

cout << "The size of bool = " << sizeof(bool) << endl; //bool占用空间大小为 1
```

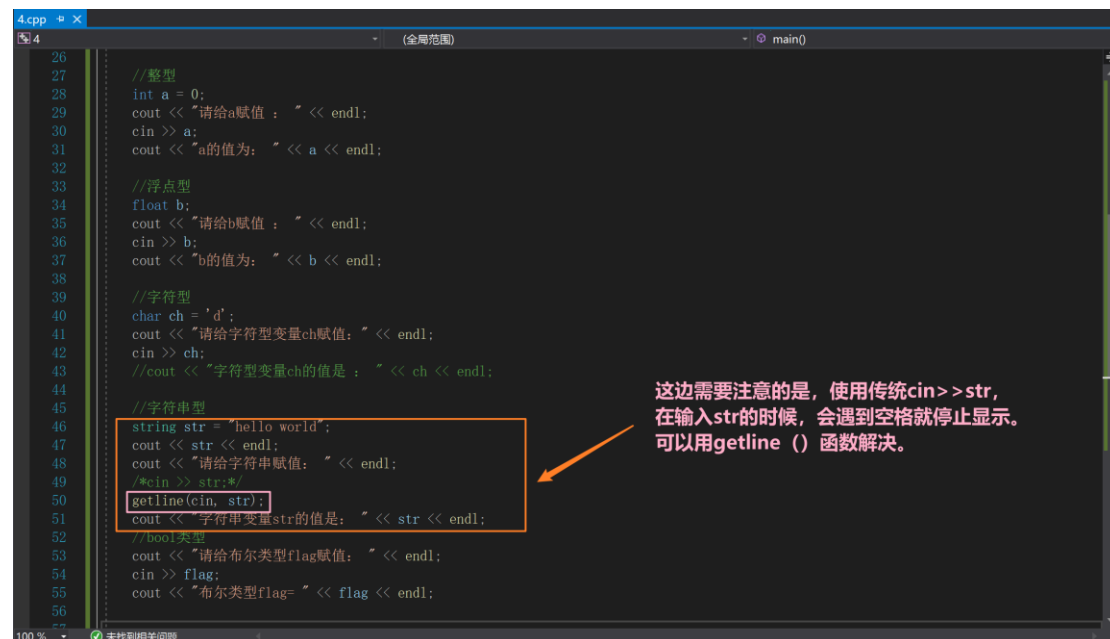
```
1
0
The size of bool = 1
请按任意键继续. . .
```

3. 数据的输入

用于从键盘中获取数据

关键字：cin

语法：cin >> 变量



```
4.cpp x
26 //整型
27 int a = 0;
28 cout << "请给a赋值： " << endl;
29 cin >> a;
30 cout << "a的值为： " << a << endl;
31
32 //浮点型
33 float b;
34 cout << "请给b赋值： " << endl;
35 cin >> b;
36 cout << "b的值为： " << b << endl;
37
38 //字符型
39 char ch = 'd';
40 cout << "请给字符型变量ch赋值： " << endl;
41 cin >> ch;
42 //cout << "字符型变量ch的值是： " << ch << endl;
43
44 //字符串型
45 string str = "hello world";
46 cout << str << endl;
47 cout << "请给字符串赋值： " << endl;
48 /*cin >> str;*/
49 getline(cin, str);
50 cout << "字符串变量str的值是： " << str << endl;
51 //bool类型
52 cout << "请给布尔类型flag赋值： " << endl;
53 cin >> flag;
54 cout << "布尔类型flag= " << flag << endl;
55
56
```

这边需要注意的是，使用传统cin>>str, 在输入str的时候，会遇到空格就停止显示。可以用getline () 函数解决。

