

河南工业大学

课程设计报告

猜数字游戏

课程名称: 面向对象程序设计

专业班级: 软件 1305 班

小组成员: 田劲锋 201316920311

邢志鹏 201316920315

王增辉 201316920309

任课教师: 王献荣

完成时间: 2015 年 1 月 15 日

目录

1	题目内容及设计要求	2
2	总体方案设计	3
2.1	总体功能框图	3
2.2	类的设计说明	3
2.2.1	Number 类	3
2.2.2	Score 类	5
2.2.3	UI 类	7
2.3	主要算法流程图	10
3	程序清单及注释	12
3.1	许可证	12
3.2	工程文件	12
3.3	头文件	13
3.4	主程序	14
3.5	邢志鹏的实现	15
3.6	田劲锋的实现	20
3.7	辅助文件	24
4	运行结果与分析	26
5	总结	27
5.1	田劲锋的总结	27
5.2	邢志鹏的总结	27
5.3	王增辉的总结	27
A	股票交易系统	28
A.1	设计类图	28
A.2	题目内容	29
A.3	头文件	29
	参考文献	35

1 题目内容及设计要求

猜数字游戏

内容及要求：

猜数：用户从键盘输入4位不重复的数，来匹配计算机给出的4位随机数，若数字和位置均等同，表示用户赢了。每猜一次，计算机均给出提示信息 (x, y) ， x 表示数字、位置都匹配的个数， y 表示数字匹配但位置不匹配的个数。

- (1) 设计有好的中文交互界面；
- (2) 按8888键，可以得到更详细的帮助信息，如：第1位数字正确等。
- (3) 按7777键后，可以查看计算机所给的4位数，但需要输入密码，密码自定。
- (4) 猜的结果以分数给出，每猜错一次扣40分，若猜对1个数，奖励20分。
- (5) 每次游戏结束后将分值存盘，文件名自定。

难度系数：

1.1

以上题目要求解决问题步骤为：

- (1) 应用系统分析，建立该系统的功能模块框图；
- (2) 分析系统中的各个实体及它们之间的关系；
- (3) 根据问题描述，设计系统的类层次；
- (4) 完成类层次中各个类的描述；
- (5) 完成类中各个成员函数的定义；
- (6) 完成系统的应用模块；
- (7) 功能调试；
- (8) 完成系统总结报告。

2 总体方案设计

2.1 总体功能框图

考虑猜数功能的实现，为了实现最大程度的解耦，将与用户交互的操作全部封装在 UI 类中。

考虑对于全局唯一的用户，需要一个全局唯一的计分器，这里设计为单例模式的 Score 类。

考虑到每次运行程序用户会进行多次猜数，对于每次猜数都实例化一个 Number 对象，并提供对猜数的检查功能。

图 1 展示了各个类之间的关系。

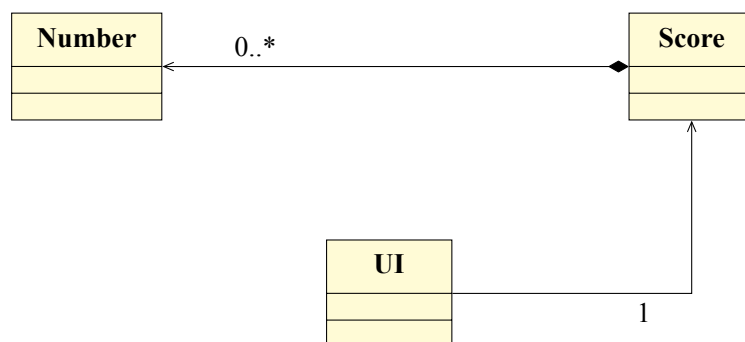


图 1: 猜数字游戏各个类之间的关系

2.2 类的设计说明

2.2.1 Number 类

Number		
# number: <i>int</i>	//待猜的数	
- numbers: <i>int</i> []	//数的拆分数组	
# count: <i>int</i>	//猜数次数	
+ Number()		//初始化产生随机数
+ setNumber(number: <i>int</i>): void		//拆分
- genRand(): <i>int</i>		//产生符合要求的随机数
+ guess(number: <i>int</i>): x: <i>int</i> , y: <i>int</i>		//猜数并返回提示信息
+ detail(number: <i>int</i>): <i>int</i> []		//猜数并返回正确的位数组
+ answer(): <i>int</i>		//返回正确的答案
+ ~Number()		//析构函数

图 2: Number 类

Number::Number

Number::Number();

返回 构造函数

描述 调用 `Number::genRand()` 来产生符合要求的随机数，调用 `this→setNumber()` 来设置 `this→number` 和 `this→numbers`。初始化 `this→count` 为 0。

Number::genRand

int Number::genRand();

返回 生成的随机数

描述 生成符合要求的随机数。算法见第 10 页的图 5。

Number::setNumber

void Number::setNumber(int number);

返回 无

描述 将 `number` 放置到 `this→number` 里，并将 `this→numbers` 置为对应的拆分数组，这两个是一一对应的关系，不可分拆，只是单纯的方便后面的计算。

Number::guess

std::pair<int, int> Number::guess(int number);

返回 提示信息 (x, y)

描述 猜数。用户猜的是 `number`，判断是否正确并返回提示信息 (x, y) ， x 表示数字、位置都匹配的个数， y 表示数字匹配但位置不匹配的个数。全部正确时应该返回 $(4, 0)$ 。每调用该方法时 `this→count` 应该自增 1。算法见第 11 页的图 6。

Number::detail

std::vector<int> Number::detail(int number);

返回 详细信息的数组

描述 对应 8888 键的功能。用户上一次猜的是 `number`，返回一个数组，数组的每个元素表示哪一位正确了。全部正确时应该返回 $\{1, 2, 3, 4\}$ 。每调用该方法时 `this→count` 应该自增 1。算法见第 11 页的图 7。

Number::answer

int Number::answer();

返回 正确答案

描述 对应 7777 键的功能。返回正确答案的 `this→number`。每调用该方法时 `this→count` 应该自增 1。

2.2.2 Score 类

Score	
# PLUS: <i>int</i> = 20	//加分分值
# MINUS: <i>int</i> = 40	//减分分值
# score: <i>int</i>	//得分
+ <u>getScore(): <i>int</i></u>	//获取得分
# lastNumber: <i>int</i>	//用户上一次猜的数
# numbers: <i>Number</i> []	//存储每次猜数对象
- password: <i>string</i>	//密码
- Score()	//构造函数
+ <u>getInstance(): <i>Score</i>&</u>	//获取实例
+ ~Score()	//析构函数
+ newGame(): <i>void</i>	//新建游戏
+ guess(number: <i>int</i>): <i>bool</i> , <i>string</i>	//猜数
# plus(): <i>void</i>	//加分
# minus(): <i>void</i>	//减分
+ read(): <i>int</i>	//读入分数
+ write(): <i>void</i>	//写出分数
+ checkPassword(password: <i>string</i>): <i>bool</i>	//检查密码

图 3: Score 类

Score::Score

Score::Score();

返回 构造函数

描述 单例模式的构造函数是 `private` 的。构造函数尝试调用 `this→read()` 从文件中读取分数和密码，初始化 `this→score` 和 `this→password`。初始化 `this→lastNumber` 为特殊值表示用户还没有输入任何数。

Score::getInstance

static Score& Score::getInstance();

返回 唯一的 Score 实例引用

描述 单例模式中，获取唯一的 Score 实例。这是个静态方法。

Score::~~Score

Score::~~Score();

