

## Números de Fibonacci

Allan Jair Escamilla Hernández

A lo largo del desarrollo de este ejercicio se va a realizar la comparación entre un método recursivo y un método iterativo para calcular los n números de a serie de Fibonacci.

### Pseudocódigo iterativo

```
Principal(){
    fp =
AbrirArchivo("iterative.csv",
"wt");
    counter = 0;
    init = 0, second = 1, num;
    n = 0;
    imprimir("Ingresa n: ");
    leer(n);
    imprimir("0, 1, ");
    start = clock();
    mientras(init <= n){
        num = init + second;
        imprimir(num);
        init = second;
        second = num;
        stop = clock();
        time = (stop-start)/
CLOCKS_PER_SEC;
        imprimirEnArchivo(counter,
time);
        counter++;
    }
```

### Pseudocódigo recursivo

```
Principal(){
    imprimir("Ingresa n: ");
    leer("%d", &n);
    imprimir("0, 1, ");
    start = clock();
    fibo(0, 1, n - 2, 0, start);
    imprimir("\n");
}

fibo(init, second, n, counter,
start){
    si(init <= n){
        stop;
        fp =
abrirArchivo("recursive.csv",
"wt");
        numero = init + second;
        imprimir(numero);
        init = second;
        second = numero;
        stop = clock();
        imprimireEnArchivo(counter,
(double) (stop - start) /
CLOCKS_PER_SEC);
        cerrarArchivo(fp);
        fibo(init, second, n,
counter+1, start);
    }
}
```

## Números de Fibonacci

Allan Jair Escamilla Hernández

### Código del algoritmo iterativo

```
#include <stdio.h>
#include <time.h>
int main(){
    clock_t start, stop;
    double time;
    unsigned long t;
    FILE* fp = fopen("iterative.csv", "wt");
    int counter = 0;
    long long int init = 0, second = 1, num;
    int n = 0;
    printf("Ingresa n: ");
    scanf("%d", &n);
    printf("0, 1, ");
    start = clock();
    while(init <= n){
        num = init + second;
        printf("%lld, ", num);
        init = second;
        second = num;
        stop = clock();
        time = (double)(stop-start)/ CLOCKS_PER_SEC;
        fprintf(fp, "%d, %f\n", counter, time);
        counter++;
    }

    fclose(fp);
    printf("\n");
    return 0;
}
```

