C++函数模板

1.定义模板

template<typename T>

T const& max(T const& a, T const& b)

{

return a < b ? b : a;

}

或者

template<typename T>

auto const& max(T const& a, T const& b) //函数的参数不允许自动类型

{

return a < b ? b : a;

}

或者

template<typename T1, typename T2>

auto max(T1 const& a, T2 const& b)

{

return a < b ? b : a;

}

通用函数可变参数模板： //通用可变参数模板 处理不限定个数的参数，处理不同类型

template<typename T, typename...Args>

void showall(const T &value, const Args &...args)

{

std::cout << value << std::endl;

showall(args...); //继续传递

}

可变参数模板:

template <typename T>

void show(T t, ...)

{

std::cout << t << std::endl;

}

2.函数模板的使用

add(T a，T b);

3.模板的重载，模板的重载会根据数据类型自动匹配

//函数模板可以对类型进行优化重载，根据类型会覆盖

//如果仍然要使用模板函数，需要实例化

如：

一个类中：

int add(int a,int b);

template<class T>

T add(T a,T b);

//这种情况下如果ab是int类型，则会默认调用普通函数，要用模板函数则要实例化模板函数：add<int>(a,b);

4.函数模板覆盖

函数模板可以被覆盖，覆盖时候可以加上类型覆盖

如：

父类中：

template<typename T>

void swap(T &a, T &b)

子类中：

//template<>   //这里可有可无

void swap(int &int1,int &int2)