

4-3-2021



# Recursividad

Practica: 1

Materia: Seminario de estructura de datos 1

Sección: D13.

Código: 216584703

Carrera: Ingeniería en computación.

Nombre alumno: Padilla Pérez Jorge Daray

Nombre profesor: Julio Esteban Valdes Lopez

## Índice general

Índice .....	1
Introducción .....	2
Pantallazos .....	3-10
Conclusión .....	11
Código fuente .....	12-15

# Introducción

Mi práctica consiste en la implementación de la recursividad en lenguaje c++, en donde se puede apreciar la integración de la frase: divide y vencerás, además de introducir el llamado de una función dentro de otra (recursividad), mediante estos procesos tomé en cuenta tanto a la serie de Fibonacci, mínimo común divisor y la suma de vectores, para la mejora de la estructura del código, que permite optimizar y simplificar este en la mayoría de los casos del programa a realizar.

Al realizar mi practica obtuve como resultado la correcta integración e implementación de la recursividad dentro de cualquier código, en el cual es necesario asegurar que cada llamada recursiva debe estar definida sobre un problema menos complejo que el que dio lugar a la llamada, y que debe existir al menos un caso base al que se garantice la llegada del problema en un momento determinado, como el que yo diseñe, al cual hago referencia en la página 12 logrando obtener un mejor conocimiento del tema.

# Pantallazos

```
1  #include <iostream>
2  #include <cstring>
3  #include <cstdlib>
4  #include <string>
5  #include <time.h>
6  using namespace std;
7
8  void menu();
9  void mostrar_fibo();
10 long fibonacci (int);
11 int MCD(int,int);
12 void mostrar_mcd();
13 int suma_datos_vector(int[],int);
14 void mostrar_suma();
15
16 int main()
17 {
18     int opc=0;
19     do{
20         system("cls");
21         menu();
22         cout<<"Continuar 1 salir 2:"<<endl;cin>>opc;
23     }while(opc!=2);
24     system("pause>>cls");
25     return 0;
26 }
```

Aquí se hace la inicialización de las funciones principales como el menú, la serie de Fibonacci etc.

También se pueden apreciar las librerías, además de poder observar un bucle do-while en el cual se hace llamado a la función menú, además de limpiar la pantalla luego de imprimir el menú.

```
28 long fibonacci(int n){
29     if (1 == n || 2 == n) {
30         return 1;
31     } else {
32         return (fibonacci(n-1) + fibonacci(n-2));
33     }
34 }
```

En este fragmento de código se aprecia la primera función llamada Fibonacci con parámetro de un entero llamado 'n', además de aplicar la recursividad al implementar la función Fibonacci dentro de ella misma, con un

parámetro 'if' que define los casos base, para su correcta implementación.

```
36 void menu() {
37     int opc=0;
38     cout<<" Practica 1"<<endl;
39     cout<<"1) fibonacci"<<endl;
40     cout<<"2) MCD"<<endl;
41     cout<<"3) Suma de vectores"<<endl;
42     cout<<"4) salir"<<endl;
43     cout<<"Seleccione opcion:"<<endl;
44     cin>>opc;
45     switch(opc) {
46
47         case 1: {
48             mostrar_fibo();
49             }break;
50
51         case 2: {
52             mostrar_mcd();
53             }break;
54         case 3: {
55             mostrar_suma();
56             }break;
57     }
```

En este fragmento de código se implementa el menú el cual contiene 4 opciones las cuales le aparecerán al usuario que ejecute el programa, y en su interior (parte que no ve el usuario) un switch que permite dependiendo la opción que tome el usuario abrir las 3 funciones de mostrar al usuario, además de una opción de salir del programa (cerrarlo).

Aquí se mira la función que muestra el resultado de la función implementada que se aprecia anteriormente además de dejarle ingresar al usuario el número positivo del cual quiera saber su resultado en la serie de Fibonacci.

```
59 void mostrar_fibo()
60 {
61     cout<<"::serie de fibonacci::"<<endl;
62     int n;
63     cout<<"Introduce un numero positivo: "<<endl;cin>> n;
64     cout<<"El numero " <<n<<" de Fibonacci es :"<<fibonacci(n)<<endl;
65 }
```

```
67 int MCD(int x, int y)
68 {
69     if(y==0)
70         return x;
71     else
72         return MCD(y, x%y);
73 }
74
75 void mostrar_mcd()
76 {
77     int num1=0,num2=0;
78     cout<<"::Maximo Comun Divisor::"<<endl;
79     cout<<"Introduce el primer numero: "<<endl;cin>>num1;
80     cout<<"Introduce el segundo numero: "<<endl;cin>>num2;
81     cout<<"El resultado es: "<<MCD(num1, num2)<<endl;
82 }
```

