Assignment 4

Problem 1:

The player profile of a chess game contains name, login times, played matches, won matches, won ratio, rank(beginner, professional, expert). Implement a class named User_Prof with following operations: login, logout, test equivalence, test not equivalence. Its constructor should provide default player's rank and default name "guest". There may be multiple "guest" users at one time, never to mix up their login sessions.

Problem 2:

Implement the class of a 4X4 float matrix, providing interface as follows:

- matrix add
- matrix multiply
- print
- operator +=
- subscripting function object
  - float& operator()(int row, int column);
  - float operator(int row, int column) const;
- Default constructor that accepts 16 values from standard device
- Another constructor that accept an array of size 16

Problem 3:

Create two classes called Traveler and Pager without default constructors, but with constructors that take an argument of type string, which they simply copy to an internal string variable. For each class, write the correct copy-constructor and assignment operator. Now inherit a class BusinessTraveler from Traveler and give it a member object of type Pager. Write the correct default constructor, a constructor that takes a string argument, a copy-constructor, and an assignment operator.

Problem 4:

Write a class with one virtual function and one non-virtual function. Inherit a new class, make an object of this class, and upcast to a pointer of the base-class type. Use the clock() function found in <ctime>(you'll need to look this up in your local C library guide) to measure the difference between a virtual call and non-virtual call. You'll need to make multiple calls to each function inside your timing loop to see the difference.

Problem 5:

Create a base class containing a clone() function that returns a pointer to a copy of the current object. Derive two subclasses that override clone() to return copies of their specific types. In main(), create and upcast objects of your two derived types, then call clone() for each and verify that the cloned copies are the correct subtypes. Experiment with your clone() function so that you return the base type, then try returning the exact derived type. Can you think of situations in which the latter approach is necessary?

Problem 6:

请使用泛型相关知识完成以下需求：

定义一个函数模版，计算出某些类型数据的最大值，如果取得的最大值的元素不唯一，则输出第一个取得最大值的元素。

```
Template<typename T>
T max(T t1, T t2, T t3){
    ……
}
```

需要计算的数据类型有：int, double, char, string, Rectangle 结构体（其中包含 x 和 y 两个整形变量，思考一下如何定义该结构体）, Rectangle*。

- int：例如 max(1，2，3)会输出 3，max(2，3，4)会输出 4
- double：与 int 类似，输出类型是 double（无需格式化）
- char：例如 max('a','b','c')会输出 c（ASCII 码大的更大）
- string：例如 max(string("abc")，string("abcd")，string("abcde"))，输出 abcde（长度长的更大）
- Rectangle：例如 max(Rectangle{1,2},Rectangle{2,6},Rectangle{3,4})，输出(2,6) (x,y 分别代表长方形的长和宽，面积大的更大，(2,6)是第一个取得最大值面积 12 的元素)
- Rectangle*：与 Rectangle 输出类型一样，需要思考如何传参以及如何格式化输出

输入：按照顺序输入上述类型的三个数据，样例

```
1 2 3
1.3 2.4 4.6
a b d
csd swe sfdfg
1,2 3,4 2,6
1,2 2,6 3,4
```

第 5 行的 1,2 3,4 2,6 是指三个结构体变量：(1,2)、(3,4)、(2,6)，思考如何输入。

输出：对应输入每一行输出一行结果（可以输入完一行，回车后直接输出，也可以数据全部输入完之后输出），对应样例

```
3
4.6
d
sfdfg
3,4
2,6
```