设

a=

b=

变换：

a=,

b=

=

有

N=2019>e

所以

|  |  |  |
| --- | --- | --- |
| 对于： |  |  |
| 都取自然对数： | blna | alnb |
| 都乘以ab |  |  |

设f(x)=

F’(x)==

xf’(x)>0, f(x)单调增加；

xf’(x)<0, f(x)单调减少；

理解变量或类型定义作用域中的一些细节

1 不同关键字以及变量标识符在不同位置的声明和定义产生了作用域的概念，变量标识符保存到了内存中不同的区域（如BSS段、数据段等全局或静态区、栈段、堆段）；

2 作用域除了用关键字区分以外，最重要的是在上下文中通过其在代码中的位置进行区分。局部作用域是函数体内用“{}”括起来的语句块，局部作用域的局部变量在其声明和定义处开始有效，“}”是一个局部变量作用域结束的标志。“{}”语句块可以是控制结构的，也可以是直接“{}”形成的语句块，“{}”语句块可以嵌套，按最近原则进行匹配。自然，全局变量不隶属于任何“{}”，其作用域也是开始于其声明和定义处。

3 函数声明中的参数的标识符对于C编译器来说是可以是省略的，C编译器只关心其类型信息，所以也就无作用域的问题。函数定义中的参数的作用域为其整个函数体。需要注意的是，在函数调用时，如果函数参数列表中含有同一变量的表达式，如fn(a++, a++, a++); 其结果的结果是从右至左的，因为参数压栈后的出栈是一个后进先出的顺序，注意逗号运算符的结合性是从左至右。

4 关键字extern用于声明在此处之后或其他文件中定义的全局变量，自然无法通过extern在同一文件中去声明其定义的局部变量，因为局部变量一超出其作用域便失效了。

5 在局部作用域内，同名局部变量会屏蔽掉同名的全局变量，在C++中，可以使用域操作符“::”来引用同名全局变量；

6 对于函数作用域，有一个返回值的问题。不管是局部变量还是局部指针变量，都是保存在栈中，称为栈变量，超出其作用域后，自动失效，内存自动回收，所以不能用作返回值。自然，返回其值是可以的，编译器对于值会做为一个临时结果保存起来。返回堆指针或参数指针都是可以的，堆指针指向的内存不会自动回收，而参数指针指向的是函数体作用域以外的内存空间。

7 作用域通常是说变量、常量的作用域，但其实也包括自定义类型的作用域。自定义类型也可以定义在局部作用域中，但通常的做法是定义在文件的最前面或单文件的main函数前面。当然，也可以将类型定义（也包括函数声明）放到一个单独的文件中，其它文件需要时可以包括进去，其实质的效果也是相当于放到了文件的最前面。

迭代器是如何泛化指针与算法独立于容器的？

A goal of generic programming is to write code that is independent of data type. Templates are the C++ tools for creating generic programs. Templates, of course, let you define a function or class in terms of generic type. The STL goes further by providning a generic representation of algorithm. Templates make this possible, but not without the added element of