返回 析构函数

描述 调用 `this→write()` 向文件中写出得分。释放所有申请的动态内存。

Score::newGame

`void Score::newGame();`

返回 无

描述 创建一个新的 Number 实例，添加到 `this→numbers` 数组的尾部。

Score::guess

`std::pair<bool, std::string> Score::guess(int number);`

返回 是否猜对以及提示信息

描述 用户输入了 `number`。

对于特殊情况，如果 `number` 是 8888，则调用当前 Number 对象（即 `this→numbers` 数组的最后一个元素）的 `detail()` 方法，参数为 `this→lastNumber`，并返回猜数失败和详细的帮助信息，即“第 *a, b, c* 位数字正确”，注意如果 `this→lastNumber` 被标记成特殊值则表示用户直接输入了 8888 来作弊，这时候应该返回猜数失败和错误提示，而不应该调用 `detail()` 来返回帮助信息；如果是 7777，则调用 `answer()` 方法来查看答案，并返回猜数失败和正确答案，即“正确答案是 *number*”。特殊情况不加减分数。

对于一般的猜数，则置 `this→lastNumber` 为 `number`，并调用 `guess()` 进行正常的猜数流程，如果猜对则返回猜数成功和祝贺信息；猜错则返回猜数失败和提示信息 (*x, y*)，即“数位匹配 *x* 个，数匹配位不符 *y* 个”。猜对的加分，猜错的则减分。

Score::plus

`void Score::plus();`

返回 无

描述 给 `this→score` 加上 `Score::PLUS` 的分值。

Score::minus

`void Score::minus();`

返回 无

描述 给 `this→score` 减去 `Score::MINUS` 的分值。

Score::read

`int Score::read();`

返回 读入的得分

描述 从文件中读取得分和密码，分别放到 `this→score` 和 `this→password` 里。如果文件不存在，则初始化 `this→score` 为 0，初始化 `this→password` 为“root”，并调用 `this→write()` 方法将其写出。写出后再尝试读入。

Score::write

void Score::write();

返回 无

描述 向文件中写出得分 `this→score` 和密码 `this→password`。要写入的文件名应该和 `this→read()` 中的读入文件保持一致，为了避免 Magic Number 是使用，这里应该使用一个全局的常量或者静态变量来存储。

Score::checkPassword

bool Score::checkPassword(std::string password);

返回 密码是否正确

描述 对比 `password` 和 `this→password` 是否一致。

2.2.3 UI 类

«utility» UI	
+ Main(): void	//主循环
+ MainMenu(): int	//主菜单
+ NewGame(): void	//新游戏
+ GuessNumber(n: int): bool	//猜数字
+ ViewDetail(): void	//8888
+ ViewAnswer(): void	//7777
+ ShowScore(): void	//显示得分
+ InputPassword(): bool	//输入密码
+ ReadHelp(): void	//查阅帮助

图 4: UI 类

UI::Main

void UI::Main();

返回 无

描述 循环调用 `UI::MainMenu()`，根据其返回值调用 `UI::NewGame()` 或者 `UI::ShowScore()`。直到其返回 0 表示退出，此时中止循环。

UI::MainMenu

void UI::MainMenu();

返回 无

描述 显示主菜单，等待用户输入选项，输入后返回选项值。

定义：(1) 新游戏；(2) 显示得分；(3) 显示帮助；(0) 退出。

对于用户的其他不合理输入，一律解析为 0，表示退出。

UI::NewGame

void UI::NewGame();

返回 无

描述 开始新游戏。首先调用单例 Score 的 newGame 方法，然后显示提示信息，等待用户输入。判断用户的输入，如果是小于等于 0 的数则返回上一级表示结束本轮游戏；否则调用 UI::GuessNumber() 进入猜数流程，循环这个过程直到该方法返回真表示猜对，显示祝贺信息。

UI::GuessNumber

bool UI::GuessNumber();

返回 是否猜对

描述 判断用户的输入，如果是 8888 则调用 UI::ViewDetail()，如果是 7777 则调用 UI::ViewAnswer()。对于正常的输入，则直接调用 Score 实例的 guess() 方法，显示其返回的提示字符串，显示当前得分。

UI::ViewDetail

void UI::ViewDetail();

返回 无

描述 8888 功能。调用 Score 实例的 guess() 方法，并显示提示字符串。

UI::ViewAnswer

void UI::ViewAnswer();

返回 无

描述 7777 功能。首先调用 UI::InputPassword() 来进行密码输入和验证，允许三次密码输入，如果验证失败则直接返回；如果验证成功，则调用 Score 实例的 guess() 方法，并显示提示字符串。

UI::ShowScore

void UI::ShowScore();

返回 无

描述 显示得分。

UI::InputPassword

bool UI::InputPassword();

返回 密码是否验证通过

描述 提示用户输入密码，设置控制台属性，等待用户输入。在 Windows 下，密码输入后回显成星号 “*”；在 Linux 下，密码输入不回显，但退格键仍应可用。这些操作可以调用 `getpass()` 函数，在 `Password.h` 中提供。密码输入后，调用 `Score` 实例的 `checkPassword()` 方法来验证密码的正确性。

UI::ReadHelp

void UI::ReadHelp();

返回 无

描述 显示如下帮助提示信息：

1. 游戏目的是猜一个四位数，且这个四位数每位都不相同
2. 每次猜数都会有提示
 分别表示数字、位置都匹配的个数，数字匹配但位置不匹配的个数
3. 猜对了加 20 分，猜错了减 40 分
4. 猜 8888 可以得到详细的提示
5. 猜 7777 可以直接看答案，但需要密码
6. 猜 0 或负数会退出该轮游戏，但仍会计分哦

2.3 主要算法流程图

产生一个指定的随机数的 `Number::genRand()` 方法，即每次从 0..9 中选取一个数放到数组里。由于产生的是四位数，要求千位不能为 0，所以产生千位时采用了特殊处理。具体算法的伪代码描述见图 5。

```
NUMBER::GENRAND()
1  S = {}
2  T = {1, 2, ..., 9}
3  S[0] = 从 T 中随机选取一个数
4  T = T - {S0} + {0}
5  for i = 1 to 3
6      S[i] = 从 T 中随机选取一个数
7      T = T - {Si}
8  n =  $\overline{S_0 S_1 S_2 S_3}$ 
9  return n
```

图 5: 生成符合要求的随机数

验证猜数并返回指定的 (x, y) 的 `Number::guess()` 方法，首先需要从 `this→numbers` 获取答案的分解，然后把传进来的数字也进行分解。分解后的两个数组按位比较，累加后得 x 。两个数组求交集之后得到 $x + y$ 。具体算法见图 6。

验证猜数并返回详细信息的 `Number::detail()` 方法，和 `Number::guess()` 非常类似，只是这回返回的是一个数组。由于要求首位表示为 1，所以需要加一处理。具体算法伪代码见图 7。

语言细节相关的实现，不是前提设计的重点，这里不再赘述算法具体应该怎样用语言实现，请参考语言相关文档和标准库文档。

```

NUMBER::GUESS( $n$ )
1   $x = 0$ 
2   $y = 0$ 
3   $S = \text{this} \rightarrow \text{numbers}$ 
4   $\overline{A_0 A_1 A_2 A_3} = n$ 
5  for  $i = 0$  to 3
6      if  $A[i] == S[i]$ 
7           $x++$ 
8   $y = |S \cap A| - x$ 
9   $\text{this} \rightarrow \text{count}++$ 
10 return  $(x, y)$ 