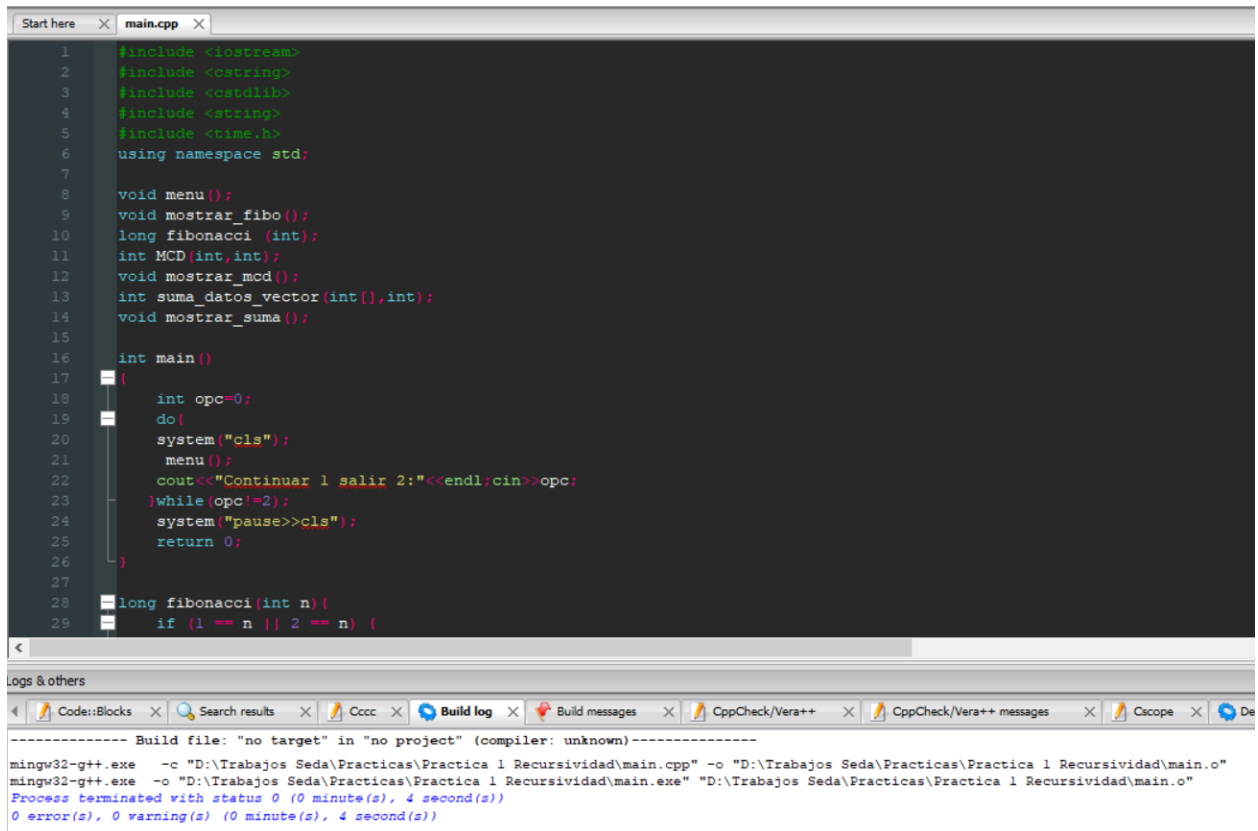
Implementación de la función Mínimo Común Divisor el cual recibe 2 parámetros de tipo entero seguido de una condición 'if' que indica el caso base, regresando el resultado dependiendo de lo que ingrese el usuario, también se aprecia la función que muestra al usuario la opción de introducir el primer y segundo número del máximo común divisor, como el resultado después de ingresarlos.

```
84 int suma_datos_vector(int vec[],int n){
85     if (n == 0){
86         return vec[n];
87     }else{
88         n--;
89         return vec[n+1] + suma_datos_vector(vec, n);
90     }
91 }
92
93
94 void mostrar_suma()
95 {
96     cout<<"::Suma de vectores::"<<endl;
97     int n;
98     cout<< "ingresa el valor de n: "<<endl;cin>>n;
99     int vec[n];
100     srand(time(NULL));
101     for (int i=0;i < n;i++){
102         vec[i] =(int) ((rand()/32767.1)*1000);
103         cout<<vec[i]<<endl;
104     }
105     cout<<"la suma de datos del vector es: "<<suma_datos_vector(vec, n-1)<<endl;
106 }
```

