template <class T> T calc(const T &a, const T &b);//T为模板类型形参

1. 模板形参的作用域

模板形参的名字可以在声明为模板形参之后直到模板声明或定义的末尾处使用。

模板形参遵循常规名字屏蔽规则。模板形参会屏蔽同名的全局作用域声明的对象。例：

typedef double T;

template <class T> T calc(const T &a, const T &b)

{

//temp是模板形参，会屏蔽double的全局类型别名

T temp = a;

return temp;

}

1. 模板形参名只能在同一模板中使用一次。

template <typename T> T calc(const T &a, const T &b)

{

typedef double T; //error，不能重用

}

1. 同一模板的声明或定义中，模板形参的名字不必相同

template <class T> T calc(const T &a, const T &b);

template <class U> U calc(const U &a, const U &b);

这两个模板的声明得到是同一模板。

非类型模板

//非类型模板形参，即变量类型是确定的

template <class T, size\_t N> void array\_init(T (&parm) [N] /\* 数组的引用\*/)

{

for (size\_t i = 0; i != N; ++i) // i需要定义为size\_t类型的，而且使用!=而不是<=

{

parm[i] = 0;

}

}