```

图 6: 验证猜数并返回指定的 (x, y)

```

NUMBER::DETAIL( $n$ )
1   $V = \{\}$ 
2   $S = \text{this} \rightarrow \text{numbers}$ 
3   $\overline{A_0 A_1 A_2 A_3} = n$ 
4  for  $i = 0$  to 3
5      if  $A[i] == S[i]$ 
6           $V = V + \{i + 1\}$ 
7   $\text{this} \rightarrow \text{count}++$ 
8  return  $V$ 

```

图 7: 验证猜数并返回详细信息

3 程序清单及注释

3.1 许可证

版权所有 (c) 2014 田劲锋 邢志鹏

保留所有权利

这份授权条款，在使用者符合以下三条件的情形下，授予使用者使用及再散播本软件包装原始码及二进制可执行形式的权利，无论此包装是否经改作皆然：

- * 对于本软件源代码的再散播，必须保留上述的版权宣告、此三条件表列，以及下述的免责声明。
- * 对于本套件二进制可执行形式的再散播，必须连带以文件以及 / 或者其他附于散播包装中的媒介方式，重制上述之版权宣告、此三条件表列，以及下述的免责声明。
- * 未获事前取得书面许可，不得使用柏克莱加州大学或本软件贡献者之名称，来为本软件之衍生物做任何表示支持、认可或推广、促销之行为。

免责声明：本软件是由作者及本软件之贡献者以现状提供，本软件包装不负任何明示或默示之担保责任，包括但不限于就适售性以及特定目的的适用性为默示性担保。作者及本软件之贡献者，无论任何条件、无论成因或任何责任主义、无论此责任为因合约关系、无过失责任主义或因非违约之侵权（包括过失或其他原因等）而起，对于任何因使用本软件包装所产生的任何直接性、间接性、偶发性、特殊性、惩罚性或其他结果的损害（包括但不限于替代商品或劳务之购用、使用损失、资料损失、利益损失、业务中断等等），不负任何责任，即在该种使用已获事前告知可能会造成此类损害的情形下亦然。

3.2 工程文件

Listing 1: Makefile

```
1 RM=rm -f
2 CC= gcc
3 CXX= g++
4 DEFS=
5 PROGNAME= number
6 INCLUDES= -I.
7 LIBS=
8
9 ifdef WIN32
10 OSDEF= -fexec-charset=gbk
11 else
12 OSDEF= -DSYS_UNIX=1
13 endif
14
15 DEFINES= $(INCLUDES) $(DEFS) $(OSDEF)
16 CFLAGS= $(DEFINES)
17
18 SRCS = Number.cpp Score.cpp UI.cpp mylib.cpp cli.cpp
19
20 OBJS = Number.o Score.o UI.o mylib.o cli.o
21
22 .cpp.o:
23     $(RM) $@
24     $(CXX) $(CFLAGS) -c $*.cpp
25
26 all: $(PROGNAME)
27
28 $(PROGNAME) : $(OBJS)
29     $(CXX) $(CFLAGS) -o $(PROGNAME) $(OBJS) $(LIBS)
30
31 clean:
32     $(RM) $(OBJS) $(PROGNAME) core *~
```

Listing 2: Makefile.vs

```

1 CC= cl
2 CXX= cl
3 #DEFS= -nologo -DSTRICT -G3 -Ow -W3 -Zp -Tp
4 DEFS= -nologo /EHsc
5 PROGRAM= number.exe
6 LINKER=link -nologo
7
8 INCLUDES= -I.
9
10 DEFINES= $(INCLUDES) $(DEFS) -DWINNT=1 -DWIN32=1
11
12 CFLAGS= $(DEFINES)
13
14 SRCS = Number.cpp Score.cpp UI.cpp mylib.cpp cli.cpp
15
16 OBJS = Number.obj Score.obj UI.obj mylib.obj cli.obj
17
18 .cpp.obj:
19     $(CXX) $(CFLAGS) -c $< -Fo$@
20
21 all: $(PROGRAM)
22
23 $(PROGRAM) : $(OBJS)
24     $(LINKER) $(OBJS) /OUT:$@ $(LIBS)
25
26 clean:
27     del $(OBJS) $(PROGRAM) core

```

3.3 头文件

Listing 3: Number.h

```

1  /* Number.h
2  * 描述: 猜数字类
3  * 作者: 田劲锋
4  * 创建时间: 2015-1-11
5  * 修改时间: 2015-1-15
6  */
7  #pragma once
8
9  #include <vector>
10 #include <algorithm>
11
12 class Number
13 {
14 public:
15     Number();
16     virtual ~Number();
17
18     void setNumber(int number);
19
20     std::pair<int, int> guess(int);
21     std::vector<int> detail(int);
22     int answer();
23
24 protected:
25     int number;
26     int count; // 猜数次数
27 private:
28     int numbers[4];
29     int genRand();
30 };

```

Listing 4: Score.h

```

1  /* Score.h
2  * 描述: 得分类
3  * 作者: 田劲锋
4  * 创建时间: 2015-1-11
5  * 修改时间: 2015-1-15
6  */
7  #pragma once
8
9  #include <vector>
10 #include <string>
11 #include <algorithm>
12
13 #include "Number.h"
14
15 class Score

```

```

16 {
17     friend class Number;
18 public:
19     virtual ~Score();
20
21     static Score& getInstance();
22
23     void newGame();
24
25     std::pair<bool, std::string> guess(int);
26
27     int read();
28     void write();
29
30     int getScore() const;
31
32     bool checkPassword(std::string password);
33
34 protected:
35     int score;
36     int lastNumber; // 用户上一次猜的数
37     std::vector<Number> numbers;
38
39     const static int PLUS = 20;
40     const static int MINUS = 40;
41
42     void plus();
43     void minus();
44
45 private:
46     Score();
47     std::string password;
48 };

```

Listing 5: UI.h

```

1  /* UI.h
2  * 描述: 用户界面类, 用于用户交互
3  * 作者: 田劲锋
4  * 创建时间: 2015-1-11
5  * 修改时间: 2015-1-15
6  */
7  #pragma once
8
9  #include "Score.h"
10
11 class UI
12 {
13 public:
14     static void Main();
15
16     static int MainMenu();
17
18     static void NewGame();
19
20     static bool GuessNumber(int n);
21
22     static void ViewDetail();
23
24     static void ViewAnswer();
25
26     static void ShowScore();
27
28     static bool InputPassword();
29
30     static void ReadHelp();
31 };

```

3.4 主程序

Listing 6: cli.cpp

```

1  /* cli.cpp
2  * 描述: 猜数字游戏的命令行界面
3  * 作者: 田劲锋
4  * 创建时间: 2015-1-11
5  * 修改时间: 2015-1-12
6  */
7  #include "UI.h"
8
9  int main(int argc, char *argv[])

