```cpp
/*****************************************************************************
 * AUTHOR           : Nick Reardon
 * Assignment #6    : Priority Queues
 * CLASS            : CS1D
 * SECTION          : MW - 2:30p
 * DUE DATE         : 02 / 24 / 20
 *****************************************************************************/
#include "main.h"
#include <queue>
using std::cout; using std::endl;


int main()
{

    /*
     * HEADER OUTPUT
     */
    PrintHeader(cout, "Prompt.txt");

    /*******************************************************************/

    cout << endl << " Now doing written Priority Queue based on a heap" << endl
      << endl;

    const int MAX_CARE_TIME = 25;
    const int MAX_TOTAL_MINUTES = 300;

    ArrayMaxHeap<std::string, int> heap;

    cout << endl << "  -- 12::00 -- \n Creating priority queue of waiting
      patients..." << endl << endl;

    heap.insert("Bob Bleeding", 5);
    heap.insert("Frank Feelingbad", 3);
    heap.insert("Cathy Coughing", 2);
    heap.insert("Paula Pain", 4);
    heap.insert("Alice Ailment", 7);
    heap.insert("Irene Ill", 1);
    heap.insert("Tom Temperature", 6);

    int timer = 0;
    int emergencyTimer = 0;
    bool emergency = false;


    for (int time = 0; time <= MAX_TOTAL_MINUTES; )
    {
        if (!heap.empty())
        {
            switch (time)
            {
```

```
            case 74:
                heap.insert("Sam Sneezing", 1100);
                emergency = true;
                break;

            case 181:

                heap.insert("Sid Sickly", 100);
                emergency = true;
                break;
            }

            if (emergency)
            {
                if (emergencyTimer == 0)
                {
                    if (!heap.empty())
                    {
                        InterruptPatient(heap, time, timer);

                        PriorityPatient(heap, time);
                    }
                }

                emergencyTimer++;
                time++;

                if (emergencyTimer == 25)
                {
                    DischargePatient(heap, time);

                    heap.remove();

                    if (!heap.empty())
                    {
                        ResumePatient(heap, time, timer);
                    }
                    emergency = false;

                    emergencyTimer = 0;

                }
            }
            else if (!emergency)
            {
                if (timer == 0)
                {
                    if (!heap.empty())
                    {
                        AdmitPatient(heap, time);
                    }
                }
```

```cpp

                timer++;
                time++;

                if (timer == MAX_CARE_TIME)
                {
                    timer = 0;

                    DischargePatient(heap, time);

                    heap.remove();

                }
            }

        }
        else
        {
            time++;
        }
    }

    cout << endl << "  --- END OF DAY  " << ConvertTime(MAX_TOTAL_MINUTES, 12) << " ----"
        << endl << endl;


    // *************************************************************************************

    // *************************************************************************************
    cout << std::string(60, '_') << endl;
    cout << endl << " Now doing STL Priority Queue" << endl << endl;

    std::priority_queue< std::pair< int, std::string>> STL_PrioQ;

    cout << endl << "  -- 12::00 -- \n Creating priority queue of waiting patients... " << endl << endl;

    STL_PrioQ.push(std::make_pair(5, "Bob Bleeding"));
    STL_PrioQ.push(std::make_pair(3, "Frank Feelingbad"));
    STL_PrioQ.push(std::make_pair(2, "Cathy Coughing"));
    STL_PrioQ.push(std::make_pair(4, "Paula Pain"));
    STL_PrioQ.push(std::make_pair(7, "Alice Ailment"));
    STL_PrioQ.push(std::make_pair(1, "Irene Ill"));
    STL_PrioQ.push(std::make_pair(6, "Tom Temperature"));

    timer = 0;
    emergencyTimer = 0;
    emergency = false;


    for (int time = 0; time <= MAX_TOTAL_MINUTES; )
    {
        if (!STL_PrioQ.empty())
        {
            switch (time)
            {
            case 74:
                STL_PrioQ.push(std::make_pair(999, "Sam Sneezing"));
                emergency = true;
                break;

            case 181:

                STL_PrioQ.push(std::make_pair(999, "Sid Sickly"));
                emergency = true;
                break;
            }

            if (emergency)
            {
                if (emergencyTimer == 0)
                {
                    if (!STL_PrioQ.empty())
                    {
                        cout << "Patient Care Interrupted:" << endl
                            << "Name:  " << STL_PrioQ.top().second << endl
                            << "Care interrupted at " << ConvertTime(time, 12, 
                     false) << endl
                            << "Minutes in visit remaining: " << 25 - timer