Aquí como es los anteriores pantallazos este pedazo de código desarrolla la función de sumar vectores al azar más sus respectivos parámetros de implementar la recursividad como en los casos anteriores, además de tener una función llamada mostrar como las otras, la cual permite que el usuario ingrese la cantidad de números random que quiera generar y sumar, utilizando la librería time.h para poder utilizar 'srand (time(NULL))' el cual hace el random y lo incluye en un for para que se repite las veces que sean necesarias para que el usuario tenga los números que introdujo que quería, y al final se implementa un rand el cual devuelve un número aleatorio que este adentro de el limite especifico el cual aparece ahí, para terminar se muestra el resultado al usuario.

Cabe resaltar que todo está implementado en funciones para aplicar el llamado divide y vencerás.

## | Pantallazos



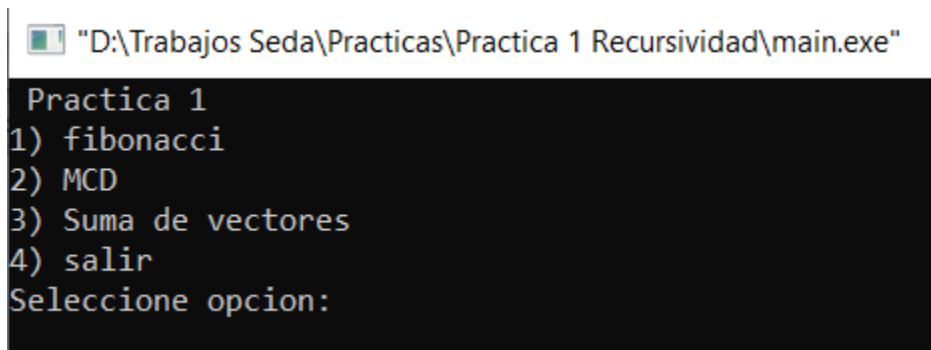
The screenshot shows a C++ IDE with two panes. The top pane displays the source code of 'main.cpp', which includes headers for <iostream>, <string>, <cstdlib>, <string>, and <time.h>, and uses the std namespace. It defines several functions: menu(), mostrar\_fibo(), fibonacci(int), MCD(int, int), mostrar\_mcd(), suma\_datos\_vector(int[], int), and mostrar\_suma(). The main function is a loop that calls menu() and prompts the user to continue (1) or exit (2). The bottom pane shows the build log, indicating that the file was built successfully with no errors or warnings.

```
1 #include <iostream>
2 #include <string>
3 #include <cstdlib>
4 #include <string>
5 #include <time.h>
6 using namespace std;
7
8 void menu();
9 void mostrar_fibo();
10 long fibonacci (int);
11 int MCD(int,int);
12 void mostrar_mcd();
13 int suma_datos_vector(int[],int);
14 void mostrar_suma();
15
16 int main()
17 {
18     int opc=0;
19     do{
20         system("cls");
21         menu();
22         cout<<"Continuar 1 salir 2:"<<endl;cin>>opc;
23     }while(opc!=2);
24     system("pause>>cls");
25     return 0;
26 }
27
28 long fibonacci(int n){
29     if (1 == n || 2 == n) {
```

Build log:

```
----- Build file: "no target" in "no project" (compiler: unknown)-----
mingw32-g++.exe -c "D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.cpp" -o "D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.o"
mingw32-g++.exe -o "D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe" "D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.o"
Process terminated with status 0 (0 minute(s), 4 second(s))
0 error(s), 0 warning(s) (0 minute(s), 4 second(s))
```

Aquí se aprecia que no arroja ningún error y esta listo para su funcionamiento




The screenshot shows a Windows command prompt window titled "D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe". The program output is as follows:

```
Practica 1
1) fibonacci
2) MCD
3) Suma de vectores
4) salir
Seleccione opcion:
```


Esta es la consola donde se muestran las opciones que puede elegir el usuario.