```

```

10 {
11     UI::Main();
12
13     return 0;
14 }

```

3.5 邢志鹏的实现

Listing 7: Number.cpp

```

1  /* Number.cpp
2  * 描述: 猜数字类
3  * 作者: 邢志鹏
4  * 创建时间: 年月日 20151113
5  * 修改时间: 年月日 20151114
6  */
7  #include<iostream>
8  #include<vector>
9  #include<cstdlib>
10 #include<utility>
11 #include<ctime>
12 using namespace std;
13
14 #include "Number.h"
15
16 Number::Number()
17 {
18     srand(time(0)); //用于生成随机的正确答案, 否则每次游戏答案都是一样的
19     int n = Number::genRand();
20     this->setNumber( n );
21     this->count = 0;
22 }
23
24 Number::~Number()
25 {
26 }
27
28 int Number::genRand()
29 {
30     vector<int> s(4);
31     vector<int> t;
32
33     for (int i = 1; i < 10; i++)
34         t.push_back(i);
35
36     int randIndex = rand() % t.size();
37     s[0] = t[randIndex];
38
39     t.erase( t.begin() + randIndex);
40     t.push_back(0);
41
42     for (int i = 1; i <= 3; i++)
43     {
44         int randIndex = rand() % t.size();
45         s[i] = t[randIndex];
46
47         t.erase(t.begin() + randIndex);
48     }
49
50     int nrand = 1000 * s[0] + 100*s[1] + 10*s[2] + s[3];
51
52     return nrand;
53 }
54
55 void Number::setNumber( int n )
56 {
57     this->number = n;
58     this->numbers[3] = n % 10;
59     this->numbers[2] = (n / 10) % 10;
60     this->numbers[1] = (n / 100) % 10;
61     this->numbers[0] = n / 1000; //初始化数据成员numbers[];
62 }
63
64 pair<int, int> Number::guess(int n)
65 {
66     int v[4] = {0};
67
68     v[3] = n % 10;
69     v[2] = (n / 10) % 10;
70     v[1] = (n / 100) % 10;
71     v[0] = n / 1000;
72 }

```



```

73         int a[10] = {0};
74
75         int x = 0; //表示数字，位置都匹配的个数
76         int y = 0; //表示数字匹配但是位置不匹配的个数
77         for (int i = 0; i < 4; i++){
78             if (v[i] == numbers[i]){
79                 x++;
80             }
81             a[v[i]]++;
82             a[numbers[i]]++;
83         }
84         //求两个集合中元素的交集
85         for (int i = 0; i < 10; i++){
86             if (a[i] == 2)
87                 y++;
88         }
89         y = y - x;
90         this->count ++;
91         return make_pair( x , y);
92     }
93
94     vector<int> Number::detail(int n)
95     {
96         std::vector<int> v(4);
97         std::vector<int> s; //表示要返回的数组
98
99         v[3] = n % 10;
100        v[2] = (n / 10) % 10;
101        v[1] = (n / 100) % 10;
102        v[0] = (n / 1000) % 10;
103
104        for(int i = 0; i < 4; i++){
105            if( v[i] == numbers[i]) //这里数组可以使用直接使用指针吗? numbersthisthis
106                s.push_back(i+1);
107        }
108        this->count ++;
109        return s;
110    }
111
112     int Number::answer()
113     {
114         this->count ++;
115         return this->number;
116     }

```

Listing 8: Score.cpp

```

1  /* Score.cpp
2  * 描述：得分类
3  * 作者：邢志鹏
4  * 创建时间：年月日20151111
5  * 修改时间：年月日20151111
6  */
7
8  #include<iostream>
9  #include<string>
10 #include<sstream>
11 #include<fstream>
12 #include<utility>
13 #include<vector>
14 using namespace std;
15
16 #include"Score.h"
17
18 Score::Score()
19 {
20     this->score = 0;
21     this->password = "root";
22     this->lastNumber = -1;
23     this->read();
24 }
25
26 int Score::getScore() const
27 {
28     return this->score;
29 }
30
31 Score& Score::getInstance()
32 {
33     static Score instance;
34     return instance; //返回的直接就是对象的引用??
35 }
36
37 Score::~Score()
38 {
39     this->write();
40 }

```

```

41
42 void Score::newGame()
43 {
44     Number n;
45     this->numbers.push_back(n);
46 }
47
48 pair<bool, string> Score::guess(int number)
49 {
50     Number &reNumber = this->numbers.back();
51
52     ostringstream s;
53     if (number == 8888){
54         if (this->lastNumber == -1){
55             s << " 您还没有猜数, 查看详细帮助属于作弊行为"          "\n";
56             s << " 返回上一级"                                     ";
57             return make_pair(false, s.str());
58         }
59         else
60         {
61             vector<int> v = reNumber.detail(this->lastNumber);
62
63             if (v.empty() == true)
64             {
65                 s << " 详细帮助: 您的数字所有数字的位置都是错误的"          "\n";
66
67                 return make_pair(false, s.str());
68             }
69             s << " 详细帮助: 您的第"                                     ";
70             size_t i;
71             for (i = 0; i < v.size(); i++){//要强制类型转换为类型int
72                 if (i != v.size() - 1)
73                     s << v[i] << ",";
74                 else
75                     s << v[i];
76             }
77             s << "位数字正确\n";
78             return make_pair(false, s.str());
79         }
80     }
81
82     if (number == 7777){
83         s << " 本游戏正确答案是"                                     ":";
84         s << reNumber.answer();
85         return make_pair(false, s.str());
86     }
87     this->lastNumber = number;
88     pair<int, int> temp = reNumber.guess(number);
89
90     if (temp.first == 4){
91         s << " 恭喜您竟然猜对了!"                                     ~, "\n";
92         Score::plus();
93         return make_pair(true, s.str());
94     }
95     else{
96         Score::minus();
97         s << " 数位匹配"                                     ";
98         s << temp.first;
99         s << "个,";
100        s << "数匹配位不符";
101        s << temp.second;
102        s << "个\n";
103        return make_pair(false, s.str());
104    }
105 }
106
107
108 void Score::plus()
109 {
110     this->score += Score::PLUS;
111 }
112
113 void Score::minus()
114 {
115     this->score -= Score::MINUS;
116 }
117
118 const char* filename = "SAP.txt";
119
120
121 int Score::read()
122 {
123
124     ifstream fin;
125     fin.open("SAP.txt");
126     if(fin.good()){
127         fin >> this->score;
128         fin >> this->password;
129     }

```

```

130         else{
131             this->write();
132         }
133
134         return this->score;
135     }
136
137     void Score::write()
138     {
139         ofstream fout("SAP.txt");
140
141         if (fout.good()){
142             fout << this->score;
143             fout << this->password;
144         }
145         else
146             cout << "文件写入失败";
147     }
148
149     bool Score::checkPassword( string word)
150     {
151         if(word == this->password)
152             return true;
153         else
154             return false;
155     }

```

Listing 9: UI.cpp

```

1  /* UI.cpp
2  * 描述：用户界面类的实现，用于用户交互
3  * 作者：邢志鹏
4  * 创建时间：年月日2015112
5  * 修改时间：年月日2015112
6  */
7
8  #include <iostream>
9  #include <cstdlib>
10 #include <algorithm>
11 #include <string>
12 using namespace std;
13
14 #include "UI.h"
15 #include "mylib.h" //提供密码输入转换成的函数*
16
17
18 void UI::Main()
19 {
20     int n = 1;
21     while (n)
22     {
23         n = UI::MainMenu();
24         if (n == 1)
25         {
26             UI::NewGame();
27         }
28         else if (n == 2)
29         {
30             UI::ShowScore();
31         }
32         else if (n == 3)
33         {
34             UI::ReadHelp();
35         } else
36         {
37             break;
38         }
39         pause();
40     }
41 }
42
43 int UI::MainMenu()
44 {
45     cls();
46     int selection;
47     cout << endl;
48     cout << endl;
49     cout << "猜数字游戏" << endl;
50     cout << "\n" << endl;
51     cout << "1) 新游戏" << endl;
52     cout << "2) 看成绩" << endl;
53     cout << "3) 游戏讲解" << endl;
54     cout << "0) 退出" << endl;
55     cout << "=====" << endl;
56     cout << endl;
57     cout << "----> ";
58

