```cpp
//: Mixins.cpp
#include <string>
#include <ctime>
#include <iostream>
using namespace std;
template <class T> class TimeStamped: public T {
    long timeStamp;
    public:
    TimeStamped() {
        timeStamp = time(0);
        cout << "TimeStamped() " << endl;
    }
    TimeStamped(string str) :T(str){
        timeStamp = time(0);
        cout << "TimeStamped(string str)" << endl;
    }
    long getStamp() { return timeStamp;}
};

template<class T> class SerialNumbered: public T {
    long serialNumber;
    static long counter;  /* = 1; */
    public:
    SerialNumbered() {
        serialNumber = counter++;
        cout << "SerialNumbered()" << endl;
    }
    SerialNumbered(string str):T(str) {
        serialNumber = counter++;
        cout << "SerialNumbered(string str)" << endl;
    }
    long getSerialNumber() {return serialNumber;}
};

//Define and initialize the static storage:
template<class T> long SerialNumbered<T>::counter = 1;

class Basic {
    string value;
    public:
    Basic() { cout << "Basic() " << endl; }
    Basic(string init) {
        value = init;
        cout << "Basic(string init)" << endl;
    }
    void set(string val) {value = val;}
    string get() {return value;}
};

int main() {
    TimeStamped<SerialNumbered<Basic> >  mixin1;
    TimeStamped<SerialNumbered<Basic> >  mixin2("test string 2");
    mixin1.set("test string 1");
    cout << mixin1.get() << " " << mixin1.getStamp() <<
        " " << mixin1.getSerialNumber() << endl;
    cout << mixin2.get() << " " << mixin2.getStamp() <<
        " " << mixin2.getSerialNumber() << endl;
    return 0;
}
```

输出结果：

```
Basic()
SerialNumbered()
TimeStamped()
Basic(string init)
SerialNumbered(string str)
TimeStamped(string str)
test string 1 1293175992 1
test string 2 1293175992 2
```

这里有个注意的地方：虽然是从Basic开始调用，但是如果`TimeStamped(string str)`及`SerialNumbered(string str)`没有初始化列表`T(str)`,那么参数是传不下去的，最终是：

```
Basic()
SerialNumbered()
TimeStamped(string str)
```