

```
1 //
2 // Class source file for A4
3 //
4 // Hal Bettle
5 //
6 // 5 September 2008
7 //
8
9 #include "CPTN230A4class_bettle.h"
10
11 Date::Date()
12 {
13     temp_time = time(NULL);
14     time_struct = localtime(&temp_time);
15     year = (time_struct->tm_year + 1900);
16     month = (time_struct->tm_mon + 1);
17     day = (time_struct->tm_mday);
18     validate_date(day, month, year);
19 }
20
21 Date::Date(int d)
22 {
23     temp_time = time(NULL);
24     time_struct = localtime(&temp_time);
25     year = (time_struct->tm_year + 1900);
26     month = (time_struct->tm_mon + 1);
27     set_day(d);
28     validate_date(day, month, year);
29 }
30
31 Date::Date(int d, int m)
32 {
33     temp_time = time(NULL);
34     time_struct = localtime(&temp_time);
35     year = (time_struct->tm_year + 1900);
36     set_month(m);
37     set_day(d);
38     validate_date(day, month, year);
39 }
40
41 Date::Date(int d, int m, int y)
42 {
43     set_year(y);
44     set_month(m);
45     set_day(d);
46     validate_date(day, month, year);
47 }
48
49 int Date::get_day(void)
50 {
51     return day;
52 }
53
54 int Date::get_month(void)
55 {
56     return month;
57 }
58
59 int Date::get_year(void)
60 {
61     return year;
62 }
63
64 bool Date::operator==(const Date &source)
65 {
66     if ( (year == source.year) &&
```

```
67         (month == source.month) &&
68         (day == source.day)
69     )
70     {
71         compare_status = true;
72     }
73     else
74     {
75         compare_status = false;
76     }
77     return compare_status;
78 }
79
80 bool Date::operator!=(const Date &source)
81 {
82     if ( (year != source.year) ||
83         (month != source.month) ||
84         (day != source.day)
85     )
86     {
87         compare_status = true;
88     }
89     else
90     {
91         compare_status = false;
92     }
93     return compare_status;
94 }
95
96 bool Date::operator<(const Date &source)
97 {
98     if (year < source.year)
99     {
100         compare_status = 1;
101     }
102     else if ( (year == source.year) && (month < source.month) )
103     {
104         compare_status = 1;
105     }
106     else if ( (year == source.year) && (month == source.month) && (day < source.day) )
107     {
108         compare_status = 1;
109     }
110     else
111     {
112         compare_status = 0;
113     }
114     return compare_status;
115 }
116
117 bool Date::operator<=(const Date &source)
118 {
119     if ( (*this < source) || (*this == source) )
120     {
121         compare_status = 1;
122     }
123     else
124     {
125         compare_status = 0;
126     }
127     return compare_status;
128 }
129
130 bool Date::operator>(const Date &source)
131 {
132     if (year > source.year)
```

```
133     {
134         compare_status = 1;
135     }
136     else if ( (year == source.year) && (month > source.month) )
137     {
138         compare_status = 1;
139     }
140     else if ( (year == source.year) && (month == source.month) && (day > source.day) )
141     {
142         compare_status = 1;
143     }
144     else
145     {
146         compare_status = 0;
147     }
148     return compare_status;
149 }
150
151 bool Date::operator>=(const Date &source)
152 {
153     if ( (*this > source) || (*this == source) )
154     {
155         compare_status = 1;
156     }
157     else
158     {
159         compare_status = 0;
160     }
161     return compare_status;
162 }
163
164 Date &Date::operator++()
165 {
166     if ( (month == 12) && (day == 31) )
167     {
168         month = 1;
169         day = 1;
170         year++;
171     }
172     else if ( ( (num_days == 31) && (day == 31) ) ||
173              ( (num_days == 30) && (day == 30) ) )
174     {
175     }
176     {
177         day = 1;
178         month++;
179     }
180     else if ( ( leap_year && (day == 29) ) ||
181              ( !leap_year && (day == 28) ) )
182     {
183         day = 1;
184         month = 3;
185     }
186     else
187     {
188         day++;
189     }
190     return *this;
191 }
192
193 // Private support member functions
194
195 void Date::set_day(int d)
196 {
197     day = d;
198     return;
```

```

199 }
200
201 void Date::set_month(int m)
202 {
203     month = m;
204     return;
205 }
206
207 void Date::set_year(int y)
208 {
209     year = y;
210     return;
211 }
212
213 void Date::validate_date(int da, int mon, int ye)
214 {
215     int error = 0;
216
217     if ( ( ( (ye % 4) == 0) && ( !(ye % 100) == 0) ) ||
218         ( (ye % 400) == 0)
219         )
220     {
221         leap_year = true;
222     }
223     else
224     {
225         leap_year = false;
226     }
227
228     switch (mon)
229     {
230         case 1:
231         case 3:
232         case 5:
233         case 7:
234         case 8:
235         case 10:
236         case 12: if ((da < 1) || (da > 31))
237                 {
238                     error = 1;
239                 }
240                 else
241                 {
242                     num_days = 31;
243                 }
244                 break;
245         case 4:
246         case 6:
247         case 9:
248         case 11: if ((da < 1) || (da > 30))
249                 {
250                     error = 1;
251                 }
252                 else
253                 {
254                     num_days = 30;
255                 }
256                 break;
257         case 2: if ( ( !leap_year && ( (da < 1) || (da > 28) ) ) ||
258                 ( leap_year && ( (da < 1) || (da > 29) ) )
259                 )
260                 {
261                     error = 1;
262                 }
263                 break;
264         default: error = 1;

```

```
265             break;
266         }
267
268     if (error)
269     {
270         cout << "Month or day invalid mon = " << mon << ", day = " << da << endl;
271         num_days = 31;
272         leap_year = true;
273         set_year(2000);
274         set_month(1);
275         set_day(1);
276     }
277
278     return;
279 }
```