```

```

59         cin >> selection;
60         return selection;
61     }
62
63     void UI::NewGame()
64     {
65         Score& s = Score::getInstance();
66         s.newGame();
67         int judge = 0;
68         while (judge == 0){
69             cout << "\n 请输入四位不重复的正整数" << endl;
70             int number;
71             cin >> number;
72             if (number <= 0)
73             {
74                 cout << " 结束本轮游戏" << endl;
75                 return;
76             }
77             if (UI::GuessNumber(number) == true){
78                 judge = 1;
79                 cout << " 恭喜你猜对了" << endl;
80             }
81         }
82     }
83
84     bool UI::GuessNumber(int number)
85     {
86         if (number == 8888){
87             UI::ViewDetail();
88             return false;
89         }
90         else if (number == 7777)
91         {
92             UI::ViewAnswer();
93             return false;
94         }
95
96         Score& s = Score::getInstance();
97         pair<bool, string> s1 = s.guess(number);
98         cout << s1.second;
99         UI::ShowScore();
100        return s1.first;
101    }
102
103     void UI::ViewDetail()
104     {
105         Score& s = Score::getInstance();
106         pair<bool, string> s0ther = s.guess(8888);
107         cout << s0ther.second;
108         cout << endl;
109     }
110
111     void UI::ViewAnswer()
112     {
113         Score& s = Score::getInstance();
114         cout << " 注：您有三次输入密码的机会，三次均不对则验证失败，然后返回上一层" << endl;
115         for (int i = 1; i <= 3; i++){
116             if (UI::InputPassword() == true){
117                 cout << "\n 密码输入正确" << endl;
118                 cout << s.guess(7777).second << endl;
119                 return;
120             }
121             else
122                 cout << "\n 密码输入错误" << endl;
123         }
124         cout << " 验证失败，返回上一层" << endl;
125     }
126
127     bool UI::InputPassword()
128     {
129         Score& s = Score::getInstance();
130         char *words = getpass(" 请输入查看答案所需的密码：");
131         string word(words);
132         return s.checkPassword(word);
133     }
134
135     void UI::ShowScore()
136     {
137         Score& s = Score::getInstance();
138         cout << " 您的得分是" << endl;
139         cout << s.getScore() << endl;
140     }
141
142     void UI::ReadHelp()
143     {
144         cls();
145         cout << "\n\n\n\n" << endl;
146         cout << " 游戏目的是猜一个四位数，且这个四位数每位都不相同1." << endl;
147         cout << " 每次猜数的提示分别表示数字、位置都匹配的个数，数字匹配但位置不匹配的个数 2.\n" << endl;

```

```

148         << " 猜对了加3. 20 分, 猜错了减 40 分" << endl;
149         << " 猜4. 8888 可以得到详细的提示" << endl;
150         << " 猜5. 7777 可以直接看答案, 但需要密码" << endl;
151         << " 猜6. 0 或负数会退出该轮游戏, 但仍会计分哦" << endl;
152         << " " << endl;
153     }
154 }

```

3.6 田劲锋的实现

Listing 10: Number_tjf.cpp

```

1  /* Number_tjf.cpp
2  * 描述: 数字类的实现
3  * 作者: 田劲锋
4  * 创建时间: 2015-1-15
5  * 修改时间: 2015-1-15
6  */
7  #include <cstdlib>
8  #include <iostream>
9  #include <vector>
10 #include <algorithm>
11 using namespace std;
12
13 #include "Number.h"
14
15 Number::Number()
16 {
17     this->setNumber(Number::genRand());
18     this->count = 0;
19 }
20
21 Number::~Number()
22 {}
23
24 int Number::genRand()
25 {
26     int s[4];
27     vector<int> t;
28     for (int i = 1; i <= 9; i++)
29         t.push_back(i);
30     int k = rand() % t.size();
31     s[0] = t[k];
32     t.erase(t.begin() + k);
33     t.push_back(0);
34     for (int i = 1; i <= 3; i++) {
35         k = rand() % t.size();
36         s[i] = t[k];
37         t.erase(t.begin() + k);
38     }
39     int n = s[0] * 1000 + s[1] * 100 + s[2] * 10 + s[3];
40     return n;
41 }
42
43 vector<int> itov(int n)
44 {
45     vector<int> v(4);
46     v[0] = n / 1000;
47     v[1] = n % 1000 / 100;
48     v[2] = n % 100 / 10;
49     v[3] = n % 10;
50     return v;
51 }
52
53 void Number::setNumber(int number)
54 {
55     this->number = number;
56     vector<int> v = itov(number);
57     for (int i = 0; i < 4; i++) {
58         this->numbers[i] = v[i];
59     }
60 }
61
62 pair<int, int> Number::guess(int number)
63 {
64     int x = 0, y = 0;
65     vector<int> s = itov(this->number);
66     vector<int> a = itov(number);
67     int u[10] = { 0 };
68     int v[10] = { 0 };
69     for (int i = 0; i < 4; i++) {
70         x += (a[i] == s[i]);

```

```

71         u[a[i]]++;
72         v[s[i]]++;
73     }
74     for (int i = 0; i < 10; i++) {
75         y += (u[i] && v[i]);
76     }
77     y = y - x;
78     this->count++;
79     return make_pair(x, y);
80 }
81
82 vector<int> Number::detail(int number)
83 {
84     vector<int> v;
85     int *s = this->numbers;
86     vector<int> a = itov(number);
87     for (int i = 0; i < 4; i++) {
88         if (a[i] == s[i]) {
89             v.push_back(i + 1);
90         }
91     }
92     this->count++;
93     return v;
94 }
95
96 int Number::answer()
97 {
98     this->count++;
99     return this->number;
100 }

```

Listing 11: Score_tjf.cpp

```

1  /* Score_tjf.cpp
2  * 描述: 得分类的实现
3  * 作者: 田劲锋
4  * 创建时间: 2015-1-15
5  * 修改时间: 2015-1-15
6  */
7
8  #include <iostream>
9  #include <fstream>
10 #include <sstream>
11 #include <string>
12 #include <vector>
13 #include <algorithm>
14 #include <cstdlib>
15 #include <ctime>
16 using namespace std;
17
18 #include "Score.h"
19
20 int Score::getScore() const
21 {
22     return this->score;
23 }
24
25 Score::Score()
26 {
27     srand((unsigned int) time(NULL));
28     this->score = 0;
29     this->password = "root";
30     this->lastNumber = -1;
31     this->read();
32 }
33
34 Score& Score::getInstance()
35 {
36     static Score instance;
37     return instance;
38 }
39
40 Score::~Score()
41 {
42     this->write();
43 }
44
45 void Score::newGame()
46 {
47     Number n;
48     this->numbers.push_back(n);
49 }
50
51 pair<bool, string> Score::guess(int number)
52 {
53     Number &n = this->numbers.back();
54     if (number == 8888) {

