面试题11 数值的整数次方

分析：求x的n次方，x是整数或小数，n是整数（...-2,-1,0,1,2...）

|  |  |  |  |
| --- | --- | --- | --- |
| 幂  底数 | 0 | 正数 | 负数 |
| 0 | 0 | 0 | Error |
| 正数 | 1 |  |  |
| 负数 | 1 |  |  |

1. 处理异常和特殊情况【尤其是底数为0，幂为负数的致命异常】
2. 子函数：求一个非0数值的正整数次方
3. 判断int型数值是否等于0，if(0 == x);判断double和float型数值是否等于0或两个小数是否相等，if(abs(num1 - num2)<0.0000001),因为小数都是不精确的，具有一定精确度，判断两个小数是否相等只能规定一个精确度，差值小于这个精确度便认为相等，否则，不等。
4. （2）有公式：a^n = a^(n/2)\* a^(n/2) n为偶数； a^n = a^((n-1)/2)\* a^((n-1)/2)\*a n为奇数； 故可用递归

/\*判断两个小数是否相等\*/

bool DoubleIs0(double num1, double num2) {

if (num1 - num2 < 0.00000001 || num2 - num1 < 0.00000001)

return true;

else

return false;

}

/\*计算x的n次方；x！=0，n = 1,2,3...\*/

double ExponentIsUnsigned(double x, unsigned int n) {

if (1 == n)/\*递归结束条件\*/

return x;

if ((n & 1) == 0)/\*判断一个整数的奇偶用位运算效率高\*/

return ExponentIsUnsigned(x, n >> 1)\*ExponentIsUnsigned(x, n >> 1);

else/\*整数的除2乘2用位运算代替效率高\*/

return x\*ExponentIsUnsigned(x, (n - 1) >> 1)\*ExponentIsUnsigned(x, (n - 1) >> 1);

}

/\*计算x的n次方\*/

/\*value保存计算结果，bool返回异常，底数为0幂为负数返回0，其他返回1\*/

bool Power(double x, int n, double &value) {

if (0 == x && n >= 0) {

value = 0;

return 1;

}

if (0 == x && n < 0)

return 0;

if (x != 0 && n == 0) {

value = 1;

return 1;

}

if (x != 0 && n != 0) {

unsigned int absn;

if (n>0) {

absn = (unsigned int)n;/\*有符号数转换成无符号数\*/

value = ExponentIsUnsigned(x, absn);

}

else {

absn = (unsigned int)(~n + 1);/\*位运算取相反数\*/

value = 1 / ExponentIsUnsigned(x, absn);

}

return 1;

}

}

pow函数：pow(a,b),a,b都是double类型，本函数，实现double的整数次方。