Amarilda Celhaka

Part C

*(C)*

*Compare running time of recursive and iterative functions for calculating Fibonacci Numbers*

**RECURSIVE FUNCTION SOURCE CODE**

#include "stdafx.h"

#include <iostream>

#include <chrono> //this header file used to measure time

using namespace std;

int calculateFibonaci(int);

int main()

{

int number;

cout << "Enter a number: ";

cin >> number;

cout << endl;

//starting to measure time by enabling steady\_clock

auto startTime = chrono::steady\_clock::now();

//validating user input, for number greater than zero only

while (number < 0)

{

cout << "ERROR! Your number must be greater than zero." << endl;

cout << "Enter the number again: ";

cin >> number;

}

//calling the function to calculate Fibunaci number

cout << "Fibonacci number for n = " << number <<

" is " << calculateFibonaci(number) << endl << endl;

auto stopTime = chrono::steady\_clock::now();

auto measuredTime = stopTime - startTime;

cout << "Measured time is ";

cout << chrono::duration <double, nano>(measuredTime).count() << " ns" << endl;

system("Pause");

return 0;

}

/\* Function to calculate Fibunaci Number //

// if n is 0, then it will return 0 since the first element is 0 //

// if n is 1 then it will return 1 since the first element of //

// the sequence is 0 and the second one is 1, their sum equal to 1 //

// if it is a bigger then 1 then by calling itself the function will //

// calculate the sum of the previous elements \*/

int calculateFibonaci(int n)

{

if (n == 1)

{

return 1;

}

if (n == 0)

{

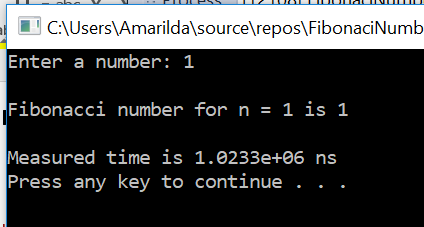
return 0;

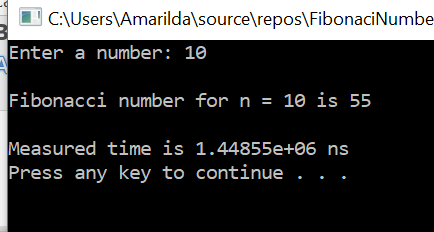
}

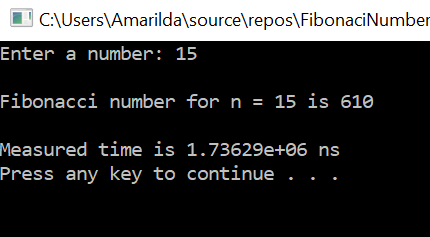
return calculateFibonaci(n - 1) + calculateFibonaci(n - 2);

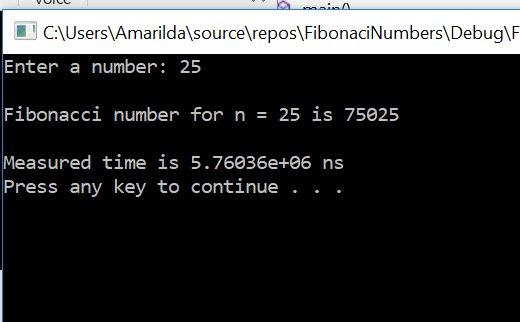
}

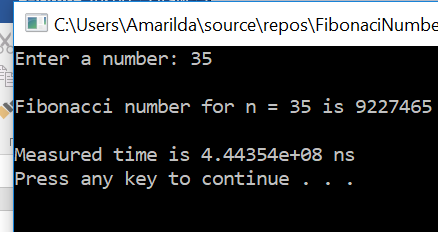
**EXCECUTION TIME**

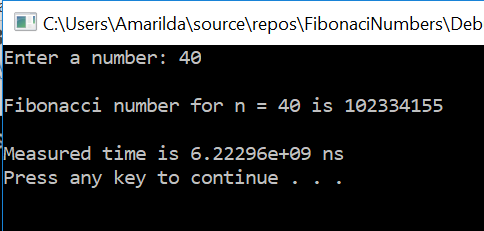












**ITERATIVE FUNCTION SOURCE CODE**

#include "stdafx.h"

#include <iostream>

#include <chrono>

using namespace std;

int fibonacciFunction(int);

int main()

{

cout << "PART B" << endl;

int subscript;

cout << "Enter the subscript: ";

cin >> subscript;

//starting to measure time by enabling steady\_clock

auto startTime = chrono::steady\_clock::now();

//validating user input

while (subscript < 0)

{

cout << "ERROR! Your number must be greater than zero." << endl;

cout << "Enter the subscript again: ";

cin >> subscript;

}

//calling the function to calculate Fibonacci Number

cout << "Fibonaci number of subscript = " << subscript << " is " <<

fibonacciFunction(subscript) << endl;

auto stopTime = chrono::steady\_clock::now();

auto measuredTime = stopTime - startTime;

cout << "Measured time is ";

cout << chrono::duration <double, nano>(measuredTime).count() << " ns" << endl;

system("Pause");

return 0;

}

/\* Calculating Fibonacci Number using Iterative Function //

// Firstpostion holds the value of the first element and //

// second positon holds the value of the second element //

// of the sequence. The sum of the two previous elements //

// will reflect the desired fibonacci number \*/

int fibonacciFunction(int n)

{

int firstPosition = 0,

secondPosition = 1,

fibonacciNumber = 0;

if (n == 1)

{

fibonacciNumber = secondPosition;

}

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

{

fibonacciNumber = firstPosition + secondPosition;

firstPosition = secondPosition;

secondPosition = fibonacciNumber;

}

return fibonacciNumber;

}

**EXCECUTION TO MEASURE TIME**