```

```

55         if (this->lastNumber == -1) {
56             return make_pair(false, "还没有猜过任何数");
57         }
58         vector<int> a = n.detail(this->lastNumber);
59         ostream s;
60         s << "第 ";
61         for (size_t i = 0; i < a.size() - 1; i++) {
62             s << (i + 1) << ", ";
63         }
64         s << a.back() << " 位数字正确";
65         return make_pair(false, s.str());
66     } else if (number == 7777) {
67         int a = n.answer();
68         ostream s;
69         s << "正确答案是 " << a;
70         return make_pair(false, s.str());
71     } else {
72         pair<int, int> a = n.guess(this->lastNumber = number);
73         int x = a.first, y = a.second;
74         ostream s;
75         if (x == 4 && y == 0) {
76             s << "猜中了, 答案就是 " << number;
77             plus();
78             return make_pair(true, s.str());
79         } else {
80             s << "数位匹配 " << x << " 个, 数位匹配不符 " << y << " 个";
81             minus();
82             return make_pair(false, s.str());
83         }
84     }
85 }
86
87 void Score::plus()
88 {
89     this->score += Score::PLUS;
90 }
91
92 void Score::minus()
93 {
94     this->score -= Score::MINUS;
95 }
96
97 const char* score_filename = "score.dat";
98
99 int Score::read()
100 {
101     ifstream fin(score_filename);
102     if (fin.good()) {
103         fin >> this->password;
104         fin >> this->score;
105     } else {
106         this->write();
107     }
108     return this->score;
109 }
110
111 void Score::write()
112 {
113     ofstream fout(score_filename);
114     if (fout.good()) {
115         fout << this->password << endl;
116         fout << this->score << endl;
117     }
118 }
119
120 bool Score::checkPassword(string password)
121 {
122     return password == this->password;
123 }

```

Listing 12: UI_tjf.cpp

```

1  /* UI_tjf.cpp
2  * 描述: 用户界面类的实现
3  * 作者: 田劲锋
4  * 创建时间: 2015-1-15
5  * 修改时间: 2015-1-15
6  */
7
8  #include <iostream>
9  #include <cstdlib>
10 #include <string>
11 #include <algorithm>
12 using namespace std;
13
14 #include "UI.h"
15 #include "mylib.h"

```

```

16
17 void UI::Main()
18 {
19     int n = 1;
20     while (n) {
21         n = UI::MainMenu();
22         if (n == 1) {
23             UI::NewGame();
24         } else if (n == 2) {
25             UI::ShowScore();
26         } else if (n == 3) {
27             UI::ReadHelp();
28         } else {
29             break;
30         }
31         pause();
32     }
33 }
34
35 int geti()
36 {
37     cin.clear();
38     string str;
39     getline(cin, str);
40     int n = atoi(str.c_str());
41     return n;
42 }
43
44 int UI::MainMenu()
45 {
46     cls();
47
48     cout << endl;
49     cout << " 猜数字游戏 " << endl;
50     cout << "-----" << endl;
51     cout << "1) 新游戏 " << endl;
52     cout << "2) 查成绩 " << endl;
53     cout << "3) 看帮助 " << endl;
54     cout << "0) 退出 " << endl;
55     cout << endl;
56     cout << "----> ";
57
58     int n = geti();
59     return n;
60 }
61
62 void UI::ReadHelp()
63 {
64     cout << endl
65     << " 猜数字游戏帮助 " << endl
66     << "-----" << endl
67     << "1. 游戏目的是猜一个四位数, 且这个四位数每位都不相同 " << endl
68     << "2. 每次猜数都会有提示 " << endl
69     << " 分别表示数字、位置都匹配的个数, 数字匹配但位置不匹配的个数 " << endl
70     << "3. 猜对了加 20 分, 猜错了减 40 分 " << endl
71     << "4. 猜 8888 可以得到详细的提示 " << endl
72     << "5. 猜 7777 可以直接看答案, 但需要密码 " << endl
73     << "6. 猜 0 或负数会退出该轮游戏, 但仍会计分哦 " << endl
74     << endl
75     << "**** 享受你的游戏! ****" << endl;
76 }
77
78 void UI::NewGame()
79 {
80     Score& s = Score::getInstance();
81     s.newGame();
82     bool flag = false;
83     while (!flag) {
84         cout << "请猜一个四位数: " << endl;
85         cout << "==" << endl;
86         int n = geti();
87         if (n <= 0) {
88             cout << "结束本轮游戏! " << endl;
89             return;
90         }
91         flag = UI::GuessNumber(n);
92     }
93     cout << "恭喜你猜对了! " << endl;
94 }
95
96 bool UI::GuessNumber(int n)
97 {
98     if (n == 8888) {
99         UI::ViewDetail();
100         return false;
101     } else if (n == 7777) {
102         UI::ViewAnswer();
103         return false;
104     }

```



```

105     cout << "你猜的是: " << n << endl;
106     Score& s = Score::getInstance();
107     pair<bool, string> a = s.guess(n);
108     cout << a.second << endl;
109     UI::ShowScore();
110     return a.first;
111 }
112
113 void UI::ViewDetail()
114 {
115     Score& s = Score::getInstance();
116     cout << s.guess(8888).second << endl;
117 }
118
119 void UI::ViewAnswer()
120 {
121     for (int i = 0; i < 3; i++) {
122         if (UI::InputPassword()) {
123             Score& s = Score::getInstance();
124             cout << s.guess(7777).second << endl;
125             return;
126         } else {
127             cout << "密码错误! " << endl;
128         }
129     }
130 }
131
132 bool UI::InputPassword()
133 {
134     Score& s = Score::getInstance();
135     string t("请输入密码: ");
136     string password(getpass((char *) t.c_str()));
137     cout << endl;
138     return s.checkPassword(password);
139 }
140
141 void UI::ShowScore()
142 {
143     Score& s = Score::getInstance();
144     cout << "当前得分: " << s.getScore() << endl;
145 }

```

3.7 辅助文件

这个是用 C 语言一些跨平台控制台控制的代码，提供在这里供调用。

Listing 13: mylib.h

```

1  /* mylib.h
2  * 描述：提供一些跨平台的控制台控制函数
3  * 作者：田劲锋
4  * 创建时间：2015-1-12
5  * 修改时间：2015-1-14
6  */
7  #pragma once
8
9  char *getpass(char *prompt);
10 // 输入密码
11
12 void cls();
13 // 清屏
14
15 void pause();
16 // 暂停

```

Listing 14: mylib.cpp

```

1  /* mylib.cpp
2  * 描述：提供密码的输入功能
3  * 作者：田劲锋
4  * 创建时间：2015-1-12
5  * 修改时间：2015-1-15
6  */
7
8  #include <stdio.h>
9  #include <stdlib.h>
10 #include <string.h>
11
12 #include "mylib.h"

```

```

13
14 #ifdef WIN32
15 #include <conio.h>
16 #else
17 #include <termios.h>
18 #endif
19
20 #define PWD_MAX 256
21
22 char *getpass(char *prompt)
23 {
24     static char passwd[PWD_MAX] = "";
25     printf("%s", prompt);
26 #ifdef WIN32
27     int i = 0;
28     char c;
29     while ((c = _getch()) != '\n' && c != '\r' && i < PWD_MAX) {
30         if (c == '\b' && i > 0) {
31             passwd[i--] = '\0';
32             printf("\b \b");
33         } else {
34             passwd[i++] = c;
35             printf("*");
36         }
37     }
38     passwd[i] = '\0';
39 #else
40     struct termios oldflags, newflags;
41     tcgetattr(fileno(stdin), &oldflags);
42     newflags = oldflags;
43     newflags.c_lflag &= ~ECHO;
44     newflags.c_lflag |= ECHONL;
45     if (tcsetattr(fileno(stdin), TCSANOW, &newflags) != 0) {
46         perror("tcsetattr");
47         return NULL;
48     }
49     fgets(passwd, PWD_MAX, stdin);
50     if (tcsetattr(fileno(stdin), TCSANOW, &oldflags) != 0) {
51         perror("tcsetattr");
52         return NULL;
53     }
54 #endif
55     size_t l = strlen(passwd) - 1;
56     while (passwd[l] == '\r' || passwd[l] == '\n') {
57         passwd[l--] = '\0';
58     }
59     return passwd;
60 }
61
62 void cls()
63 {
64 #ifdef WIN32
65     system("cls");
66 #else
67     system("clear");
68 #endif
69 }
70
71 void pause()
72 {
73 #ifdef WIN32
74     system("pause");
75 #else
76     system("read -p 请按任意键继续\$. . .\");
77 #endif
78 }

