```
// Heartrate.cpp - HEART RATE MONITOR
//
// MAINTENANCE HISTORY
// DATE         PROGRAMMER AND DETAILS
// 06-10-15     JS      Original
//
//-----------------------------------------------------------------------------

#include <ctime>                // C-style system time functions
#include <cstring>              // C-style string manipulation functions
#include <cstdlib>              // C-style standard library
#include <iostream>             // C++ I/O stream declarations
#include <list>                 // C++ list declarations
using namespace std;    // Expand the standard namespace
#include "heartio.h"    // Heart rate infrastructure declarations

//-----------------------------------------------------------------------------

// DEFINITIONS

const long TICK_PERIOD = 10;
                                // Clock tick period in milliseconds
const long DEBOUNCE_TICKS = 5;
                                // Debounce clock ticks
const long TIME_MODULUS = 65536;
                                // Time modulus in clock ticks
const long INITIAL_REFRESH = 6000;
                                // Initial refresh period in clock ticks
const long REFRESH_PERIOD = 100;
                                // Refresh period in clock ticks
const long SAMPLE_PERIOD = 6000;
                                // Sample period in clock ticks

//-----------------------------------------------------------------------------

// MAIN LINE

int
main ()
{
        long            currentTime;    // Current time in clock ticks
        bool            beatActive;     // Heartbeat active flag
        long            debounceCnt;   // Debounce count
        list<long>      beatList;       // Beat time list
        long            nextRefresh;    // Next refresh time in clock ticks

        // Initialise the state variables

        currentTime = 0;
        beatActive = heart_beat();
        debounceCnt = 0;
        nextRefresh = INITIAL_REFRESH;

        // Clock tick loop

        for (;;) {
                heart_sleep (TICK_PERIOD);
                currentTime ++ ;
                if (currentTime > TIME_MODULUS) currentTime -= TIME_MODULUS;

                // Push new heartbeat times onto the heartbeat list

                if (heart_beat()) {
                        if ( ! beatActive) {
```

```
                        beatList.push_back (currentTime);
                        beatActive = 1;
                }
                debounceCnt = 0;
        } else if (beatActive) {
                debounceCnt ++ ;
                if (debounceCnt >= DEBOUNCE_TICKS)
                        beatActive = 0;
        }

        // If the refresh period has expired, display the
        // updated heart rate

        if (currentTime >= nextRefresh) {
                while (
                        beatList.begin() != beatList.end() &&
                        (currentTime + 2*TIME_MODULUS - beatList.front()
                                - SAMPLE_PERIOD) % TIME_MODULUS
                                        < TIME_MODULUS/2
                ) beatList.pop_front();
                cout << beatList.size() << '\n';
                nextRefresh += REFRESH_PERIOD;
        }
    }

    return 0;
}
```