HW6

181220076 周韧哲

一. 概念题

1. 作用是使protected类成员不能够从类外部访问。而在基类中，其与private的成员可看作具有相同的访问限制，即只能被该基类的成员函数访问。另外，通过public继承得到的子类，其成员函数能够直接访问基类的protected成员。
2. 普通派生类对象在创建时会先自动调用基类构造函数，然后调用自身构造函数；销毁时会先自动调用自身的析构函数，然后调用基类的析构函数。因为要先初始化基类成员，派生类对象里可能会有基类成员的调用，析构也同理。

二. 编程题

1. 不适合用public继承，Square子类中的成员函数set_side和get_side其实就调用了父类的public成员函数，用public继承，将这些父类成员函数暴露给用户在子类中调用可能会出现使用混乱的情况，不符合OOP设计思想，容易造成bug。所以最好用private和protected继承。

2.
```
enum TimeZone{
    W12 = -12, W11, W10, W9, W8, W7, W6, W5, W4, W3,  W2,  W1,
    GMT,  E1,  E2,  E3,  E4, E5, E6, E7, E8, E9, E10, E11, E12};

#define SET(h, m, s) do{ \
              this->h = h; \
              this->m = m; \
              this->s = s;}while(0)

class Time{
    int h,m,s;
public:
    Time(){}
    Time(int h, int m, int s){
       SET(h, m, s);
    }
    void set(int h, int m, int s){
       SET(h, m ,s);
    }
    void display(){
        printf("%02d:%02d:%02d.\n",h,m,s);
    }
    int timestamp() const{
        return h*3600 + m*60 + s;
    }
    bool equal(Time &other_time){
        return this->timestamp() == other_time.timestamp();
    }
    bool less_then(Time &other_time){
        return this->timestamp() < other_time.timestamp();
    }
    void increment(){
        if(s<59)  s++;
        else{
            s = 0;
```

```
            if(m<59) m++;
            else{
                m = 0;
                if(h<23) h++;
                else{
                    h = 0;
                }
            }
        }
    }
};

class ExtTime:private Time{
    TimeZone tz;
public:
    ExtTime(){
        Time::set(0, 0, 0);
        tz = GMT;
    }
    ExtTime(int h, int m, int s, TimeZone t){
        Time::set(h, m, s);
        tz = t;
    }
    void display(){
        cout<<"TimeZome ";
        if(tz < 0) cout<<"W"<<-tz<<": ";
        else if(tz == 0) cout<<"GMT"<<": ";
        else cout<<"E"<<tz<<": ";
        Time::display();
    }
    bool equal(const ExtTime &other_time){
        return timestamp() == other_time.timestamp() + (tz -
other_time.tz)*3600;
    }
    bool less_than(const ExtTime &other_time){
        return timestamp() < other_time.timestamp() + (tz -
other_time.tz)*3600;
    }
};
```

此题也可用public继承。公有继承的特点是基类的公有成员和保护成员作为派生类的成员时，它们都保持原有的状态，而基类的私有成员仍然是私有的，不能被这个派生类的子类所访问。所以需要在外部使用基类共有成员函数时需要用public继承。由于private继承将基类的所有public都改为private，因此，可以将private继承视为继承子类的实现而略去子类的接口。同理，protected继承使得子类可以使用protected的成员，在子类以及派生类需要使用时用protected继承。