```
请给a赋值：
5
a的值为： 5
请给b赋值：
3.56
b的值为： 3.56
请给字符型变量ch赋值：
t
hello world
请给字符串赋值：
字符串变量str的值是：
请给布尔类型flag赋值：
i love you
布尔类型flag= 0
请按任意键继续. . .
```

4. 运算符

用于执行代码的运算

主要运算符种类:

- 算术运算符: 用于处理四则运算
- 赋值运算符: 用于将表达式的值赋给变量
- 比较运算符: 用于表达式比较, 并且返回一个真值或假值
- 逻辑运算符: 用于根据表达式的值返回真值或假值

算术运算符

+ - * / : 算术运算的结果与变量类型有关

两个整数相除, 结果还是整数形式, 即只取整数部分

两数相除, 除数不能为 0;

```
int r = 10;  
int u = 4;
```

```
cout << r + u << endl;  
cout << r - u << endl;  
cout << r * u << endl;  
cout << r / u << endl;
```

这边比较注意的就是结果的形式, 若原来变量都是整型, 那么结果也是整型。

```
14  
6  
40  
2  
请按任意键继续. . .
```

取模运算: % 本质就是求余数 但是两个小数之间不能做取模运算。

```
10 int t = 10;  
11 int r = 3;
```

```
13 cout << t % r << endl;
```

```
15 C:\Users\19504\Desktop\mygithub\c++学习笔记 (3) \4\Debug\4.exe
```

```
16 1  
17 请按任意键继续. . .  
18
```

递增递减运算符: ++ --, 但是分成前置和后置运算

区别:

- 前置递增 先让变量+1 然后进行表达式运算
- 后置递增 先进行表达式运算 后让变量递增

```
5.cpp* 4.cpp
4 (全局范围) main()
7 int main()
8 {
9
10 int t = 10;
11 int r = 3;
12 cout << t % r << endl;
13
14 int a1 = 10;
15 a1++;
16 cout << a1 << endl;
17
18 int b1 = 10;
19 ++b1;
20 cout << b1 << endl;
21
22 int a2 = 10;
23 int b2;
24 b2 = a2++ * 10;
25 cout << "a2=: " << a2 << endl;
26 cout << "b2= " << b2 << endl;
27
28 int a3 = 10;
29 int b3;
30 b3 = ++a3 * 10;
31 cout << "a3=: " << a3 << endl;
32 cout << "b3= " << b3 << endl;
33
34 system("pause");
35 return 0;
36
37
```

这边要注意前置和后置的区别

前置: 先改变变量, 然后进行运算
后置: 先运算, 然后变量改变

记忆方法: 前置后置从名字上, 只要记住前后是对于变量而言。前置, 先改变变量; 后置, 后改变变量;

```
1
11
11
a2=:11
b2= 100
a3=:11
b3= 110
请按任意键继续. . .
```

赋值运算符:

运算符	术语	示例	结果
=	赋值	a=2; b=3;	a=2; b=3;
+=	加等于	a=0; a+=2;	a=2;
-=	减等于	a=5; a-=3;	a=2;
=	乘等于	a=2; a=2;	a=4;
/=	除等于	a=4; a/=2;	a=2;
%=	模等于	a=3; a%=2;	a=1;

```

int a4 = 6;
cout << a4 << endl;

a4 += 2;
cout << a4 << endl;

a4 -= 5;
cout << a4 << endl;

a4 *= 10;
cout << a4 << endl;

a4 /= 5;
cout << a4 << endl;

a4 %= 2;
cout << a4 << endl;

```

```

6
8
3
30
6
0
请按任意键继续. . .

```

比较运算符：真输出 1，假输出 0

运算符	术语	示例	结果
==	相等	4 == 3	0
!=	不等	4 != 3	1
<	小于	4 < 3	0
>	大于	4 > 3	1
<=	小于等于	4 <= 3	0
>=	大于等于	4 >= 1	1

逻辑运算符：与/或/非 -> &&/ || / !

（在 c++ 中，不为 0 都是真）

非: !，类似于取反操作

与: a && b 如果两者都为真，则结果为真 (1)， 否则为假 (0)

或: a || b 有一个为真则为真；只有当两者都是假时为假。

