

```
1  /*****
2  * AUTHOR          : Nick Reardon
3  * Assignment #6   : Priority Queues
4  * CLASS           : CS1D
5  * SECTION        : MW - 2:30p
6  * DUE DATE       : 02 / 24 / 20
7  *****/
8  #include "main.h"
9  #include <queue>
10 using std::cout; using std::endl;
11
12
13 int main()
14 {
15     /*
16     * HEADER OUTPUT
17     */
18     PrintHeader(cout, "Prompt.txt");
19
20     /*****/
21
22     cout << endl << " Now doing written Priority Queue based on a heap" << endl
23         << endl;
24
25     const int MAX_CARE_TIME = 25;
26     const int MAX_TOTAL_MINUTES = 300;
27
28     ArrayMaxHeap<std::string, int> heap;
29
30     cout << endl << " -- 12::00 -- \n Creating priority queue of waiting
31         patients..." << endl << endl;
32
33     heap.insert("Bob Bleeding", 5);
34     heap.insert("Frank Feelingbad", 3);
35     heap.insert("Cathy Coughing", 2);
36     heap.insert("Paula Pain", 4);
37     heap.insert("Alice Ailment", 7);
38     heap.insert("Irene Ill", 1);
39     heap.insert("Tom Temperature", 6);
40
41     int timer = 0;
42     int emergencyTimer = 0;
43     bool emergency = false;
44
45     for (int time = 0; time <= MAX_TOTAL_MINUTES; )
46     {
47         if (!heap.empty())
48         {
49             switch (time)
50             {
```

```
51         case 74:
52             heap.insert("Sam Sneezing", 1100);
53             emergency = true;
54             break;
55
56         case 181:
57
58             heap.insert("Sid Sickly", 100);
59             emergency = true;
60             break;
61     }
62
63     if (emergency)
64     {
65         if (emergencyTimer == 0)
66         {
67             if (!heap.empty())
68             {
69                 InterruptPatient(heap, time, timer);
70
71                 PriorityPatient(heap, time);
72             }
73         }
74
75         emergencyTimer++;
76         time++;
77
78         if (emergencyTimer == 25)
79         {
80             DischargePatient(heap, time);
81
82             heap.remove();
83
84             if (!heap.empty())
85             {
86                 ResumePatient(heap, time, timer);
87             }
88             emergency = false;
89
90             emergencyTimer = 0;
91         }
92     }
93
94     else if (!emergency)
95     {
96         if (timer == 0)
97         {
98             if (!heap.empty())
99             {
100                 AdmitPatient(heap, time);
101             }
102         }
103     }
```

```

103
104         timer++;
105         time++;
106
107         if (timer == MAX_CARE_TIME)
108         {
109             timer = 0;
110
111             DischargePatient(heap, time);
112
113             heap.remove();
114
115         }
116     }
117
118 }
119 else
120 {
121     time++;
122 }
123 }
124
125 cout << endl << "   --- END OF DAY   " << ConvertTime(MAX_TOTAL_MINUTES, 12) << "\n"
126     << endl << endl;
127
128
129 //
130     *****
131 //
132     *****
133
134 cout << std::string(60, '_') << endl;
135 cout << endl << " Now doing STL Priority Queue " << endl << endl;
136
137 std::priority_queue< std::pair< int, std::string>> STL_PrioQ;
138
139 cout << endl << "  -- 12:00 -- \n Creating priority queue of waiting
140     patients... " << endl << endl;
141
142 STL_PrioQ.push(std::make_pair(5, "Bob Bleeding"));
143 STL_PrioQ.push(std::make_pair(3, "Frank Feelingbad"));
144 STL_PrioQ.push(std::make_pair(2, "Cathy Coughing"));
145 STL_PrioQ.push(std::make_pair(4, "Paula Pain"));
146 STL_PrioQ.push(std::make_pair(7, "Alice Ailment"));
147 STL_PrioQ.push(std::make_pair(1, "Irene Ill"));
148 STL_PrioQ.push(std::make_pair(6, "Tom Temperature"));
149
150 timer = 0;
151 emergencyTimer = 0;

```

```
149     emergency = false;
150
151
152     for (int time = 0; time <= MAX_TOTAL_MINUTES; )
153     {
154         if (!STL_PrioQ.empty())
155         {
156             switch (time)
157             {
158                 case 74:
159                     STL_PrioQ.push(std::make_pair(999, "Sam Sneezing"));
160                     emergency = true;
161                     break;
162
163                 case 181:
164
165                     STL_PrioQ.push(std::make_pair(999, "Sid Sickly"));
166                     emergency = true;
167                     break;
168             }
169
170             if (emergency)
171             {
172                 if (emergencyTimer == 0)
173                 {
174                     if (!STL_PrioQ.empty())
175                     {
176                         cout << "Patient Care Interrupted:" << endl
177                             << "Name: " << STL_PrioQ.top().second << endl
178                             << "Care interrupted at " << ConvertTime(time, 12, 7
179                             false) << endl
180                             << "Minutes in visit remaining: " << 25 - timer
181                             << endl << endl;
182
183                         cout << "High Priority Patient Recieved" << endl
184                             << "Immediate attention administered:" << endl