                            << endl << endl;

                        cout << "High Priority Patient Recieved" << endl
                            << "Immediate attention administered:" << endl
                            << "Name:  " << STL_PrioQ.top().second << endl
                            << "Care began at " << ConvertTime(time, 12, false)
                            << endl << endl;
                    }
                }

                emergencyTimer++;
                time++;

                if (emergencyTimer == 25)
                {
                    cout << "Patient Discharge:" << endl
                        << "Name:  " << STL_PrioQ.top().second << endl
                        << "Care ended at " << ConvertTime(time, 12, false)
                        << endl << endl;

                    STL_PrioQ.pop();

                    if (!STL_PrioQ.empty())
                    {
                        cout << "Patient Care Resumed:" << endl
                            << "Name: " << STL_PrioQ.top().second << endl
                            << "Care resumed at " << ConvertTime(time, 12, false) ⤶
                      << endl
                            << "Minutes in visit remaining: " << 25 - timer
                            << endl << endl;
                    }
                    emergency = false;

                    emergencyTimer = 0;

                }
            }
            else if (!emergency)
            {
                if (timer == 0)
                {
                    if (!STL_PrioQ.empty())
                    {
                        cout << "Patient Admitted:" << endl
                            << "Name: " << STL_PrioQ.top().second << endl
                            << "Care began at " << ConvertTime(time, 12, false)
                            << endl << endl;
                    }
                }

                timer++;
                time++;

                if (timer == MAX_CARE_TIME)
                {
                    timer = 0;

                    cout << "Patient Discharge:" << endl
                        << "Name: " << STL_PrioQ.top().second << endl
                        << "Care ended at " << ConvertTime(time, 12, false)
                        << endl << endl;

                    STL_PrioQ.pop();

                }
            }

        }
        else
        {
            time++;
        }
```

```
    }

    cout << endl << "  --- END OF DAY  " << ConvertTime(MAX_TOTAL_MINUTES, 12) << 
      " ----"
        << endl << endl;

    system("pause");
    return 0;
}



std::string ConvertTime(int totalMinutes, int startHour, bool hours24Style)
{
    int minutes;
    int hours;
    std::string output = "";

    minutes = totalMinutes % 60;

    if (!hours24Style)
    {
        if (startHour == 12)
        {
            hours = totalMinutes / 60;
            if (hours == 0)
            {
                hours = 12;
            }
        }
        else
        {
            hours = startHour + (totalMinutes / 60);
        }
    }



    if ((hours < 10))
    {
        output += '0' + std::to_string(hours);
    }
    else
    {
        output += std::to_string(hours);
    }

    output += ":";

    if ((minutes < 10))
    {
        output += '0' + std::to_string(minutes);
    }
    else
    {
        output += std::to_string(minutes);
    }



    return output;

}

void DischargePatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes)
{

    cout << "Patient Discharge:" << endl
        << "Name: " << heap.max() << endl
        << "Care ended at " << ConvertTime(totalMinutes, 12, false)
        << endl << endl;


}

void AdmitPatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes)
{

    cout << "Patient Admitted:" << endl
        << "Name: " << heap.max() << endl
        << "Care began at " << ConvertTime(totalMinutes, 12, false)
        << endl << endl;
}

void InterruptPatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes, 
  int currentTimer)
{

    cout << "Patient Care Interrupted:" << endl
        << "Name: " << heap.max() << endl
        << "Care interrupted at " << ConvertTime(totalMinutes, 12, false) << endl
        << "Minutes in visit remaining: " << 25 - currentTimer
        << endl << endl;
}

void ResumePatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes, int 
  currentTimer)
{

    cout << "Patient Care Resumed:" << endl
        << "Name: " << heap.max() << endl
        << "Care resumed at " << ConvertTime(totalMinutes, 12, false) << endl
        << "Minutes in visit remaining: " << 25 - currentTimer
        << endl << endl;
```

```
}

void PriorityPatient(ArrayMaxHeap <std::string, int>& heap, int totalMinutes)
{

    cout << "High Priority Patient Recieved" << endl
        << "Immediate attention administered:" << endl
        << "Name: " << heap.max() << endl
        << "Care began at " << ConvertTime(totalMinutes, 12, false)
        << endl << endl;
}
```