## Números de Fibonacci

Allan Jair Escamilla Hernández

### Código del algoritmo recursivo

```
#include<stdio.h>
#include<time.h>

void fibo(long long int init, long long int second, int n, int counter,
clock_t start);

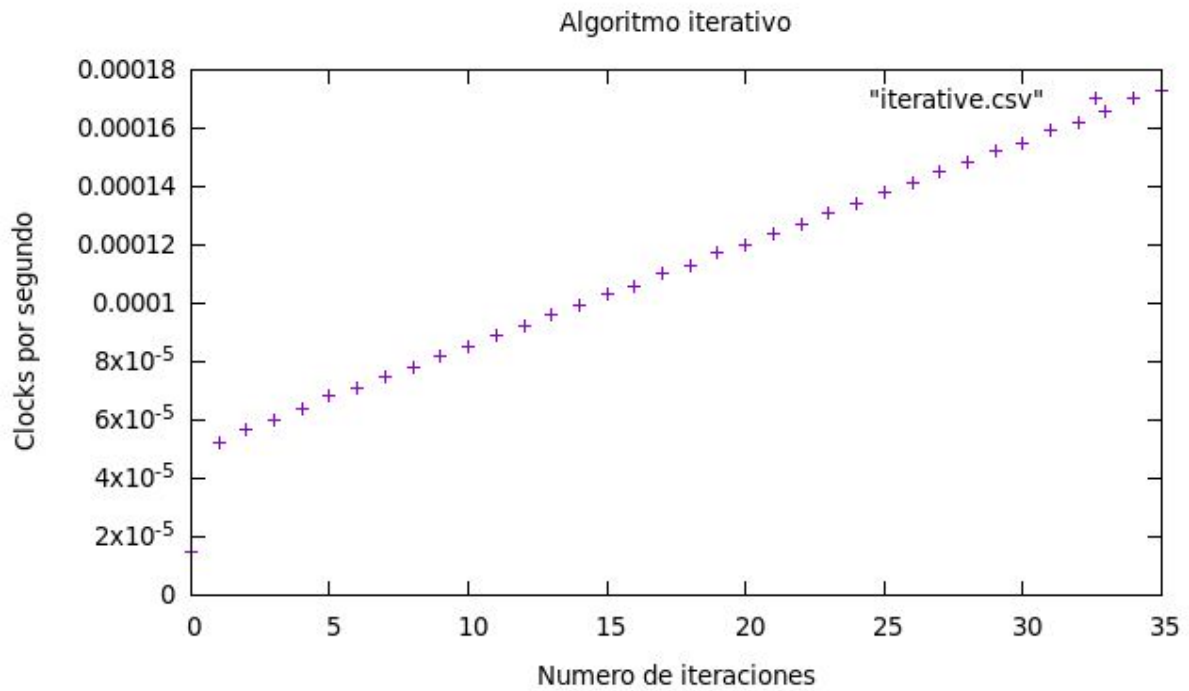
int main(){
    int n;
    clock_t start;
    printf("Ingresa n: ");
    scanf("%d", &n);
    printf("0, 1, ");
    start = clock();
    fibo(0, 1, n - 2, 0, start);
    printf("\n");
    return 0;
}

void fibo(long long int init, long long int second, int n, int counter,
clock_t start){
    if(init <= n){
        clock_t stop;
        FILE* fp = fopen("recursive.csv", "wt");
        long long int numero = init + second;
        printf("%lld, ", numero);
        init = second;
        second = numero;
        stop = clock();
        fprintf(fp, "%d, %f\n", counter, (double) (stop - start) /
CLOCKS_PER_SEC);
        fclose(fp);
        fibo(init, second, n, counter+1, start);
    }
}
```