185                             << "Name: " << STL_PrioQ.top().second << endl
186                             << "Care began at " << ConvertTime(time, 12, false)
187                             << endl << endl;
188                     }
189                 }
190
191                 emergencyTimer++;
192                 time++;
193
194                 if (emergencyTimer == 25)
195                 {
196                     cout << "Patient Discharge:" << endl
197                         << "Name: " << STL_PrioQ.top().second << endl
198                         << "Care ended at " << ConvertTime(time, 12, false)
199                         << endl << endl;
```

```
200         STL_PrioQ.pop();
201
202         if (!STL_PrioQ.empty())
203         {
204             cout << "Patient Care Resumed:" << endl
205                  << "Name: " << STL_PrioQ.top().second << endl
206                  << "Care resumed at " << ConvertTime(time, 12, false) << endl
207                  << "Minutes in visit remaining: " << 25 - timer
208                  << endl << endl;
209         }
210         emergency = false;
211
212         emergencyTimer = 0;
213
214     }
215 }
216 else if (!emergency)
217 {
218     if (timer == 0)
219     {
220         if (!STL_PrioQ.empty())
221         {
222             cout << "Patient Admitted:" << endl
223                  << "Name: " << STL_PrioQ.top().second << endl
224                  << "Care began at " << ConvertTime(time, 12, false)
225                  << endl << endl;
226         }
227     }
228
229     timer++;
230     time++;
231
232     if (timer == MAX_CARE_TIME)
233     {
234         timer = 0;
235
236         cout << "Patient Discharge:" << endl
237              << "Name: " << STL_PrioQ.top().second << endl
238              << "Care ended at " << ConvertTime(time, 12, false)
239              << endl << endl;
240
241         STL_PrioQ.pop();
242
243     }
244 }
245
246 }
247 else
248 {
249     time++;
250 }
```

```
251     }
252
253     cout << endl << "   --- END OF DAY   " << ConvertTime(MAX_TOTAL_MINUTES, 12) << "
254         "   ---"
255         << endl << endl;
256     system("pause");
257     return 0;
258 }
259
260
261
262 std::string ConvertTime(int totalMinutes, int startHour, bool hours24Style)
263 {
264     int minutes;
265     int hours;
266     std::string output = "";
267
268     minutes = totalMinutes % 60;
269
270     if (!hours24Style)
271     {
272         if (startHour == 12)
273         {
274             hours = totalMinutes / 60;
275             if (hours == 0)
276             {
277                 hours = 12;
278             }
279         }
280         else
281         {
282             hours = startHour + (totalMinutes / 60);
283         }
284     }
285
286
287
288     if ((hours < 10))
289     {
290         output += '0' + std::to_string(hours);
291     }
292     else
293     {
294         output += std::to_string(hours);
295     }
296
297     output += ":";
298
299     if ((minutes < 10))
300     {
301         output += '0' + std::to_string(minutes);
```

```
302     }
303     else
304     {
305         output += std::to_string(minutes);
306     }
307
308
309
310     return output;
311 }
312 }
313
314 void DischargePatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes)
315 {
316
317     cout << "Patient Discharge:" << endl
318         << "Name: " << heap.max() << endl
319         << "Care ended at " << ConvertTime(totalMinutes, 12, false)
320         << endl << endl;
321
322
323 }
324
325 void AdmitPatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes)
326 {
327
328     cout << "Patient Admitted:" << endl
329         << "Name: " << heap.max() << endl
330         << "Care began at " << ConvertTime(totalMinutes, 12, false)
331         << endl << endl;
332 }
333
334 void InterruptPatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes,  ➤
    int currentTimer)
335 {
336
337     cout << "Patient Care Interrupted:" << endl
338         << "Name: " << heap.max() << endl
339         << "Care interrupted at " << ConvertTime(totalMinutes, 12, false) << endl
340         << "Minutes in visit remaining: " << 25 - currentTimer
341         << endl << endl;
342 }
343
344 void ResumePatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes, int  ➤
    currentTimer)
345 {
346
347     cout << "Patient Care Resumed:" << endl
348         << "Name: " << heap.max() << endl
349         << "Care resumed at " << ConvertTime(totalMinutes, 12, false) << endl
350         << "Minutes in visit remaining: " << 25 - currentTimer
351         << endl << endl;
```

```
352 }
353
354 void PriorityPatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes)
355 {
356
357     cout << "High Priority Patient Recieved" << endl
358         << "Immediate attention administered:" << endl
359         << "Name: " << heap.max() << endl
360         << "Care began at " << ConvertTime(totalMinutes, 12, false)
361         << endl << endl;
362 }
```