```
/*
Adam Novoa
CS 372
2/25/2015
*/

#include<iostream>
#include<chrono>

using namespace std;

class Timer
{
public:
        Timer() : beg_(clock_::now()) {}
        void reset() { beg_ = clock_::now(); }
        double elapsed() const {
                return std::chrono::duration_cast<second_>
                        (clock_::now() - beg_).count();
        }
private:
        typedef std::chrono::high_resolution_clock clock_;
        typedef std::chrono::duration<double, std::ratio<1> > second_;
        std::chrono::time_point<clock_> beg_;
};

int iterative(int n);
int recursive(int n);
int val = 0;

int main()
{
        int n = 7;
        Timer timer1;
        Timer timer2;

        timer1.reset();
        cout << "The " << n << " Fibonacci numer is :" << recursive(n) << endl;
        cout << "Time taken to find :" << timer1.elapsed() << endl;
        timer2.reset();
        cout << "The " << n << " Fibonacci numer is :" << iterative(n) << endl;
        cout << "Time taken to find :" << timer2.elapsed() << endl;
        cout << val << endl;
        system("pause");
        return 0;
}

int iterative(int n)
{
        if (n == 0 || n == 1)
        {
                return n;
        }
        int f1 = 0;
        int f2 = 1;
        int answer = f1;

        for (int i = 2; i <= n; i++)
        {
                answer = f1 + f2;
```

```
                f1 = f2;
                f2 = answer;
        }
        return answer;
}

int recursive(int n)
{
        val++;
        if (n == 0 || n == 1)
        {
                return n;
        }
        return recursive(n - 1) + recursive(n - 2);
}
/*
Output
The 43 Fibonacci numer is :433494437
Time taken to find :54.8401
The 43 Fibonacci numer is :433494437
Time taken to find :0.0020001
Press any key to continue . . .

Output 2
The 44 Fibonacci numer is :701408733
Time taken to find :88.4541
The 44 Fibonacci numer is :701408733
Time taken to find :0.0060003
Press any key to continue . . .

So the largest number the recursive algorithm can return
within 60 secconds is 43. This is because it must first find the
previous two numbers before it can return the current number.
This means it takes (n-1)+(n-2) calls to find the nth number
where as it only takes n runs to find it iteratively.
*/
```