 "D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe"

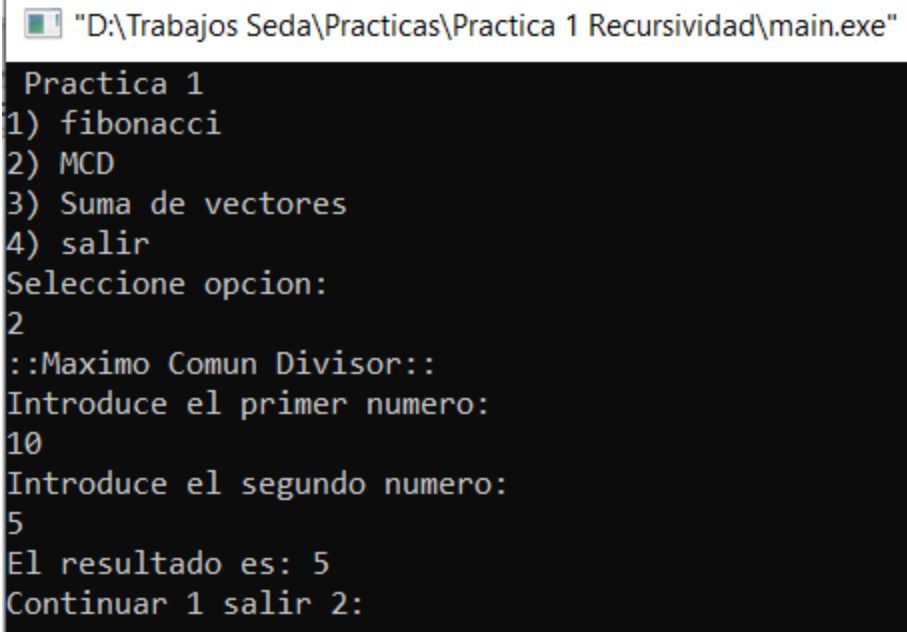
```
Practica 1
1) fibonacci
2) MCD
3) Suma de vectores
4) salir
Seleccione opcion:
1
::serie de fibonacci::
Introduce un numero positivo:
5
El numero 5 de Fibonacci es :5
Continuar 1 salir 2:
```

Al seleccionar la opción 1 abre la opción Fibonacci el cual te pide un numero positivo, al momento de introducir un 5 como se aprecia ahí, el resultado es 5 como debe ser.

 "D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe"

```
Practica 1
1) fibonacci
2) MCD
3) Suma de vectores
4) salir
Seleccione opcion:
1
::serie de fibonacci::
Introduce un numero positivo:
10
El numero 10 de Fibonacci es :55
Continuar 1 salir 2:
```

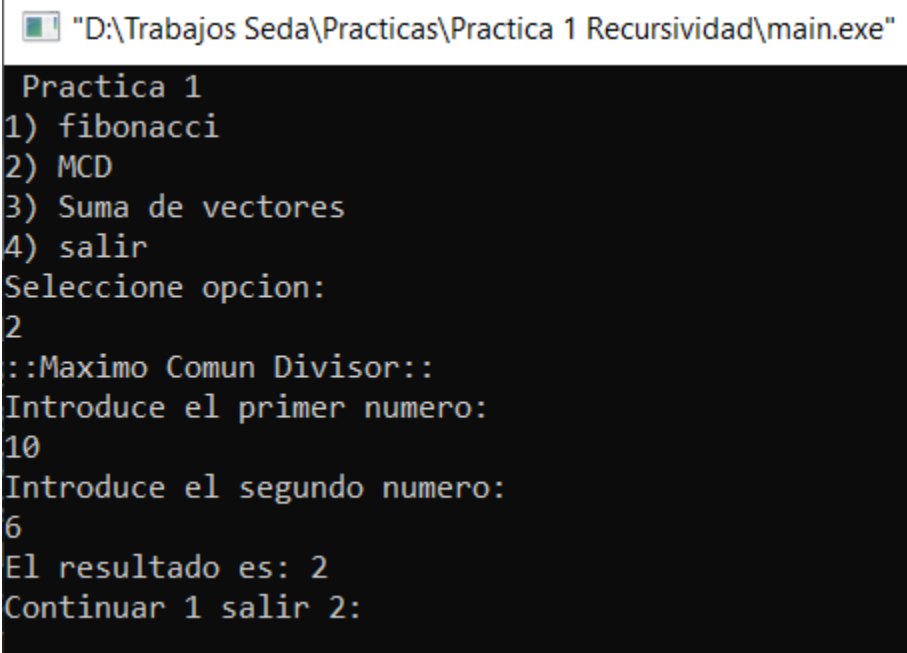
Aquí con otro numero para comprobar que funciona.



```
"D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe"

Practica 1
1) fibonacci
2) MCD
3) Suma de vectores
4) salir
Seleccione opcion:
2
::Maximo Comun Divisor::
Introduce el primer numero:
10
Introduce el segundo numero:
5
El resultado es: 5
Continuar 1 salir 2:
```

Al seleccionar la opción 2 abre el menú el cual te pide introducir el primer y segundo numero como pueden ver el resultado es correcto.



```
"D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe"

Practica 1
1) fibonacci
2) MCD
3) Suma de vectores
4) salir
Seleccione opcion:
2
::Maximo Comun Divisor::
Introduce el primer numero:
10
Introduce el segundo numero:
6
El resultado es: 2
Continuar 1 salir 2:
```

