## Chapter 1 Accustoming Yourself to C\_Plus\_Plus

No matter what’s your programming background, c++ would make you feel familiar since it is such a powerful language, with a lot of features, but before you understand all its feature and power, you need to figure out its rules.

Rule 01: View C++ as a federation Language.

*History:*

At first, C++ is only C with Object - Oriented Feature, which is the initial C With Classes stands for.

However, when it goes maturity, it turns to more active and no constraint, more bold and more adventure, and starts to accept all kinds of viewpoints, features and programming strategies which is different from C with Classes.

Today, C++ is the Multi - paradigm Program Language which supports Procedural, Object - Oriented, Functional, Generic, Meta - Programming language.

*Principle:*

*The simplest method is to view C++ as the Federation Language which consists of related Programming Language but not the Single Language.* In the Sub - Language, all kinds of rules and examples are all easy to understand, read, and easy to remember. However, when you move from one Language to another Language, then rules maybe change. To understand C++, you must need to know its main four Sub - Languages.

1. *C* - *C++ is C.* *Blocks, statements, pre - processor, built - in data types, arrays, pointers which are all comes from C.* However, many solutions of C++ is just much more sophisticated C solutions, but when you work by using the C part in C++, then the high efficiency of C++ may show that the deficiency of C: no templates, no exceptions, no overload...
2. *Object - Oriented C++* - *This part is also required by C with Classes, including: classes ( Constructor and Destructor ), encapsulation, inheritance, poly - morph - ism, virtual function ( No Dynamic )..., and so on.*
3. *Template C++* - *This is the Generic Programming part of C++, and it’s also the least experience that Programmer has. Actually, Generic Programming is so powerful that it brings the new Programming Paradigm.*
4. *STL* - *STL is one Standard Template Library, but it is the most special one.* It has close cooperation with containers, iterator, algorithms, and function objects, but templates and Programming Library can be built by other methods. STL has its own special method, when you work with STL, then you need to obey its rule.

Remember that for all these four Sub - Languages, when you need to convert from one Sub - Language to another, may be High Efficiency Programming needs you to change your strategy, then there has no need to surprise.

*Example:*

1. *When you are using C part of C++, then it is obvious that Pass - By - Value is much more efficient than Pass - By - Reference.*
2. *When you move from C Part of C++ to Object - Oriented C++, since user - defined Constructor and Destructor, so it’s better to use Pass - By - Reference - to - Const.*
3. *When you step into STL, you should know that, iterators and functions are all created on the base of C Pointers, so for iterators and function object in STL, then Old Pass - By - Value rule can be used again.*

*Conclusion:*

C++ is the Programming Federation Language which has four main Sub - Languages, which has its own constraint. Remember these four rules would help you understand the Programming Language better.