Object Oriented Programming

Shaobai Kan

chapter 10

Outline

Outline

- Const (constant) objects and const member functions

Outline

- Const (constant) objects and const member functions
- *friend* Functions and *friend* Classes

Const (constant) objects and const member functions

Constant object

Fact. Some objects need to be modifiable and some do not. You may use keyword const to specify that an object is not modifiable and that any attempt to modify the object should result in a compilation error.

Example.

```
class Time
{
 :....
};
const Time noon (12, 0, 0);
```

Constant function

Fact. . C++ disallows member function calls for const objects unless the member functions themselves are also declared const.

- Defining as const a member function that modifies a data member of the object is a compilation error.
- Defining as const a member function that calls a non-const member function of the class on the same object is a compilation error.
- Invoking a non-const member function on a const object is a compilation error.

Example: Time

Time.h

```
class Time
{
 public:
        Time( int = 0, int = 0, int = 0);

        void setTime(int, int, int);
        void setHour ( int );
        void setMinute ( int );
        void setSecond ( int );

        int getHour () const;
        int getMinute () const;
        int getSecond () const;

        void printUniversal() const;
        void printStandard();

  private:
        int hour;
        int minute;
        int second;
  };
```

Example: Time

Time.cpp

```
#include <iostream>
#include <iomanip>
#include “Time.h”
using namespace std;

Time::Time( int hour, int minute, int second)
{
        setTime(hour, minute, second);
}

void Time::setTime(int hour, int minute, int second)
{
        setHour ( hour );
        setMinute ( minute );
        setSecond ( second );
}

void Time::setHour (int h)
{
        hour = ( h >= 0 && h < 24 ) ? h : 0;
}
```

Example: Time

Time.cpp

```
void Time::setMinute( int m)
{
      minute = ( m >= 0 && m < 60) ? m : 0;
}

void Time::setSecond( int s )
{
      second = ( s>= 0 && s < 60 ) ? s : 0;
}


int Time::getHour( ) const
{
       return minute;
}


int Time::getSecond() const
{
       return second;
}
```

Example: Time

Time.cpp

```
int Time::getMinute() const
{
    return minute;
}


void Time::printUniversal() const
{
        cout << setfill('0') << setw(2) << hour << ":"
         << setw(2) << minute << ":" << setw(2) << second;
}


void Time::printStandard()
{
    cout << ( ( hour == 0 || hour == 12 ) ? 12 : hour % 12) << ":"
        << setfill('0') << setw(2) << minute << ":" << setw(2)
        << second << ( hour < 12 ? " AM" : " PM");
}
```

Example: Time

Driver program

```
#include <iostream>
#include "Time.h"
using namespace std;

int main()
{
    Time wakeUp(6, 45, 0);
    const Time noon(12, 0 ,0);


    wakeUp.setHour(18);

//  noon.setHour(12);

    wakeUp.getHour( );

    noon.getMinute( );

    noon.printUniversal();

//  noon.printStandard();

}
```

friend Functions and friend Classes

friend function

Definition. A *friend function* of a class is defined outside that class's scope, yet has the right to access the non-public (and public) members of the class.

friend function

Definition. A *friend function* of a class is defined outside that class's scope, yet has the right to access the non-public (and public) members of the class.

Fact. Standalone functions, entire classes or member functions of other classes may be declared to be friends of another class.

Example: friend function

```
# include <iostream>
using namespace std;

class Count
{
     friend void setX(Count &, int);    //friend declaration

public:
     Count()
          : x(0)
     {

     }

     void print() const
     {
          cout << x << endl;
     }
private:
     int x;
};
```

Example: friend function

```
void setX (Count &c, int val)
{
        c.x = val;
}


int main()
{
        Count counter;

        cout << “counter.x after instantiation: ” ;
        counter.print();

        setX(counter, 8);

        cout << "counter.x after call to setX friend function: ";
        counter.print();

 }
```

Example: friend function

```
counter.x after instantiation: 0
counter.x after call to setX friend function: 8
```

Homework:

- Read Sec. 10.1-10.4