数据类型转换

C++语言中，不同数据类型的运算对象进行混合运算，或者需要将一个表达式的结果转换成期望的类型时，就需要依据数据类型转换规则进行转换。

混合运算时的数据类型转换

整型、实型、字符型数据间可以混合运算。在这种情况下，系统自动将不一致的数据类型转换成一致的数据类型，然后进行运算。为了保证运算精度，系统在运算时的转换规则是将存储长度较短的运算对象转成存储长度较长的类型，然后再进行处理。

具体规则如下:

- 所有的浮点运算都是以双精度进行的，即使仅含 float 单精度量运算的表达式，也要先转换成 double 型，再作运算。
- char 型和 short 型参与运算时，必须先转换成 int 型。
- 若两种类型的字节数相同，且一种有符号，一种无符号，则转换成无符号类型。

赋值时的数据类型转换

在进行赋值运算时，如果赋值运算符两边的数据类型不同，系统将会自动进行类型转换，即将赋值运算符右边的数据类型自动转换成左边的变量类型。

观察以下程序的运行结果：

```
1.  #include<iostream>
2.  using namespace std;
3.  int main()
4.  {
5.      float f1 = 97.5;
6.      float f2 = 97;
7.      float f3 = 'a';
8.      int i1 = 98;
9.      int i2 = 98.6;
10.     int i3 = 'b';
11.     char c1 = 99;
12.     char c2 = 99.1;
13.     char c3 = 'c';
14.     cout << "f1= " << f1 << endl;
15.     cout << "f2= " << f2 << endl;
16.     cout << "f3= " << f3 << endl;
17.     cout << endl;
18.     cout << "i1= " << i1 << endl;
19.     cout << "i2= " << i2 << endl;
20.     cout << "i3= " << i3 << endl;
21.     cout << endl;
22.     cout << "c1= " << c1 << endl;
23.     cout << "c2= " << c2 << endl;
24.     cout << "c3= " << c3 << endl;
25.     cout << endl;
26.     return 0;
27. }
```

【输出结果】

```
f1= 97.5
f2= 97
f3= 97
i1= 98
i2= 98
i3= 98
c1= c
c2= c
```

c3= c

赋值时的类型转换规则总结如下：

类型	转换方式	可能问题
表示数值范围较小的赋值给表示数值范围较大的	整数：char -> short -> int -> long 无符号整数：bool -> unsigned char -> unsigned short -> unsigned int -> unsigned long 浮点数：float -> double 传送数值的二进制表示形式。不会出现转换问题。	无问题
表示数值范围较大的赋值给表示数值范围较小的	整数：char <- short <- int <- long 无符号整数：bool <- unsigned char <- unsigned short <- unsigned int <- unsigned long 浮点数：float <- double 传送数值的二进制表示形式。超出的数据位被截断。	溢出
浮点数类型赋值给整数类型	浮点数截断小数部分，得到整数。然后根据整数类型赋值规则进行处理。 例如，“int i=4.6;”，根据转换原则，直接舍弃小数，所以 i 的值为 4。	小数截断；溢出
整数类型赋值给浮点数类型	根据相应的浮点类型的精度，整数转换为浮点数表示。然后根据浮点数类型赋值规则进行处理。 例如：“float f=1111111111;”，float 为 7 位有效数字，所以 f 的值为 1.1111111E9。	精度降低
无符号类型及有符号类型的互转	直接传送数值的二进制表示形式，超出的数据位被截断。	溢出

表1. 赋值时的类型转换规则

强制类型转换

在 C++语言中，还允许强制类型转换。

强制类型转换就是将某一数据的数据类型转换为指定的另一种数据类型。

强制类型转换只是临时转换。

强制类型转换运算符组成的运算表达式的一般形式为：

```
1. (类型名)(表达式);
```

比如(int)(3.5)的值就是 3。

大小写字母的转换

字符类型（char）的数据也可以做加减运算。

小写字母的 ASCII 值、大写字母的 ASCII 值分别是连续的。

【参考程序】

```
#include<iostream>
using namespace std;
int main()
{
    char c1 = 'z';
    char c2 = 'Z';
    cout << c1 << "   " << c2 << endl;
    //小写字母转换大写字母
    c1 = c1 - 'a' + 'A';
    //大写字母转换小写字母
    c2 = c2 - 'A' + 'a';
    cout << c1 << "   " << c2 << endl;
    return 0;
}
```

【运行结果】

```
z   Z
Z   z
```

平均分

【问题描述】

已知某班有男同学x位，女同学y位，男生的平均分是 87 分，女生的平均分是 85 分。那么全体同学的平均分是多少分？

【算法分析】

全体同学的平均分：

$$\frac{x \times 87 + y \times 85}{x + y}$$

【参考程序 1】

```
#include<iostream>
using namespace std;
int main()
{
    int x, y;
    cin >> x >> y;
    //数据类型强制转换，按实数格式输出
    cout << (double)(x*87+y*85)/(x+y) << endl;
    return 0;
}
```

【参考程序 2】

```
#include<iostream>
using namespace std;
int main()
{
    int x, y;
    cin >> x >> y;
    //乘上 1.0，整数变实数
    cout << 1.0*(x*87+y*85)/(x+y) << endl;
    return 0;
}
```

求小数

【问题描述】

输入两个正整数，输出这两个数相除之后的小数部分。

【输入格式】

一行，包含两个整数，依次为被除数和除数（除数非零），中间用一个空格隔开。

【输出格式】

一行，一个浮点数，保留到小数点后 9 位。

【输入样例】

```
14 6
```

【输出样例】

```
0.333333333
```

【参考代码】

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
    int a, b;
    cin >> a >> b;
    double d = 1.0 * a / b - a / b;
    cout << fixed << setprecision(9) << d << endl;
    return 0;
}
```