CISC181 Introduction to Computer Science

Dr. McCoy

Lecture 20
November 10, 2009

6.11 Using Default Arguments with Constructors

- **Constructors**
  - Can specify default arguments
  - Default constructors
    - Defaults all arguments
      OR
    - Explicitly requires no arguments
    - Can be invoked with no arguments
    - Only one per class

Outline

time2.h (1 of 1)

```
1  // Fig. 6.15: time2.h
2  // Declaration of class Time.
3  // Member functions defined in time2.cpp.
4
5  // prevent multiple inclusions of header file
6  #ifndef TIME2_H
7  #define TIME2_H
8
9  // Time abstract data type definition
10 class Time {
11
12 public:
13    Time( int = 0, int = 0, int = 0 ); // default constructor
14    void setTime( int, int, int ); // set hour, minute, second
15    void printUniversal(); // print universal-time format
16    void printStandard(); // print standard-time format
17
18 private:
19    int hour;     // 0 - 23 (24-hour clock format)
20    int minute;   // 0 - 59
21    int second;   // 0 - 59
22
23 }; // end class Time
24
25 #endif
```

Default constructor specifying all arguments.

Outline

time2.cpp (1 of 2)

```
1  // Fig. 6.16: time2.cpp
2  // Member-function definitions for class Time.
3  #include <iostream>
4
5  using std::cout;
6
7  #include <iomanip>
8
9  using std::setfill;
10 using std::setw;
11
12 // include definition of class Time from time2.h
13 #include "time2.h"
14
15 // Time constructor initializes each data member to zero;
16 // ensures all Time objects start in a consistent state
17 Time::Time( int hr, int min, int sec )
18 {
19    setTime( hr, min, sec );  // validate and set time
20
21 } // end Time constructor
22
```

Constructor calls **setTime** to validate passed (or default) values.

Outline

time2.cpp (2 of 2)

```
23 // set new Time value using universal time, perform validity
24 // checks on the data values and set invalid values to zero
25 void Time::setTime( int h, int m, int s )
26 {
27    hour = ( h >= 0 && h < 24 ) ? h : 0;
28    minute = ( m >= 0 && m < 60 ) ? m : 0;
29    second = ( s >= 0 && s < 60 ) ? s : 0;
30
31 } // end function setTime
32
33 // print Time in universal format
34 void Time::printUniversal()
35 {
36    cout << setfill( '0' ) << setw( 2 ) << hour << ":"
37         << setw( 2 ) << minute << ":"
38         << setw( 2 ) << second;
39
40 } // end function printUniversal
41
```


Outline

time2.cpp (2 of 2)

```
42 // print Time in standard format
43 void Time::printStandard()
44 {
45    cout << ( ( hour == 0 || hour == 12 ) ? 12 : hour % 12 )
46         << ":" << setfill( '0' ) << setw( 2 ) << minute
47         << ":" << setw( 2 ) << second
48         << ( hour < 12 ? " AM" : " PM" );
49
50 } // end function printStandard
```

```
1  // fig. 6.14: fig06_14.cpp
2  // Demonstrating a default constructor for class time.
3  #include <iostream>
4
5  using std::cout;
6  using std::endl;
7
8  // include definition of class time from time2.h
9  #include "time2.h"
10
11 int main()
12 {
13    time t1;                // all arguments defaulted
14    time t2( 2 );           // minute and second defaulted
15    time t3( 21, 34 );      // second defaulted
16    time t4( 12, 25, 42 );  // all values specified
17    time t5( 27, 74, 99 );  // all bad values specified
18
19    cout << "constructed with:\n\n"
20         << "all default arguments:\n  ";
21    t1.printUniversal();    // 00:00:00
22    cout << "\n  ";
23    t1.printStandard();     // 12:00:00 AM
24
25    cout << "\n\nhour specified; default minute and second:\n  ";
26    t2.printUniversal();    // 02:00:00
27    cout << "\n  ";
28    t2.printStandard();     // 2:00:00 AM
29
30    cout << "\n\nhour and minute specified; default second:\n  ";
31    t3.printUniversal();    // 21:34:00
32    cout << "\n  ";
33    t3.printStandard();     // 9:34:00 PM
34
35    cout << "\n\nhour, minute, and second specified:\n  ";
36    t4.printUniversal();    // 12:25:42
37    cout << "\n  ";
38    t4.printStandard();     // 12:25:42 PM
39
40    cout << "\n\nall invalid values specified:\n  ";
41    t5.printUniversal();    // 00:00:00
42    cout << "\n  ";
43    t5.printStandard();     // 12:00:00 AM
44    cout << endl;
45
46    return 0;
47
48 } // end main
```

Initialize **Time** objects using default arguments.