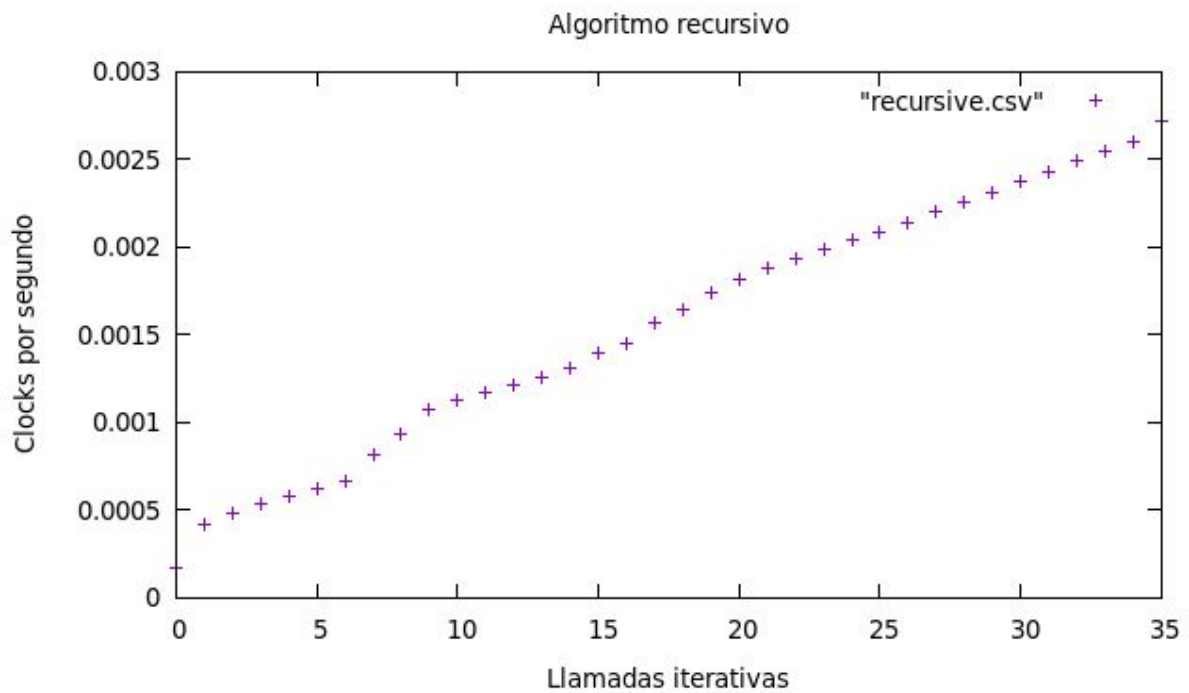
# Números de Fibonacci

Allan Jair Escamilla Hernández

## Gráfica del algoritmo iterativo



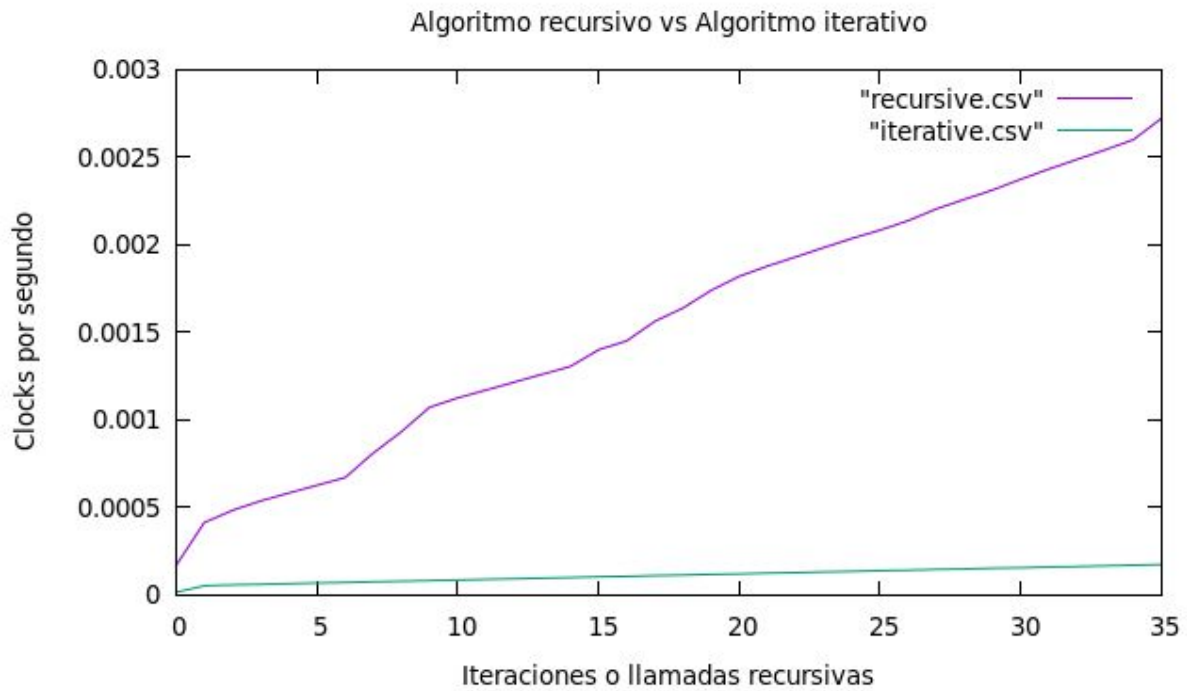
## Gráfica del algoritmo recursivo



## Números de Fibonacci

Allan Jair Escamilla Hernández

### Gráfica de algoritmo recursivo vs algoritmo iterativo



### Conclusión

Al final de este ejercicio puedo concluir que el algoritmo iterativo es sumamente más rápido que el algoritmo recursivo, y esto es debido a que con cada llamada recursiva que hace el algoritmo, tiene que volver a crear las variables y procesos que se generan en la función, por lo que se aumenta la cantidad de procesos que la computadora tiene que ejecutar.