C++ STL的三种特殊容器：栈stack，队列queue，优先队列priority\_queue。

1. 栈：后进先出LIFO。

例：程序stl\_test79

*stack*<int> st;

st.*push*(1);

st.*push*(2);

st.*push*(3);

// top()：不弹出元素

*cout* << st.*top*() << ' ';

st.*pop*();

*cout* << st.*top*() << ' ';

st.*pop*();

// modify top element

st.*top*() = 77;

st.*push*(4);

st.*push*(5);

st.*pop*();

while (!st.*empty*())

{

*cout* << st.*top*() << ' ';

st.*pop*();

}

输出为：

3 2 4 77

例：程序stl\_test79

class A {

public:

A(*string* name) : name\_(name)

{

*cout* << "A constructor" << *endl*;

}

A(const A& rhs) : name\_(rhs.name\_)

{

*cout* << "A copy constructor" << *endl*;

}

A(A&& rhs) : name\_(rhs.name\_)

{

*cout* << "A move constructor" << *endl*;

}

~A()

{

*cout* << "A destructor" << *endl*;

}

void Print()

{

*cout* << "this is " << name\_ << *endl*;

}

private:

*string* name\_;

};

*stack*<A> st1;

A a1("a1");

st1.*push*(a1);

a1.Print();

A a2("a2");

st1.*push*(*move*(a2));

// 使用move移动后

// 源对象的生命周期

// 并未结束，但在实际

// 使用过程中，可以在

// move构造函数中让其

// 处于可析构状态（释放

// 其分配的内存，文件句柄等）

// 程序员也不应再使用此源对象

a2.Print();

输出为：

A constructor

A copy constructor

this is a1

A constructor

A move constructor

this is a2

A destructor

A destructor

A destructor

A destructor

1. 队列：先进先出，FIFO

例：程序stl\_test80

*queue*<*string*> q;

q.*push*("These ");

q.*push*("are ");

q.*push*("more than ");

*cout* << q.*front*();

q.*pop*();

*cout* << q.*front*();

q.*pop*();

q.*push*("four ");

q.*push*("words!");

// 返回最后一个元素

*cout* << q.*back*();

输出为：

These are words!

1. 优先队列，priority\_queue，二叉堆。

priority\_queue内的元素已根据其值排序，默认的是以operator<形成降序排列，使用top()/pop()将会访问/移除的下一个元素就是“数值最大的元素”。

例：程序stl\_test81

*priority\_queue*<float> q;

q.*push*(66.6f);

q.*push*(22.2f);

q.*push*(44.4f);

*cout* << q.*top*() << ' ';

q.*pop*();

*cout* << q.*top*() << *endl*;

q.*pop*();

q.*push*(11.1f);

q.*push*(55.5f);

q.*push*(33.3f);

while (!q.*empty*())

{

*cout* << q.*top*() << ' ';

q.*pop*();

}

输出为：

66.6 44.4

55.5 33.3 22.2 11.1

1. bitset

bitset创建一个内含bit或Boolean值且大小固定的array。

例：程序stl\_test82

// numColors用来指定枚举的有效个数

enum Color {red, yellow, green, blue, white, black, orange, numColors} ;

// bitset的模板形参不是类型

// 而是不带正负号的整数，

// 指定可容纳的bit数

// numColors = 7u

*bitset*<numColors> useColors;

useColors.*set*(red);

useColors.*set*(blue);

*cout* << "bitfield of used colors: " << useColors << *endl*;

*cout* << "number of used colors: " << useColors.*count*() << *endl*;

*cout* << "bitfield of unused colors: " << ~useColors << *endl*;

*cout* << *endl*;

// 整数值和bit序列之间的转换

*cout* << "267 as binary short: "

<< *bitset*<*numeric\_limits*<unsigned short>::*digits*>(267)

<< *endl*;

*cout* << "267 as binary long: "

<< *bitset*<*numeric\_limits*<unsigned long>::*digits*>(267)

<< *endl*;

*cout* << "10,000,000 with 24 bits: "

<< *bitset*<24>(10000000) << *endl*;

*cout* << "12, 345, 678 with 42 bits: "

<< *bitset*<42>(12345678).*to\_string*() << *endl*;

*cout* << "\"10001010011\" as number: "

<< *bitset*<100>("1000101011").*to\_ullong*() << *endl*;

输出为：

bitfield of used colors: 0001001

number of used colors: 2

bitfield of unused colors: 1110110

267 as binary short: 0000000100001011

267 as binary long: 00000000000000000000000100001011

10,000,000 with 24 bits: 100110001001011010000000

12, 345, 678 with 42 bits: 000000000000000000101111000110000101001110

"10001010011" as number: 555