```

4 运行结果与分析

5 总结

贡献	权值	田劲锋	邢志鹏	王增辉
文档	50%	252 行 (39.6%)	384 行 (60.4%)	0 行 (0%)
代码	50%	行 (%)	行 (%)	行 (%)
合计				

表 1: 小组贡献比例表

5.1 田劲锋的总结

5.2 邢志鹏的总结

5.3 王增辉的总结

A 股票交易系统

这个附录是股票交易系统的设计文档。当初最开始的选题是难度系数为 1.2 的股票交易系统，着手设计了类和头文件之后，发现代码量有点大，不是同组成员可以一周之内可以完成的，于是抛弃了这个项目，改换了代码量较小的猜数字游戏。为了使劳动成果不至于白白浪费，就在这里把设计好的文档和代码一并放在课程设计的附录之中，以供参考。

A.1 设计类图

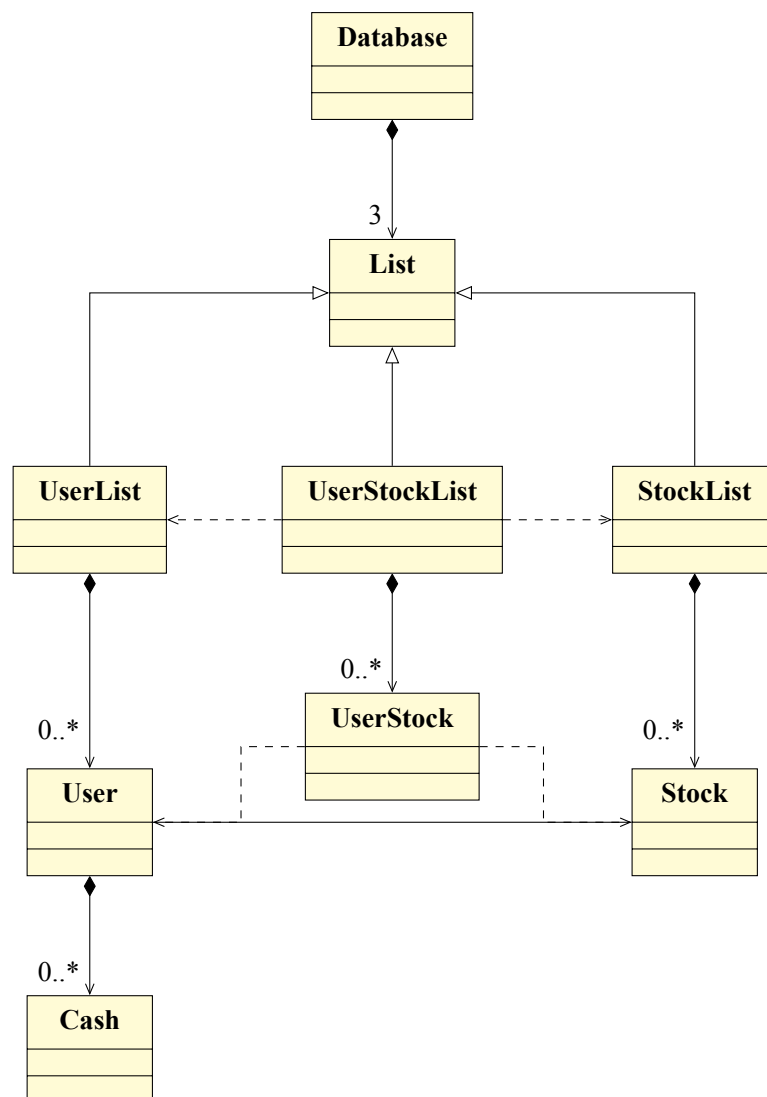


图 8: 股票交易系统各个类之间的关系

A.2 题目内容

(1) 修改数据结构，增加现金成员，每只股票增加牌价；每个用户的数据库中同样也增加现金数目的成员。

(2) 增加股票交易系统的接口程序，新增如下设计：

AddNewStock() 增加新股票

DeleteOldStock() 删除旧股票

HangUpStock() 挂起股票，停止交易

ModifyStock() 修改股票的名称、代码

以上修改均须输入密码，密码吻合后才能进入数据库进行修改，结果均存入 Stock_File.dat 中。

(3) 将股票数据的处理由数组改为链表，可以处理多只股票的交易，链表以交易代码的序号进行排序，也可根据需要对股票的牌价进行排序。(1.2)

A.3 头文件

```
1  /* Database.h
2  * 描述：数据库类，用于读取存储数据表
3  * 作者：田劲锋
4  * 创建时间：年月日20151111
5  * 修改时间：年月日20151111
6  */
7  #pragma once
8
9  #include <iostream>
10 #include <vector>
11 #include <string>
12 using namespace std;
13
14 #include "List.h"
15
16 class Database
17 {
18 public:
19     Database(string filename);
20     virtual ~Database();
21
22     void open();
23     void write();
24     void close();
25
26     List& getTable(string tablename) const;
27     List& getTable(int index) const;
28
29 protected:
30     string filename;
31     istream file;
32     vector<List> db;
33 };
```

```
1  /* List.h
2  * 描述：数据表类，用于描述数据库表
3  * 作者：田劲锋
4  * 创建时间：年月日20151111
5  * 修改时间：年月日20151111
6  */
7  #pragma once
8
9  #include <iostream>
10 #include <vector>
11 #include <string>
12 using namespace std;
13
```

```

14 class List
15 {
16 public:
17     List(string tablename);
18     virtual ~List();
19
20     void setTablenNme(string);
21     string getTableName() const;
22
23     int newId();
24
25     virtual int getTotal();
26
27 protected:
28     int max_id; // 标识表中最大的 ID
29     string tablename; // 表名
30     int total; // 表中元素数量
31 };

```

```

1  /* UserList.h
2  * 描述: 用户表类
3  * 作者: 田劲锋
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include <iostream>
10 #include <vector>
11 #include <string>
12 using namespace std;
13
14 #include "List.h"
15 #include "User.h"
16
17 class UserList : public List
18 {
19 public:
20     UserList(string tablename);
21     virtual ~UserList();
22
23     int getTotal();
24
25     User& getUser(int index) const;
26
27     User& findUser(string keyword, int startIndex = 0) const;
28     int findUserIndex(string keyword, int startIndex = 0) const;
29
30     int insert(const User&);
31     void remove(int index);
32     void remove(int startIndex, int number);
33
34     void sort(bool asc = true);
35
36 protected:
37     vector<User> users;
38 };

