

## 1. 标识符命名规则

标识符在命名的时候尽量**见名知意**

- 标识符不能是关键字
- 标识符只能由字母、数字、下划线组成
- 第一个字符必须为字母或者下划线
- 标识符中字母区分大小写

## 2. 数据类型

在指定变量的时候必须指定出想应的数据类型

数据类型意义是给变量分配合适的内存空间

区别在于所占用的内存空间不同

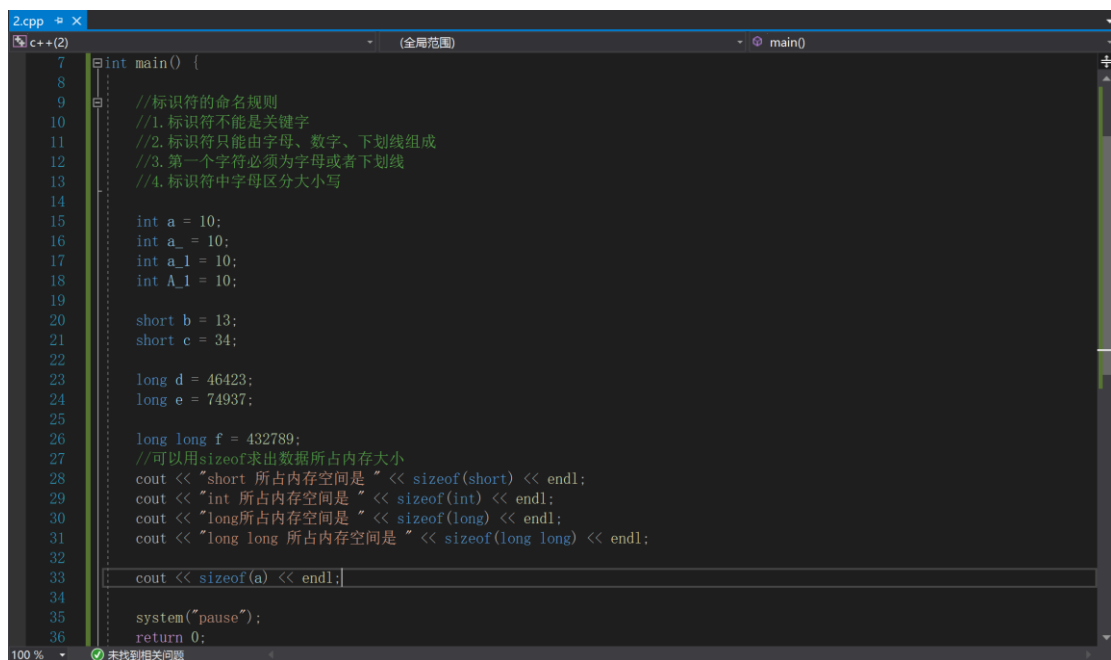
**整型：**

1. short  $(-2^{15}-2^{15}-1)$  2 字节
2. int  $(-2^{31}-2^{31}-1)$  4 字节
3. long  $(-2^{31}-2^{31}-1)$  与操作系统有关， windows 下与 int 相同
4. long long  $(-2^{63}-2^{63}-1)$  8 字节

## 3. sizeof 关键字

利用 sizeof 关键字能够统计数据类型所占内存大小

语法 sizeof (数据类型/变量)



```
2.cpp x
c++(2) (全局范围) main()
7 int main() {
8
9     //标识符的命名规则
10    //1. 标识符不能是关键字
11    //2. 标识符只能由字母、数字、下划线组成
12    //3. 第一个字符必须为字母或者下划线
13    //4. 标识符中字母区分大小写
14
15    int a = 10;
16    int a_ = 10;
17    int a_1 = 10;
18    int A_1 = 10;
19
20    short b = 13;
21    short c = 34;
22
23    long d = 46423;
24    long e = 74937;
25
26    long long f = 432789;
27    //可以用sizeof求出数据所占内存大小
28    cout << "short 所占内存空间是 " << sizeof(short) << endl;
29    cout << "int 所占内存空间是 " << sizeof(int) << endl;
30    cout << "long所占内存空间是 " << sizeof(long) << endl;
31    cout << "long long 所占内存空间是 " << sizeof(long long) << endl;
32
33    cout << sizeof(a) << endl;
34
35    system("pause");
36    return 0;
100 % 未找到相关问题
```

```
Microsoft Visual Studio 调试控制台
short 所占内存空间是 2
int 所占内存空间是 4
long所占内存空间是 4
long long 所占内存空间是 8
请按任意键继续. . .

C:\Users\19504\Desktop\C++\c++(2)\Debug\c++(2).exe (进程 25828) 已退出，返回代码为：0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

#### 4. 实型（浮点型）

用于表示小数

浮点型变量分成两种：

1. 单精度 float ： 4 字节 7 位有效数字
2. 双精度 double ： 8 字节 15-16 位有效数字

双精度的精确有效数字更多， c++中的有效数字需要加上小数点前的位数

**但是目前 c++中正常显示最多是 6 位有效数字。且按照四舍五入的方式。若最后一位是 0，则会自动不显示。无论数据类型是 float 还是 double。**

**如果需要多显示有效数字，需要另外进行配置。**

