2.decltype

1)计算表达式的类型

sizeof操作符的值是一个整数，表示类型的长度(字节数)。

typeid操作符的值是一个对象，其中包含了类型信息。

decltype操作符的值是一个类型，可用于其它对象的声明。

参见：decltype1.cpp

2)对于函数表达式，decltype将返回该函数返回值的类型，而对于左值表达式，decltype则返回该表达式的左值引用。

参见：decltype2.cpp

3)何时使用decltype

参见：decltype3.cpp

=======================

五、模板改进

1.类型别名

旧语法：typedef 源类型 目标类型;

typedef unsigned int uint\_t;

新语法：using 目标类型 = 源类型;

using uint\_t = unsigned int;

旧语法中的typedef其源类型必须是具体类型，而新语法中的using其源类型除了具体类型以外还可以是模板或者半模板。

参见：using.cpp

2.函数模板和类模板一样都可以带有缺省参数，而且如果满足隐式推断的条件，函数模板参数的缺省值不一定非要写在参数表的最后。无法隐式推断的模板参数取缺省值，否则取隐式推断的类型，只要隐式推断生效，模板参数的缺省值即被忽略。

参见：default.cpp

3.连续出现的模板右尖括号不会再被误以为是右移运算符。

参见：right.cpp

======================

六、初始列表

1.旧语法中对不同类型的变量初始化的形式也不同。

int a = 123; // 赋值式初始化

double b (1.23); // 构造式初始化

int c[] = {1, 2, 3}; // 花括号初始化

struct {

struct {

int year;

int mon;

int day

} bday;

char name[128];

int age;

} student = {{2000,1,1},"张飞",16}; // 花括号初始化

-------------------

2.新语法以旧语法中的花括号初始化形式为基础，设计了列表初始化语法，统一了不同的初始化形式：

数据类型 变量 (=){初始化列表}; //等号可有可无

参见：init1.cpp

--------------------

聚合类型包括：

1. 任意类型的数组；

2)满足特定条件的类：

A.无自定义构造函数；

B.无私有和保护的非静态成员变量；

C.无基类

D.无虚函数

E.无通过"="或"{}"在类声明部分被初始化的非静态成员变量

1. 聚合类型的元素或成员可以是聚合类型也可以是非聚合类型

4)对聚合类型使用列表初始化，相当于对其中的元素逐一初始化，而对非聚合类型使用列表初始化，相当于用初始化列表中的值作为参数，调用相应的构造函数

------------------

3.变长初始化表

template<typename T> class initializer\_list<T>;

轻量级列表容器，其中仅保存初始化列表元素的引用，而非其副本。

参见：init2.cpp