```

```

1  /* StockList.h
2  * 描述: 股票表类
3  * 作者: 田劲锋
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include <iostream>
10 #include <vector>
11 #include <string>
12 using namespace std;
13
14 #include "List.h"
15 #include "Stock.h"
16
17 class StockList : public List
18 {
19 public:
20     StockList(string tablename);
21     virtual ~StockList();
22
23     int getTotal();
24
25     Stock& getStock(int index) const;
26
27     Stock& findStock(string keyword, int startIndex = 0) const;
28     int findStockIndex(string keyword, int startIndex = 0) const;

```

```

29
30     int insert(const Stock&);
31     void remove(int index);
32     void remove(int startIndex, int number);
33
34     void sort(bool asc = true);
35     void sortByCode(bool asc = true);
36     void sortByPrice(bool asc = true);
37     void sortByTotal(bool asc = true);
38
39 protected:
40     vector<Stock&> Stocks;
41 };

```

```

1  /* UserStockList.h
2  * 描述: 用户股票表类
3  * 作者: 田劲锋
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include <iostream>
10 #include <vector>
11 #include <string>
12 using namespace std;
13
14 #include "List.h"
15 #include "UserStock.h"
16
17 class UserStockList : public List
18 {
19 public:
20     UserStockList(string tablename);
21     virtual ~UserStockList();
22
23     int getTotal();
24
25     UserStock& getUserStock(int index) const;
26
27     UserStock& findUserStock(string keyword, int startIndex = 0) const;
28     int findUserStockIndex(string keyword, int startIndex = 0) const;
29
30     int insert(const UserStock&);
31     void remove(int index);
32     void remove(int startIndex, int number);
33
34     void sort(bool asc = true);
35
36 protected:
37     vector<UserStock&> us;
38 };

```

```

1  /* User.h
2  * 描述: 用户类
3  * 作者: 田劲锋
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include <vector>
10 #include <string>
11 using namespace std;
12
13 #include "Cash.h"
14 #include "Stock.h"
15 #include "UserStock.h"
16
17 class User
18 {
19     friend class Stock;
20     friend class UserStock;
21 public:
22     User();
23     virtual ~User();
24
25     void setUsername(string);
26     string getUsername() const;
27
28     void setPassword(string);
29     bool checkPassword(string) const;
30
31     void setAdmin(bool);
32     bool isAdmin() const;
33

```



```

34     bool operator==(const User&) const;
35     bool operator<(const User&) const;
36
37     User(const User&);
38     User& operator=(const User&);
39
40     friend ostream& operator<<(ostream&, const User&);
41     friend istream& operator>>(istream&, const User&);
42
43 protected:
44     int id; // 唯一标识元素的 ID
45     string username;
46     string password;
47     double money; // 用户手头的钱
48     bool admin; // 是否为管理员
49     vector<Cash> cashes; // 用户持有的股票
50     double amount; // 用户持有股票的总金额
51 };

```

```

1  /* Stock.h
2  * 描述: 股票类
3  * 作者: 田劲峰
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include <vector>
10 #include <string>
11 using namespace std;
12
13 #include "User.h"
14 #include "UserStock.h"
15
16 class Stock
17 {
18     friend class User;
19     friend class UserStock;
20 public:
21     Stock();
22     Stock(int code, double price, string name = "");
23     virtual ~Stock();
24
25     void setCode(int);
26     void setCode(string);
27     string getCode() const;
28
29     void setName(string);
30     string getName() const;
31
32     void setPrice(double);
33     double getPrice() const;
34
35     void addTotal(int);
36     int getTotal() const;
37
38     friend ostream& operator<<(ostream&, const Stock&);
39     friend istream& operator>>(istream&, const Stock&);
40
41 protected:
42     int id; // 唯一标识元素的 ID
43     string code; // 交易代码
44     string name;
45     bool valid; // 是否可交易 / 挂起
46     double price; // 股价
47     int total; // 股票数量
48     vector<User&> users; // 持有该股票的用户
49 };

```

```

1  /* UserStock.h
2  * 描述: 用户股票类, 存储
3  * 作者: 田劲峰
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include <vector>
10 #include <string>
11 using namespace std;
12
13 #include "User.h"
14 #include "Stock.h"
15
16 class UserStock
17 {

```

```

18     friend class User;
19     friend class Stock;
20 public:
21     UserStock(const User&, const Stock&, int num = 0);
22     virtual ~UserStock();
23
24     void set(const UserStock&);
25     UserStock& get() const;
26
27     bool operator==(const UserStock&) const;
28     bool operator<(const UserStock&) const;
29
30     UserStock& operator=(const UserStock&);
31
32     friend ostream& operator<<(ostream&, const UserStock&);
33     friend istream& operator>>(istream&, const UserStock&);
34
35 protected:
36     int id; // 唯一标识元素的 ID
37     int userId;
38     int StockId;
39     int num; // 股票数量
40     double price; // 购入价 / 卖出价 (负)
41     time_t timestamp; // 交易时间
42 private:
43     User& user;
44     Stock& stock;
45 };

```

```

1  /* Cash.h
2  * 描述: 现金类, 描述有几股该股票
3  * 作者: 田劲锋
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include "Stock.h"
10
11 class Cash
12 {
13 public:
14     Cash(const Stock&, int = 0);
15     virtual ~Cash();
16
17 protected:
18     Stock& stock;
19     int num; // 股数
20     double amount; // 总价值
21 };

```

```

1  /* UI.h
2  * 描述: 用户界面类, 用于用户交互
3  * 作者: 田劲锋
4  * 创建时间: 年月日2015111
5  * 修改时间: 年月日2015111
6  */
7  #pragma once
8
9  #include <vector>
10 #include <string>
11 using namespace std;
12
13 #include "Database.h"
14 #include "User.h"
15
16 class UI
17 {
18 public:
19     UI();
20     virtual ~UI();
21
22     void MainMenu();
23
24     void UserMenu();
25     void RegistUser(); // 注册新用户
26     void LoginUser(); // 用户登录
27
28     void StockMenu();
29     void ListStock(); // 显示股票列表
30     void ListStockByCode(); // 按交易代码顺序显示股票列表
31     void ListStockByPrice(); // 按价格顺序显示股票列表
32     void ListStockByTotal();
33     void BuyStock(); // 买入股票
34     void SaleStock(); // 卖出股票
35

```

```
36     void AdminMenu();
37     void AddNewStock(); // 增加新股票
38     void DeleteOldStock(); // 删除旧股票
39     void HangUpStock(); // 挂起股票, 停止交易
40     void ModifyStock(); // 修改股票的名称、代码
41
42 protected:
43     Database& db;
44     User& user;
45 };
```

```
1 #include <iostream>
2
3 #include "TestCase.h"
4
5 int main(int argc, char *argv[])
6 {
7     return 0;
8 }
```

参考文献

- [1] H. M. Deitel, P. J. Deitel. C++ 大学基础教程. 5 edn. 北京: 电子工业出版社, 2014
- [2] M. Gregoire, N. A. Solter, S. J. Kleper. C++ 高级编程. 2 edn. 北京: 清华大学出版社, 2012
- [3] I. Horton. Visual C++ 2012 入门经典. 6 edn. 北京: 清华大学出版社, 2013
- [4] A. Shalloway, J. R. Trott. 设计模式解析. 2 edn. 北京: 人民邮电出版社, 2013
- [5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, et al. Introduction to Algorithms. 3 edn. London, England: The MIT Press, 2009
- [6] D. E. Knuth. The Art Of Computer Programming. Pearson Education, 1968–2011
- [7] 邓建松, 彭冉冉, 陈长松. L^AT_EX 2_ε 科技排版指南. 北京: 科学出版社, 2001