Otro numero para comprobar su funcionamiento.

```
"D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe"
Practica 1
1) fibonacci
2) MCD
3) Suma de vectores
4) salir
Seleccione opcion:
3
::Suma de vectores::
ingresa el valor de n:
5
939
997
160
567
426
la suma de datos del vector es: 3089
Continuar 1 salir 2:
```

Aquí selecciono la opción 3 y te pide ingresar la cantidad de números que quieres sumar, cabe resaltar que son números al azar con ciertos parámetros ya explicados y funciona bien.

```
"D:\Trabajos Seda\Practicas\Practica 1 Recursividad\main.exe"
Practica 1
1) fibonacci
2) MCD
3) Suma de vectores
4) salir
Seleccione opcion:
3
::Suma de vectores::
ingresa el valor de n:
5
951
655
397
830
402
la suma de datos del vector es: 3235
Continuar 1 salir 2:
```

En este segundo es para confirmar que son al azar y que funciona bien.

# Conclusión

Respecto a la realización del código concluyo que logré el entendimiento correcto del tema mediante la correcta implementación de esta, se entendió el tema de la recursividad tanto de su sintaxis, como la lógica para su implementación, también aprendí los fallos que puede dar a la hora de correr el código y por ende sus posibles soluciones.

Además de aprender la teoría necesaria para darme cuenta de lo funcional que es, y poder identificar cuando conviene utilizar la recursividad y si se puede resolver el problema utilizándola.

# Código fuente

```
#include <iostream>

#include <cstring>

#include <cstdlib>

#include <string>

#include <time.h>

using namespace std;


void menu();

void mostrar_fibo();

long fibonacci (int);

int MCD(int,int);

void mostrar_mcd();

int suma_datos_vector(int[],int);

void mostrar_suma();


int main()
{
    int opc=0;

    do{
        system("cls");

        menu();

        cout<<"Continuar 1 salir 2:"<<endl;cin>>opc;
    }while(opc!=2);

    system("pause>>cls");

    return 0;
```

| Código fuente

```
}
```

```
long fibonacci(int n){  
    if (1 == n || 2 == n) {  
        return 1;  
    } else {  
        return (fibonacci(n-1) + fibonacci(n-2));  
    }  
}
```

```
void menu(){  
int opc=0;  
    cout<<" Practica 1"<<endl;  
    cout<<"1) fibonacci"<<endl;  
    cout<<"2) MCD"<<endl;  
    cout<<"3) Suma de vectores"<<endl;  
    cout<<"4) salir"<<endl;  
    cout<<"Seleccione opcion:"<<endl;  
    cin>>opc;  
    switch(opc){  
  
        case 1: {  
            mostrar_fibo();  
        }break;  
    case 2: {  
        mostrar_mcd();  
    }break;  
    case 3: {  
        mostrar_suma();  
    }
```

| Código fuente

```
        }break;
    }
}

void mostrar_fibo()
{
    cout<<"::serie de fibonacci::"<<endl;
    int n;
    cout<<"Introduce un numero positivo: "<<endl;cin>> n;
    cout<<"El numero " <<n<<" de Fibonacci es :"<<fibonacci(n)<<endl;
}

int MCD(int x, int y)
{
    if(y==0)
        return x;
    else
        return MCD(y, x%y);
}

void mostrar_mcd()
{
    int num1=0,num2=0;
    cout<<"::Maximo Comun Divisor::"<<endl;
    cout<<"Introduce el primer numero: "<<endl;cin>>num1;
    cout<<"Introduce el segundo numero: "<<endl;cin>>num2;
    cout<<"El resultado es: "<<MCD(num1, num2)<<endl;
}
```

```
int suma_datos_vector(int vec[],int n){  
    if (n == 0){  
        return vec[n];  
    }else{  
        n--;  
        return vec[n+1] + suma_datos_vector(vec, n);  
    }  
  
}  
  
void mostrar_suma()  
{  
    cout<<"::Suma de vectores::"<<endl;  
    int n;  
    cout<< "ingresa el valor de n: "<<endl;cin>>n;  
    int vec[n];  
    srand(time(NULL));  
    for (int i=0;i < n;i++){  
        vec[i] =(int)((rand())/32767.1)*1000);  
        cout<<vec[i]<<endl;  
    }  
    cout<<"la suma de datos del vector es: "<<suma_datos_vector(vec, n-1)<<endl;  
}
```