Initialize **Time** object with invalid values; validity checking will set values to 0.

t5 constructed with invalid arguments; values set to 0.

fig06_14.cpp (1 of 2)

fig06_14.cpp (2 of 2)


```
constructed with:

all default arguments:
  00:00:00
  12:00:00 AM

hour specified; default minute and second:
  02:00:00
  2:00:00 AM

hour and minute specified; default second:
  21:34:00
  9:34:00 PM

hour, minute, and second specified:
  12:25:42
  12:25:42 PM

all invalid values specified:
  00:00:00
  12:00:00 AM
```


fig06_14.cpp output (1 of 1)

6.14 Using *Set* and *Get* Functions

- Set functions
  - Perform validity checks before modifying `private` data
  - Notify if invalid values
  - Indicate with return values
- Get functions
  - "Query" functions
  - Control format of data returned


```
1  // fig. 6.15: time1.h
2  // Declaration of class time.
3  // Member functions defined in time1.cpp
4
5  // prevent multiple inclusions of header file
6  #ifndef TIME1_H
7  #define TIME1_H
8
9  class time {
10
11 public:
12    time( int = 0, int = 0, int = 0 );  // default constructor
13
14    // set functions
15    void setTime( int, int, int );  // set hour, minute, second
16    void setHour( int );    // set hour
17    void setMinute( int );  // set minute
18    void setSecond( int );  // set second
19
20    // get functions
21    int getHour();          // return hour
22    int getMinute();        // return minute
23    int getSecond();        // return second
24
25    void printUniversal(); // output universal-time format
26    void printStandard();  // output standard-time format
27
28 private:
29    int hour;              // 0 - 23 (24-hour clock format)
30    int minute;            // 0 - 59
31    int second;            // 0 - 59
32
33 }; // end class time
34
35 #endif
```

Set functions.

Get functions.

time1.h (1 of 2)

time1.h (2 of 2)

```
// fig. 6.11: time1.cpp
// member-function definitions for time class.
#include <iostream>

using std::cout;

#include <iomanip>

using std::setfill;
using std::setw;

// include definition of class time from time1.h
#include "time1.h"

// constructor function to initialize private data;
// calls member function setTime to set variables;
// default values are 0 (see class definition)
Time::Time( int hr, int min, int sec )
{
   setTime( hr, min, sec );

} // end time constructor
```

Outline

time1.cpp (1 of 4)


```
// set hour, minute and second values
void Time::setTime( int h, int m, int s )
{
   setHour( h );
   setMinute( m );
   setSecond( s );

} // end function setTime

// set hour value
void Time::setHour( int h )
{
   hour = ( h >= 0 && h < 24 ) ? h : 0;

} // end function setHour

// set minute value
void Time::setMinute( int m )
{
   minute = ( m >= 0 && m < 60 ) ? m : 0;

} // end function setMinute
```

Call set functions to perform validity checking.

Set functions perform validity checks before modifying data.

Outline

time1.cpp (2 of 4)


```
// set second value
void Time::setSecond( int s )
{
   second = ( s >= 0 && s < 60 ) ? s : 0;

} // end function setSecond

// return hour value
int Time::getHour()
{
   return hour;

} // end function getHour

// return minute value
int Time::getMinute()
{
   return minute;

} // end function getMinute
```

Get functions allow client to read data.

Outline

time1.cpp (3 of 4)


```
// return second value
int Time::getSecond()
{
   return second;

} // end function getSecond

// print time in universal format
void Time::printUniversal()
{
   cout << setfill( '0' ) << setw( 2 ) << hour << ":"
        << setw( 2 ) << minute << ":"
        << setw( 2 ) << second;

} // end function printUniversal

// print time in standard format
void Time::printStandard()
{
   cout << ( ( hour == 0 || hour == 12 ) ? 12 : hour % 12 )
        << ":" << setfill( '0' ) << setw( 2 ) << minute
        << ":" << setw( 2 ) << second
        << ( hour < 12 ? " AM" : " PM" );

} // end function printStandard
```

Get function allows client to read data.

Outline

time1.cpp (4 of 4)


```
// fig. 6.12: fig06_12.cpp
// demonstrating the time class set and get functions
#include <iostream>

using std::cout;
using std::endl;

// include definition of class time from time1.h
#include "time1.h"

void incrementMinutes( Time &, const int );  // prototype

int main()
{
   Time t;                  // create Time object

   // set time using individual set functions
   t.setHour( 17 );     // set hour to valid value
   t.setMinute( 34 );   // set minute to valid value
   t.setSecond( 25 );   // set second to valid value
```

Invoke set functions to set valid values.

Outline

fig06_12.cpp (1 of 2)


```
   // use get functions to obtain hour, minute and second
   cout << "Result of setting all valid values:\n"
        << "  Hour: " << t.getHour()
        << "  Minute: " << t.getMinute()
        << "  Second: " << t.getSecond();

   // set time using individual set functions
   t.setHour( 234 );    // invalid hour set to 0
   t.setMinute( 43 );   // set minute to valid value
   t.setSecond( 6573 ); // invalid second set to 0

   // display hour, minute and second after setting
   // invalid hour and second values
   cout << "\n\nResult of attempting to set invalid hour and"
        << " second:\n  Hour: " << t.getHour()
        << "  Minute: " << t.getMinute()
        << "  Second: " << t.getSecond() << "\n\n";

   t.setTime( 11, 58, 0 );    // set time
   incrementMinutes( t, 3 );  // increment t's minute by 3

   return 0;

} // end main
```

Attempt to set invalid values using set functions.

Invalid values result in setting data members to 0.

Modify data members using function **setTime**.