Iteration vs Recursion 2: Fibonacci – Documentation

Csc 600-01

Keawa Rozet

Code available at: [https://github.com/krozet/Iteration-vs-Recursion-2.git](https://github.com/krozet/Iteration-vs-Recursion-1.git)

The Problem

Write a recursive function Frec(n) that computes Fibonacci numbers. Then write an iterative version of Fibonacci number function Fit(n). Functions Frec(n) and Fit(n) return the same value but with different performance.

Write the main program that discovers the value N10 so that Frec(N10) runs on your machine close to 10 seconds. Then measure the run time of Fit(N10) and compute how many times is Fit(N10) faster than Frec(N10). Show what is N10 on your machine.

Implementation

I used two very standard methods for calculating the fibonacci sequence and felt no need to deviate from these methods. In my source code I call the method fibTimeMeasurement() to calculate how long either method takes to run the fibonacci(n) sequence.

At first I was using regular signed integers to hold the values and realized how futile it was after calling fibonacci(20). For the return values of the methods I switched to unsigned long long and that greatly increased the precision.

Looking at my execution, it is clear that I was not able to land directly on 10 seconds, but I feel that n=46 for my program will suffice for N10 for my machine. Fit() is about 3145078 times faster than Frec(), but this is hardly surprising when comparing a linear function, O(n), to an exponetial function, O(1.618^n).

Source Code

/\*

Keawa Rozet

CSc 600-01 - Programming Languages

Homework #2

Iteration vs Recursion 2

\*/

#include <iostream>

#include <stdio.h>

#include <stdlib.h>

#include <limits>

typedef std::numeric\_limits< double > dbl;

unsigned long long Frec(unsigned int n);

unsigned long long Fit(unsigned int n);

void fibTimeMeasurement(unsigned int n);

int main(int argc, char const\*\* argv) {

fibTimeMeasurement(46);

return 0;

}

unsigned long long Frec(unsigned int n) {

return (n < 2) ? n : Frec(n-1) + Frec(n-2);

}

unsigned long long Fit(unsigned int n) {

int i, zero = 0, first = 1, second = n;

for (i = 2; i <= n; i++) {

second = first + zero;

zero = first;

first = second;

}

return second;

}

void fibTimeMeasurement(unsigned int n) {

//first test binarySearchIteration

clock\_t begin = clock();

unsigned long long value = Fit(n);

clock\_t end = clock();

double fit\_elapsed\_secs = double(end - begin)/CLOCKS\_PER\_SEC;

std::cout.precision(dbl::max\_digits10);

std::cout << "Fib Iteration = " << value << " for n=" << n <<": " << std::fixed << fit\_elapsed\_secs << " seconds." << std::endl;

//second test binarySearchRecursion

clock\_t begin2 = clock();

unsigned long long value2 = Frec(n);

clock\_t end2 = clock();

double frec\_elapsed\_secs = double(end2 - begin2)/CLOCKS\_PER\_SEC;

std::cout.precision(dbl::max\_digits10);

std::cout << "Fib Recursion = " << value2 << " for n=" << n <<": " << std::fixed << frec\_elapsed\_secs << " seconds." << std::endl;

}

Example Build and Execution through Terminal

> g++ -o fb fib.cpp

> ./fb

Fib Iteration = 1134903170 for n=45: 0.00000200000000000 seconds.

Fib Recursion = 1134903170 for n=45: 7.77687699999999982 seconds.

> g++ -o fb fib.cpp

> ./fb

Fib Iteration = 1836311903 for n=46: 0.00000400000000000 seconds.

Fib Recursion = 1836311903 for n=46: 12.58031199999999927 seconds.