

Objects and Classes

【目标】

- 理解抽象数据类型，类的概念
- 定义和实现类
 - 访问控制机制
 - 数据成员
 - 类方法（类函数成员）
- 创建和使用类对象
- 特别的成员函数
 - 构造函数【对象创建时自动调用，可重载以实现用不同参数进行创建】
 - 析构函数【对象撤销时自动调用】
 - const 成员函数
- this 指针
 - 在类内部标记类对象【实体】的指针
- 其中，控制机制
 - 访问控制【实现private数据成员的封装，只能通过public类方法调用】
 - const成员函数【专用于const对象调用的函数】

```
1 // stock20.h -- augmented version
2 #ifndef STOCK20_H_
3 #define STOCK20_H_
4 #include <string>
5
6 class Stock
7 {
8 private:
9     std::string company;
10    int shares;
11    double share_val;
12    double total_val;
```

```

13     void set_tot() { total_val = shares * share_val;
14     }
15     public:
16     Stock();          // default constructor
17     Stock(const std::string & co, long n = 0, double
pr = 0.0);
18     ~Stock();        // do-nothing destructor
19     void buy(long num, double price);
20     void sell(long num, double price);
21     void update(double price);
22     void show()const;
23     const Stock & topval(const Stock & s) const;
24 };
25 #endif
26
27 // stock20.cpp -- augmented version
28 #include <iostream>
29 #include "07-stock20.h"
30 using namespace std;
31 // constructors
32 Stock::Stock()          // default constructor
33 {
34     shares = 0;
35     share_val = 0.0;
36     total_val = 0.0;
37 }
38
39 Stock::Stock(const std::string & co, long n, double
pr)
40 {
41     company = co;
42     if (n < 0)
43     {
44         std::cout << "Number of shares can't be
negative; "
45                 << company << " shares set to
0.\n";
46         shares = 0;
47     }
48     else

```

```

49     shares = n;
50     share_val = pr;
51     set_tot();
52 }
53
54 // class destructor
55 Stock::~Stock()           // quiet class destructor
56 {
57 }
58
59 // other methods
60 void Stock::buy(long num, double price)
61 {
62     if (num < 0)
63     {
64         std::cout << "Number of shares purchased
65         can't be negative. "
66         << "Transaction is aborted.\n";
67     }
68     else
69     {
70         shares += num;
71         share_val = price;
72         set_tot();
73     }
74 }
75 void Stock::sell(long num, double price)
76 {
77     using std::cout;
78     if (num < 0)
79     {
80         cout << "Number of shares sold can't be
81         negative. "
82         << "Transaction is aborted.\n";
83     }
84     else if (num > shares)
85     {
86         cout << "You can't sell more than you have! "
87         << "Transaction is aborted.\n";
88     }

```

```

88     else
89     {
90         shares -= num;
91         share_val = price;
92         set_tot();
93     }
94 }
95
96 void Stock::update(double price)
97 {
98     share_val = price;
99     set_tot();
100 }
101
102 void Stock::show() const
103 {
104     using std::cout;
105     using std::ios_base;
106     // set format to #.###
107     ios_base::fmtflags orig =
108         cout.setf(ios_base::fixed,
109 ios_base::floatfield);
110
111     std::streamsize prec = cout.precision(3);
112
113     cout << "Company: " << company
114         << " Shares: " << shares << '\n';
115     cout << " Share Price: $" << share_val;
116     // set format to #.##
117     cout.precision(2);
118     cout << " Total worth: $" << total_val << '\n';
119
120     // restore original format
121     cout.setf(orig, ios_base::floatfield);
122     cout.precision(prec);
123 }
124
125 const Stock & Stock::topval(const Stock & s) const
126 {
127     if (s.total_val > total_val)
128         return s;
129     else

```

```
128         return *this;  
129     }
```