河南工業大學

课　程　报　告

课 程 名 称：面向对象程序设计

专 业 班 级：软件 1305 班

学 生 姓 名：田劲锋

学　　号：201316920311

任 课 教 师：

学　　期：2014–2015 学年第一学期

目录

1　题目内容及设计要求

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)

2　总体方案设计

2.1　总体功能框图

图 1 展示了各个类之间的关系。

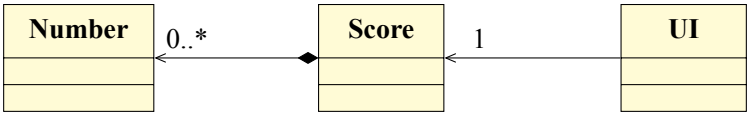

图 1: 各个类之间的关系

2.2 类的设计说明

2.2.1 Number 类

Number
number: *int* //待猜的数 – numbers: *short[]* //数的拆分数组 # count: *int* //猜数次数

+ Number() //初始化产生随机数
– genRand(): *int* //产生符合要求的随机数
+ guess(number: *int*): x*: int,* y*: int* //猜数并返回提示信息
+ detail(number: *int*): *int[]* //猜数并返回正确的位数组
+ answer(): *int* //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，这两个是一对一的关系，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。

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。

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→numbers。每调用该方法时 this→count 应该自增 1。

2.2.2　Score 类

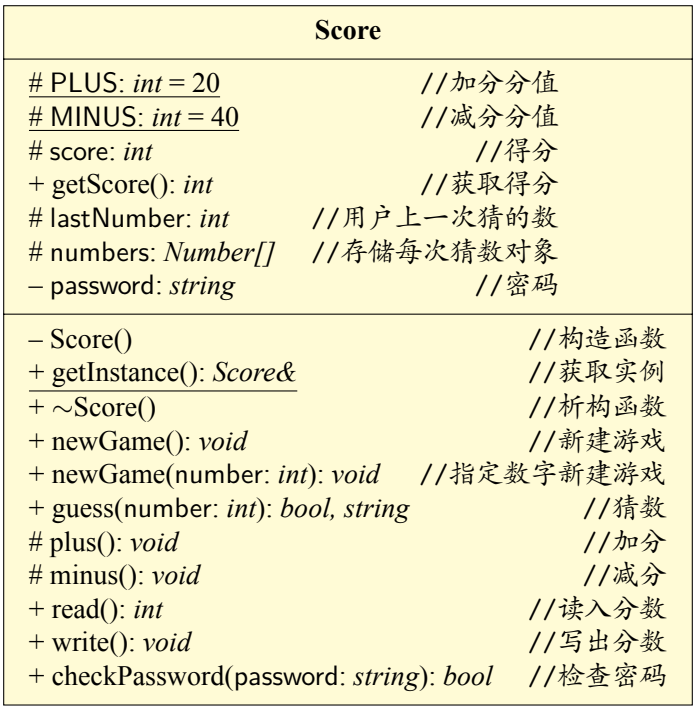

图 3: Score 类

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。

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::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::newGame

```
void Score::newGame(int number);
```

返回 无

描述 以指定的 number 为参数，创建一个新的 Number 实例，并添加到 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”；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。

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。

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

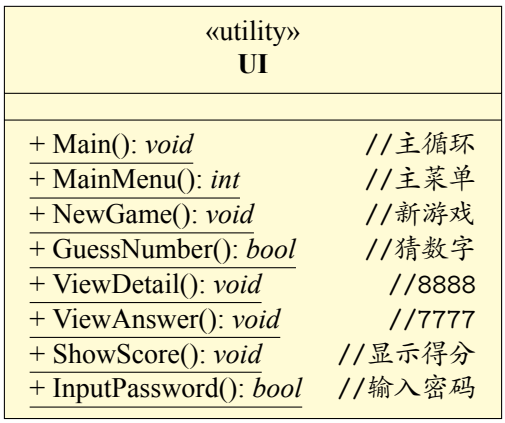

图 4: UI 类

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；(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，调用 UI::GuessNumber() 进入猜数流程。循环调用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 等待用户输入。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，如果是 0000 则返回上一级，如果是负数则直接退出程序。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。

2.3 主要算法流程图

NUMBER::GENRAND()

1 $S = \{\}$
2 $T = \{1, 2, \cdots, 9\}$
3 $S[0] = T.\text{REMOVE}(\text{RAND}(T.length))$
4 $T = T + \{0\}$
5 **for** $i = 1$ **to** 3
6 $\quad S[i] = T.\text{REMOVE}(\text{RAND}(T.length))$
7 $number = \overline{S_0 S_1 S_2 S_3}$
8 **return** $number$

图 5: 生成符合要求的随机数

3　程序清单及注释

4　运行结果与分析

5 总结

A　股票交易系统