Chapter 1 CPP

1.1 代码规范

修改注释，例如 —[[修改的代码]]——

1.2 Basic Grammer

1.2.1 Books

Bjarne Stroustrup 推荐要掌握的的 5 种语言：C++、Java、Python、Ruby 和 JavaScript

c++ primer v5, the author is the creator of the first c++ compiler.

the c++ programming language v4, the author is the creator of the c++.

the c++ standard library: a tutorial and reference.

sizeof 程序存储分布有三个区域：栈、静态和动态。

sizeof 操作符，计算的是对象在栈上的投影体积。

不管指针指向的内容在什么地方，sizeof 得到的都是指针的栈大小

C++ 中对引用的处理比较特殊；sizeof 一个引用得到的结果是 sizeof 一个被引用的对象的大小。

```
关于sizeof的两个精巧的宏实现。
非数组的sizeof:
#defne _sizeof(T) ( (size_t)((T*)0 + 1) )

数组的sizeof:
#define array_sizeof(T) ( (size_t)(&T+1) - (size_t)(&T) )
```

Reference The reference is ofen used as parameters of a function. rreference is a const pointer

```
// const T& or const T cannot be passed to T&
 …
int var(1);
```

```
int &ref = var; //ref is always the second name of var.
…
int& func(const double& iVal,...){...}
func() = 3; // We can use the function just as a variable
```

Const

```
// const T* can be transformed to T* using (T*)
 …
int var(1);
const int* pCanNotModifyVar = &var; //const T* p CANNOT modify the variable it points to.

const int& num = 10;//ok
int& num = 10;//wrong
```

const object function can only be called by const object.

```
void testFuncAdd20210718_2()
{
    const int a = 10;
    int* pConstModifier = (int*)&a;// compiler finds and allocates memory
    const int *q = &a;
    {
        *pConstModifier = 20;
    }
    std::cout <<a <<std::endl;//10
    std::cout <<*pConstModifier <<std::endl;//20
    std::cout << (&a == pConstModifier ) <<std::endl;//1
    std::cout << (q == pConstModifier ) <<std::endl;//1
}
```

IO dec,hex, oct

RTTI 与异常处理 RTTI，运行时类型信息，在类层次结构中漫游，可向上、向下、平行转换，可实现反射、高级调试等功能。

异常处理：安全的 longjmp 机制。

```
void function(void) throw(something){//...}
```

构造函数中抛出异常会产生不完整的类，永远不会调析构。解决方法：做一个有 T* 数据成员的模板类 A，我们创建的类 B 中的 int* pVal 要在 B 的构造中 new，现在改为存 A<int>pVal，在 B 的构造中 new，这样 new 委托给了类 A，在我们的类 B 的构造中 throw 也不会导致内存泄露。

1.3 Function

函数定义中可以为最右边连续若干参数有省却值, 可以用于扩充函数参数时减少对原有代码的修改。

Inline

We use Inline to tell Compiler to do copy-paste. The function is short and called many times. 调用时间少了，可执行文件体积增大

完成形参的构造（如调用类的拷贝构造等）之后，再进入函数体内。

Oveload

重载。Functions with the same name but the variables are different.We donot care the returned type.

1.4 Object

1) Do not set data member public. 2) Try the best to use reference as the io of functions. 3) Use const as most as possible. 4) use initial list in constructor. 结构化程序设计没有封装和隐藏的概念，导致数据结构与函数之间相互的关系、函数之间的调用关系不明确。把数据结构和对其进行的操作方法放在一起。

stack object, also called auto object. static object will be created once, it will not be delete until the program ends. global object, outside every , its scale is the whole program.

