```
#
# Makefile for threadex
# Author: Michael Kepple
# Date:   02 Apr 2013
#
all: threadex.c threadex.h
        gcc threadex.c threadex.h -lpthread -o threadex

clean:
        rm -rf *.o threadex
/*
 *
 * threadex.c
 *
 * Author: Michael Kepple
 * Date:   02 Apr 2013
 * Description: One thread produces one number in a sequence every three
 *   seconds while another checks the most recent number produced every second
 *   and prints out 'Odd!' and the number (in decimal and hex) if the produced
 *   number had an odd number of 'one bits'.
 *
*/
#include "threadex.h"
int producedNumber;

int main()
{
    pthread_t producer, consumer;
    pthread_create(&producer, NULL, (void*) produce, NULL);
    pthread_create(&consumer, NULL, (void*) consume, NULL);
    pthread_join(producer, NULL);
    pthread_join(consumer, NULL);
}

/*
 * Function: produce
 * Description: every three seconds, the thread function changed the global
 *   variable producedNumber to the next number in its iteration sequence, up
 *   to defined MAX_SEQUENCE.
 * Params: None.
 * Returns: Nothing.
 * Modifies: producedNumber
*/
void produce()
{
    int iter = 0;
    while (iter < MAX_SEQUENCE)
    {
        producedNumber = iter;
        sleep(PRODUCER_WAIT);
        iter++;
    }
}

/*
 * Function: consume
 * Description: every second, this thread function checks the balue of the global
 *   variable producedNumber. If the current value's binary representation
 *   contains an odd amount of one bits, number and its hex representation are
 *   printed to the screen.
 * Params: None.
 * Returns: Nothing.
 * Modifies: Nothing.
```

```
    */
    void consume()
    {
        do
        {
            sleep(CONSUMER_WAIT);
            if (oddOnes(producedNumber))
                printf("Odd! --- Decimal: %d  --- Hex: %04x \n", producedNumber,
                        producedNumber);
        } while (producedNumber < MAX_SEQUENCE);
    }

    /*
     * Function: oddOnes
     * Description: determines if the passed in integer's binary representation
     *   contain an odd amount of one bits.
     * Params: interger to be examined.
     * Return: TRUE if odd number of one bits, FALSE otherwise.
     * Modifies: Nothing.
    */
    boolean oddOnes(int number)
    {
        boolean isOdd = FALSE;
        while (number)
        {
            if (number & 1)
                isOdd = !isOdd;
            number = number >> 1;
        }
        return isOdd;
    }

    /*
     * File:   threadex.h
     * Author: Michael Kepple
     * Date:   02 Apr 2013
     *
    */
    #ifndef THREADEX_H
    #define THREADEX_H
    #include <stdlib.h>
    #include <stdio.h>
    #include <pthread.h>
    #include <time.h>
    typedef enum
    {
        FALSE,
        TRUE
    } boolean;
    void produce();
    void consume();
    boolean oddOnes(int number);
    #define MAX_SEQUENCE 1000
    #define PRODUCER_WAIT 3
    #define CONSUMER_WAIT 1
    #endif
```