

```
1 //
2 // Shaun Chemplavil U08713628
3 // shaun.chemplavil@gmail.com
4 // C/C++ Programming III : Intermediate Programming with Objects
5 // 151116 Raymond L.Mitchell III
6 // Date.h
7 // Win10
8 // Visual C++ 19.0
9 //
10 // File contains the Date class definition
11 //
12
13 #ifndef SHAUNCHEMPLAVIL_DATE_H
14 #define SHAUNCHEMPLAVIL_DATE_H
15
16 #include <iostream>
17 #include <ctime>
18 using std::cerr;
19
20 namespace ShaunChemplavil
21 {
22     class Date
23     {
24     public:
25
26         inline Date(int month, int day, int year);
27         inline Date();
28         inline int getMonth() const;
29         inline int getDay() const;
30         inline int getYear() const;
31         // Print Date in "standard" US calendar format
32         void display() const;
33
34     private:
35         int month, day, year;
36     };
37 }
38
39 // Constructor
40 ShaunChemplavil::Date::Date(int month, int day, int year)
41 {
42     // Default the max number of days equal to 31 (odd months)
43     int FebMonth = 2, maxDays = 31;
44
45     // We need to check if our input was valid
46     // Check if day is valid
47     if ((this->month != FebMonth) && (bool)(this->month % 2))
48         // number of days in an "even" month
49         maxDays = 30;
50     else // Month is February, we are ignoring leap years
51     {
52         maxDays = 28;
```

```
53     }
54
55     // Check if month, then day, then year is valid, if all valid set vars
56     if ((month < 1) || (month > 12))
57         cerr << "\nERROR: INVALID MONTH VALUE!\n";
58     else if ((day < 1) || (day > maxDays))
59         cerr << "\nERROR: INVALID DAY VALUE!\n";
60     else if (year < 1)
61         cerr << "\nERROR: INVALID YEAR VALUE!\n";
62     // Valid Date!
63     else
64     {
65         this->year = year;
66         this->day = day;
67         this->month = month;
68     }
69
70 }
71
72 // Default constructor using current date
73 ShaunChemplavil::Date::Date()
74 {
75     // Starting year of tm struct
76     int tmYearStart = 1900;
77     // tm Struct indexes month via 'zero indexing'
78     int tmMonthStart = 1;
79     time_t currTime = time(0);
80     // Converting from time_t to tm structure
81     tm *today = localtime(&currTime);
82
83     this->year = today->tm_year + tmYearStart;
84     this->month = today->tm_mon + tmMonthStart;
85     this->day = today->tm_mday;
86
87 }
88
89 int ShaunChemplavil::Date::getMonth() const
90 {
91     return this->month;
92 }
93
94
95 int ShaunChemplavil::Date::getDay() const
96 {
97     return this->day;
98 }
99
100 int ShaunChemplavil::Date::getYear() const
101 {
102     return this->year;
103 }
104
```

