Template in C ++

The template is the foundation of generic programming, ie code according to which is independent of any specific type.

The template is the foundation of generic programming, ie code according to which is independent of any specific type.

A Template is a blueprint or method to create a class or a general function. Container libraries like iterators and algorithms are examples of general programming and have been developed using the concept of Template.

Each container has a single definition, vector example, but we can define many different vector series, for example: **vector or vector** .

Template is a keyword in C ++, we can understand that it is an abstract data type, specific to the basic data types. A template is a keyword that tells the compiler that the following code defines multiple data types and that its source code will be compiled to correspond to each data type during compilation. There are two types of templates in C ++:

Function Template: is a function template that allows you to define general functions for multiple data types.

Class template: is a class template that allows to define general classes for many data types

Function Template in C ++

Here is the general syntax of a Function Template definition in C ++:

```
template  class  kieu_du_lieu >  kieu_tham_chieu ten_ham ( danh sach tham so ) 
```

Here, **kieu_du_lieu** is a name of a data type used by the function. This name can be used within the function definition.

The following is an example of Function Template that returns the maximum value of two values:

```
#include #include using namespace  std ; template  typename QTM > inline QTM co
```

At the **template** line, if you use the class instead of typename as in the syntax, the program still runs normally. Because with the class and typename defined in C ++. As for other names, like **tenKieuCuaToi** , it won't run !!

Compile and run the above C ++ program to see the results:

Class Template in C ++

Just like when we can define Function Template, we can also define the Class Template in C ++. The general syntax of defining a Class Template in C ++ is:

```
template  class  kieu_du_lieu > class  ten_lop  { . . . }
```

Here, **kieu_du_lieu** is the type name, which will be determined when a class is declared. You can define more than one generic data type by using a list separated by commas.

The following example defines the Stack class and implements general methods to push and pop elements from that Stack. (Stack: stack, push: add a new node to the top of the stack, pop: operation takes 1 element from the stack top).

```
#include #include #include #include #include using namespace  std ; template  c
```

In the above program, the **out_of_range** exception is already defined in C ++. Compile and run the above C ++ program to see the results:

According to Tutorialspoint

Previous article: Namespace in C ++

Next article: Preprocessor in C ++