static void Main(string[] args)

{

//i\*j+w\*x

ParameterExpression a = Expression.Parameter(typeof(int),"i"); //创建一个表达式树中的参数，作为一个节点，这里是最下层的节点

ParameterExpression b = Expression.Parameter(typeof(int),"j");

BinaryExpression r1 = Expression.Multiply(a,b); //这里i\*j,生成表达式树中的一个节点，比上面节点高一级

ParameterExpression c = Expression.Parameter(typeof(int), "w");

ParameterExpression d = Expression.Parameter(typeof(int), "x");

BinaryExpression r2 = Expression.Multiply(c, d);

BinaryExpression result = Expression.Add(r1,r2); //运算两个中级节点，产生终结点

Expression<Func<int, int, int, int, int>> lambda = Expression.Lambda<Func<int, int, int, int, int>>(result,a,b,c,d);

Console.WriteLine(lambda + ""); //输出‘(i,j,w,x)=>((i\*j)+(w\*x))’，z对应参数b，p对应参数a

Func<int, int, int, int, int> f= lambda.Compile(); //将表达式树描述的lambda表达式，编译为可执行代码，并生成该lambda表达式的委托；

Console.WriteLine(f(1, 1, 1, 1) + ""); //输出结果2

Console.ReadKey();

}

为了便于大家理解，这点代码构成的Lambda表达式树如下图：