```
3.cpp * 2.cpp
c++(2) (全局范围) main()
1 #include <iostream>
2 #include <string>
3 #include <vector>
4
5 using namespace std;
6
7 int main() {
8     float f1 = 3.14f;
9     cout << "f1= " << f1 << endl;
10
11     double d1 = 3.14;
12     cout << "d1= " << d1 << endl;
13
14
15
16
17 //但是目前c++中正常显示最多是6位有效数字。无论数据类型是float还是double。
18 //如果需要多显示有效数字，需要另外进行配置。
19
20 /*float f1 = 3.1415926f;
21 cout << "f1= " << f1 << endl;
22
23 double d1 = 3.141592657;
24 cout << "d1= " << d1 << endl;
25 system("pause");*/
26 return 0;
27 }
```

3.14后的f, 建议加上, 因为不加上, 系统会默认将小数变成double型, 然后在加float进行转化。加上f可以让系统认为就是float型

```
Microsoft Visual Studio 调试控制台
f1= 3.14
d1= 3.14

C:\Users\19504\Desktop\C++\c++(2)\Debug\c++(2).exe (进程 25424) 已退出，返回代码为：0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

//显示有效数字的区别上，具体显示多位数的方式见程序

```
3.cpp 2.cpp
c++(2) (全局范围) main()
1 #include <iostream>
2 #include <string>
3 #include <vector>
4
5 using namespace std;
6
7 int main() {
8
9     /*float f1 = 3.14f;
10    cout << "f1= " << f1 << endl;
11    ...
12    double d1 = 3.14;
13    cout << "d1= " << d1 << endl;*/
14
15
16
17    //但是目前c++中正常显示最多是6位有效数字，无论数据类型是float还是double。
18    //如果需要多显示有效数字，需要另外进行配置。
19
20    float f1 = 3.1415926f;
21    cout << "f1= " << f1 << endl;
22
23    double d1 = 3.141592657;
24    cout << "d1= " << d1 << endl;
25    system("pause");
26    return 0;
27 }
```

在显示有效数字上，正常情况下都最多只能显示6位

```
C:\Users\19504\Desktop\C++\c++(2)\Debug\c++(2).exe
f1= 3.14159
d1= 3.14159
请按任意键继续. . .
```

都是6位

## 5. 字符型

字符型变量用于显示单个字符

语法： `char ch = 'a';`

注意点：

1. 在显示字符变量时，用单引号将字符括起来，不要用双引号。‘
2. 单引号内只能是一个字符，而不能是字符串。

字符型变量只占用一个字节

字符型变量在实际存储的时候并不是直接存储字符，而是将对应的 ASCII 编码存入存储单元

```
//字符型变量
//一些简单的ASCII码对应
//a - 97
//A - 65
char ch = 'a';
cout << ch << endl;
cout << "占用的内存空间：" << sizeof(char) << endl;
cout << (int)ch << endl;
```

```
a
占用的内存空间： 1
97
请按任意键继续. . .
```

## 6. 转义字符

用于表示一些不能直接被显示的 ASCII 字母

用的较多的： `\n`, `\\`, `\t`

转义字符	含义	ASCII码（16/10进制）
<code>\0</code>	空字符(NULL)	00H/0
<code>\n</code>	换行符(LF)	0AH/10
<code>\r</code>	回车符(CR)	0DH/13
<code>\t</code>	水平制表符(HT)	09H/9
<code>\v</code>	垂直制表符(VT)	0B/11
<code>\a</code>	响铃(BEL)	07/7
<code>\b</code>	退格符(BS)	08H/8
<code>\f</code>	换页符(FF)	0CH/12
<code>\'</code>	单引号	27H/39
<code>\"</code>	双引号	22H/34
<code>\\</code>	反斜杠	5CH/92
<code>\?</code>	问号字符	3F/63
<code>\ddd</code>	任意字符	三位八进制
<code>\xhh</code>	任意字符	二位十六进制

```
//转义字符  
  
// 换行符\n  
  
cout << "Hello world!\n";
```

```
// 反斜杠 \\  
  
cout << "\\n";
```

```
// 水平制表符 \t  
//这个用处是能够对其  
//一般水平制表符是8个字符
```

```
cout << "aaa\ttheworld" << endl;  
cout << "aaaaa\ttheworld" << endl;  
cout << "aaaaaa\ttheworld" << endl;  
  
cout << "aaaaaaaa\theoolworld " << endl;
```

```
system("pause");  
return 0;
```

水平制表符是八个字符对齐，超过会继续增加8个字符的空格

```
Hello world!  
\  
aaa    helloworld  
aaaaa  helloworld  
aaaaaa helloworld  
aaaaaaaa  heoolworld  
请按任意键继续. . .
```