<< 左位移就是结果乘以2

>> 右位移就是结果除以2

|| 逻辑或（只要有一个为非零拿结果就是真）（真 1， 假 0）

| 位或，二进制按位进行逻辑或运算

A? B : C (A真为B（不执行C），A假为C（不执行B）；真前假后)只执行一个

A++{ // 返回变化前的

B=A;

A=A+1;

Return B;

}

++A{ // 返回变化后的

A=A+1;

Return A;

}

A && B：如果A为假就不执行B

赋值是赋给等式右边，即改变等式右边变量的值

a=(b=2)+c √ a=2+c

(b=2)+c = a × 2+c = a

A>>2{

B=A;

B>>2;

Return B;

}

rand()%k = [0, k-1]

float m = m\*100+0.5 (不会去掉小数点后的数)

int a = m; 把实型赋给整型，会去掉小数点后的数

A % B : 如果A<B那么 A%B=A；

赋值运算符 = ，把右边的值赋给左边

int a=b=c=d=1; (这里只声明了a，b)

Int a, b, c

首先d=1;

1. >=1; a=a>>1;

a = 2, b=3, c=4

a\*=16+(b++)-(++c) （表达式里用的是返回值，表达式之外的用本身的值）

（b++）== 3

（++c） == 5

a\*=16+3-5 == 14

在赋值运算符右边的视为***表达式***

赋值运算符：=

逻辑等：==

逻辑运算结果：||、&&、！只能有0或者1，但是

i = 7

i++ && 0 = 0 or 1

cos运用的是弧度值 度数转化成Π

Printf(%d或o, n)

%o: 八进制

%d：十进制

作用域：{ }一个大括号之内的就成为作用域，里面的内容对作用域外的不可见

{

int a = 1;

Int b = 2;

{

Int a = 2; (跟作用域外的有同变量，会覆盖--ps:覆盖是相对于此作用来说 --作用域外的变量)

b = 3; (作用域内没有，就看作用域外的)

}

}

总结，在一个作用域内的变量，优先看作用域内的，如果没有再看作用域外的。

----字符输入

1. scanf(“%c”, &a)

这里 %c，表示输入的一个字符（char）

1. getch()

从输入中读取一个字符

----字符串输入

1. gets(char\*)

可以输入空格

1. scanf(“%s”, s)

输入一个字符串，遇到空格终止输入

----数值输入

1. scanf(“%d”, &a)

从第一个数字开始到非数字结束，如果是 %nd则是把前n位数字赋给a

%f 和 %d一样除此之外可以读入小数点，不可以指定小数的宽度，可以指定整数的宽度

1. %8.6f

表示一共有8位其中后6位为小数。截取小数位要做四舍五入处理

1. 一些输入

%+-d 整数 :+表示显示符号，-表示左对齐

%f float

%lf double

%le 指数形式的双精度指数

1. printf(”%%d“)

%% 输出 % 符号

”a“ == “a\0”

1. 转义字符

\”

\777

\xff

1. 输出格式

%md指输出带符号的十进制整数，给定最小宽度为m位，不足m位左端补空格，超过m位按实际位数输出，%.nf表示以小数形式输出实数，小数占n位，

for(i = 0; i < 4; i++, i++){

for(k = 1; k < 3; k++); == for(k = 1; k < 3; k++){ 表示内层语为空不执行}

}

printf(“\*”);

for(i = 0; i < 4; i++, i++){

for(k = 1; k < 3; k++)

}

printf(“\*”);

C 语言把任何非零和非空的值假定为 true，把零或 null 假定为 false。