类的实现也可以直接在.h 中。

1.4.1 Basic

给出了构造函数，编译器不默认提供无参构造函数。复制构造函数，只有 1 个

```
C（const C& iC);
C();//临时对象，下一个语句就析构了。
```

返回值是类 A 的实例时，函数返回时调用 A 的复制构造函数

声明时可给出变量的初始值，后面可在构造函数中重新赋值。

解决菱形继承，在第一级继承时使用 virtual:public, 这样后面每次实例化时要调用虚基类的构造函数。

```
//类型转换构造函数
C（int iIn)
C c1 = 12, c2;
```

```
c2 = 9;//error
c2 = C(9); // ok
```

```
//When we new a class:
C* pC = new C(1,2);
// The compiler transformed into 3 steps:

void* mem = operator new(sizeof C);//Inside it calls malloc(n);
pC = static_cast<C*>(mem);
pC->C::C(1,2);

// When delete a class:
delete pC;
// It turns into:
C::~C(pC);//Clear the data.
operator delete(pC);//Insid it calls free(pC); //free the memory.
```

参数对象消亡时调用析构函数；函数返回时是生成临时对象返回，在临时对象调用的那条语句之后，临时对象消亡，调用析构。

When we do something in constructor, we need to make sure the function works well in copy constructor.

在构造函数的初始化列表中对类的成员类进行初始化,，顺序是类中成员声明顺序。

1.4.2 Decoration

没有使用成员变量的方法，可以通过空的类指针调用，相当于翻译后传了一个空的 this 指针。
static 方法也可以空类指针调用，static 方法参数中没有 this 指针
static 和 global 在 main 结束后按入栈的顺序析构。
sizeof doesn't contain static variables.
friend function, a class must declare WhichClass::WhichFunction is its friend, so that the specific function can access its private members. Also the same as friend class.
friend relationship cannot be passed, or be inherited.
每个对象的空间中都有 this 指针。（x）

```
提示编译器不生成默认复制构造函数：A(CONST A&)=delete;
提示编译器提供默认构造函数 A()=default;
默认情况一旦写了构造函数，编译器就不生成无参的默认构造函数了。
委托构造函数 A():A(0,0,0){}
```

const function cannot modidfy class members. final, means this function or class cannot be inherited. override, compiler will check the function, and if it is not override a function in its base, it tells you the error. Outsied, a const class can only use const functions.

1.4.3 关系

Composition，复合，A has a B。类 A 有 1 个 B 的成员变量。deque 读作 dek，queue，读作 q。构造时由内而外；析构时由外而内。

Delegation，委托，类 A 有 1 个 B 的指针的成员变量。Composition by reference。pimpl，pointer to implementation。Handle, body。隔离了接口与具体的实现。编译防火墙。reference counting。共享。copy on write，当多人共享，其中某类想要改变共享的类时，拷贝一份，是他脱离共享。

Inheritance, is a, 继承。子类的对象中有父类的成分。父类的析构函数需要是 virtual 的。非虚函数，不希望 derived class override 的函数。虚函数，希望 derived class override 的函数。通过子类对象调用父类的虚函数，这延缓了执行逻辑，设计模式叫做 template method。Application framework 的常用手法，例如打开文件的流程。pure virtual function, derived class 一定要 override。

```
(*(this->vptr)[n])(this);
```

1 份数据，多个窗口，或多种表达形式。

1.5 Template

sortable, comparable, assignable.

Data type int is a comparable modle.

Iterators connects the containers with algorithms.

1.6 Memory

```
new是先分配memory（调用operator new,内部调用malloc返回void*，再static_cast转到类指针），再调用构造函数

delete先调用析构函数，再调用operator delete（内部调用free）释放内存
```

Dynamic

```
T* p = new T;
if(p){
    delete p;
    p = nullptr;
}
int arraySize(6);
T* p = new T[arraySize];
if(p){
    delete[] p;
    p = nullptr;
}
```

1.7 STL

1.7.1 Contianer

```
Sequence and reversible:
list

Sequence and random access:
vector, deque

Associative and reversible:
   set, multiset, map, multimap

   array, , forward_list,

unordered_set, unordered_multiset, unordered_map, unordered_multimap
```

Chapter 2 HTML

subsectionBasic Html, 超文本标志语言。前端做的多。asp aspx jsp php 用 adobe 的 Dw 开发。

Chapter 3 JAVA

3.1 Basic Grammer

```
DOS command
cd\ //back to the root
md xxx // make directory
rd xxx // remove directory
del test.txt // delete the file
rd /s /q testFolder
```