1.继承构造函数：using A::A

2.类内初始化 ：int a {1}

3.委派构造函数：A(int a) : A() {},不可同时使用初始化列表 不可环状 构造函数的泛型化(将委托泛型化)

4.右值引用：移动构造函数 A(A &&a); 右值引用 ： A&& (c++分为 将亡值，纯右值，左值 c98中使用常量左值饮用为右值续命，true是右值，)，生命左值常量引用和右值引用构造函数是安全的设计，右值不成将会到左值构造

5.std 函数 is\_rvalue\_reference ..可以判断是什么值

6.std::move 将左值强制转化为右值

7.完美转发 不懂

8.initia\_list<> 初始化列表

9.初始化列表有更广泛的使用

10.POD概念 =default 使构造函数平凡化

11.A operator ""\_C(….) 定义后缀，用于通过字面值构造临时对象

12.inline namespace 內联名字空间

13.using 有定义类型别名的功能

Using sint = int

Template<typename T> using Mm = map<T, char>

14.多用auto

B &A();

auto b = A() b:B

auto &bb = A() b:B&

* 1. auto 对象的拷贝无法带有 volatile 以及 const 属性。但是引用可以

16. auto 可以连续声明多个相同类型的变量

17.类中 非静态成员 不能是auto

* 1. Typeid( 对象／基本类型名 int) .hash\_code() 返回类型的唯一hash值

Int a； int b; a，b的hash\_code相同

* 1. decltype 类型推导 匿名对象类型
  2. Decltype 可以 保留 volatile 与 const。 可以通过 is\_const 与 is\_volatile 判断

Int a; decltype(a) b = 1; b:int

* 1. Decltype 实现 函数追踪返回类型
  2. For\_each 例子
     1. Void action(T &t) {t \*= 2;}

数组 for\_each(pstart, pend, action )

* 1. 匿名的类 enum
  2. Enum class A ： int{c=1,s=2};
     1. 强作用域
     2. 指定类型
  3. Unique\_ptr 和 shared\_ptr

Weak\_ptr .lock() 返回 shared\_ptr指出该内存是否还被占用

* 1. Constexpr 声明 可以将运行时常亮值 转化为 编译时常量值
  2. 第六章没看完 看到变长，看第七章
  3. Nullptr 表示空指针，隐士转换只能转换为空指针类型
  4. 为了POD 使用 =default 在自定义重载了默认函数后默认函数后 添加在 相应默认函数后 恢复其POD 以求优化
  5. =delete 可以删除默认函数的缺省版本
  6. lambda函数只能捕捉父作用域的自动变量，不可捕捉父域之外，如静态变量，lambda函数默认为const添加mutable 表示去掉const特性，可以改变按值捕获的lambda中捕获的值
  7. 仿函数 ： 类重载 ()
  8. 常量函数不能在函数体中修改非静态常量