105

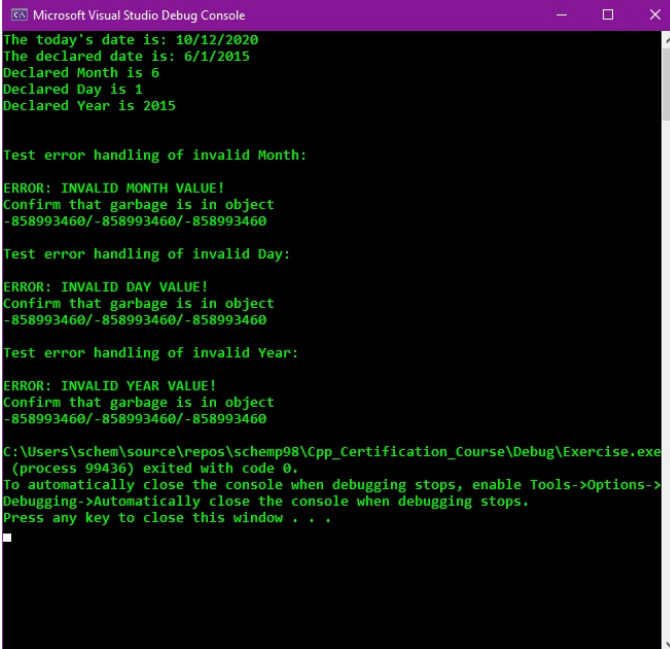
106 #endif

107

```
1 //
2 // Shaun Chemplavil U08713628
3 // shaun.chemplavil@gmail.com
4 // C/C++ Programming III : Intermediate Programming with Objects
5 // 151116 Raymond L.Mitchell III
6 // Date.cpp
7 // Win10
8 // Visual C++ 19.0
9 //
10 // File contains the display member function for the Date class
11 //
12
13 #include <iostream>
14 #include "Date.h"
15 using std::cout;
16
17 // Print Date in "standard" US calendar format
18 void ShaunChemplavil::Date::display() const
19 {
20     cout << getMonth() << "/" << getDay() << "/" << getYear() << "\n";
21 }
22
```

```
1  //
2  // Shaun Chemplavil U08713628
3  // shaun.chemplavil@gmail.com
4  // C/C++ Programming III : Intermediate Programming with Objects
5  // 151116 Raymond L.Mitchell III
6  // hw2.cpp
7  // Win10
8  // Visual C++ 19.0
9  //
10 // Test Program for the Date class
11 //
12
13 #include <iostream>
14 #include "Date.h"
15 using std::cout;
16 using std::cerr;
17
18 int main()
19 {
20     // declare original date variables (arbitrary values)
21     int month = 6, day = 1, year = 2015;
22
23     // exercise all of Date's public functions
24     ShaunChemplavil::Date validDate(month, day, year), todaysDate;
25
26     //exercising the display function
27     cout << "The today's date is: ";
28     todaysDate.display();
29
30     //exercising the display function
31     cout << "The declared date is: ";
32     validDate.display();
33
34     // exercising all of the get functions
35     cout << "Declared Month is " << validDate.getMonth() << "\n"
36         << "Declared Day is " << validDate.getDay() << "\n"
37         << "Declared Year is " << validDate.getYear() << "\n";
38
39     // testing setMonth error handling;
40     cout << "\n\nTest error handling of invalid Month:\n";
41     ShaunChemplavil::Date badMonth(year, day, year);
42     cout << "Confirm that garbage is in object\n";
43     badMonth.display();
44
45     cout << "\nTest error handling of invalid Day:\n";
46     ShaunChemplavil::Date badDay(month, year, year);
47     cout << "Confirm that garbage is in object\n";
48     badDay.display();
49
50     cout << "\nTest error handling of invalid Year:\n";
51     ShaunChemplavil::Date badYear(month, day, -year);
52     cout << "Confirm that garbage is in object\n";
```

```
53     badYear.display();  
54 }  
55
```



```
Microsoft Visual Studio Debug Console  
The today's date is: 10/12/2020  
The declared date is: 6/1/2015  
Declared Month is 6  
Declared Day is 1  
Declared Year is 2015  
  
Test error handling of invalid Month:  
  
ERROR: INVALID MONTH VALUE!  
Confirm that garbage is in object  
-858993460/-858993460/-858993460  
  
Test error handling of invalid Day:  
  
ERROR: INVALID DAY VALUE!  
Confirm that garbage is in object  
-858993460/-858993460/-858993460  
  
Test error handling of invalid Year:  
  
ERROR: INVALID YEAR VALUE!  
Confirm that garbage is in object  
-858993460/-858993460/-858993460  
  
C:\Users\schem\source\repos\schemp98\Cpp_Certification_Course\Debug\Exercise.exe  
(process 99436) exited with code 0.  
To automatically close the console when debugging stops, enable Tools->Options->  
Debugging->Automatically close the console when debugging stops.  
Press any key to close this window . . .  
^
```