

UNIVERSIDAD DE GUANAJUATO

DIVISIÓN DE INGENIERÍAS CAMPUS IRAPUATO-SALAMANCA

LIC. EN ING. EN SISTEMAS COMPUTACIONALES

ALGORITMOS Y ESTRUCTURAS DE DATOS

PROFESOR: DR. CARLOS HUGO GARCÍA CAPULÍN

NO. DE TAREA: 02

NOMBRE DE LA TAREA:

**SOLUCIÓN DE PROBLEMAS USANDO
ESTRATEGIAS ITERATIVAS Y RECURSIVAS**

ESTUDIANTE:

MANRÍQUEZ COBIÁN ROGELIO

FECHA DE ENTREGA:

01 DE SEPTIEMBRE DEL 2020



Problema

Se tiene que dar la solución a varios problemas de Sumatoria, Factorial y Fibonacci de “N” números naturales implementando dos métodos de la forma “Iterativa y Recursividad”, esto para analizar y comprender como se maneja cada código y ver que en diferentes maneras hay solución a estos problemas a realizar.

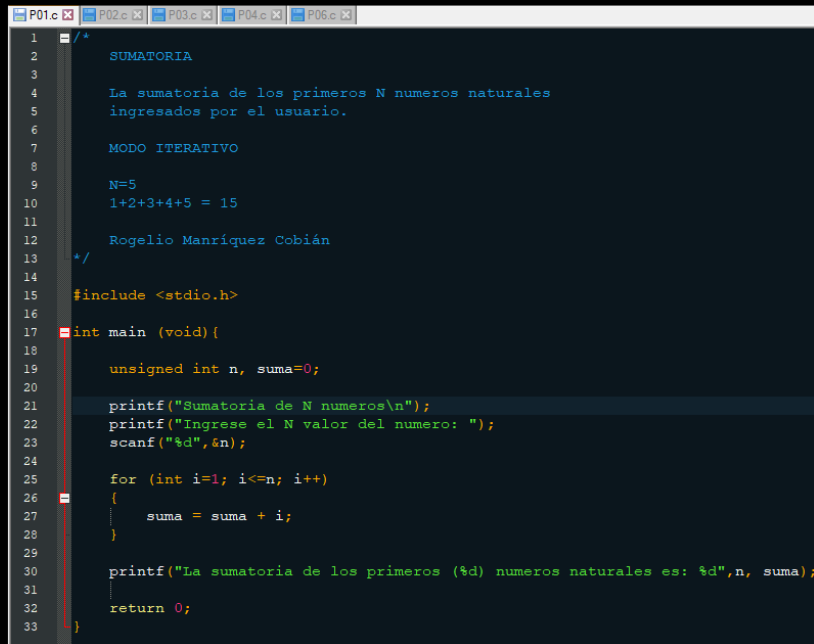
Solución Implementada

Para realizar la solución ante estos problemas en el lenguaje C, siempre es necesario iniciar con una plantilla con la cual nos facilitará realizar el código, para esto es necesario abrir nuestro editor de texto y guardar el archivo con el nombre que gustes (siempre y cuando tenga sentido con lo que vas a programar) con la extensión “.c”. Entonces empezamos a escribir una pequeña descripción de lo que hará nuestro código y después de ello, escribir el “header” con la librería de entrada y salida “#include <stdio.h>” y nuestra función “int main ()” para poder empezar a escribir código.

Estos sencillos pasos se tienen que realizar en cada programa.

PROBLEMA 1

El siguiente problema presenta en resolver el siguiente problema de una sumatoria de N números naturales que proporciona el usuario y se tiene que resolver de manera ITERATIVA.



```

1  /*
2  SUMATORIA
3
4  La sumatoria de los primeros N numeros naturales
5  ingresados por el usuario.
6
7  MODO ITERATIVO
8
9  N=5
10 1+2+3+4+5 = 15
11
12 Rogelio Manríquez Cobián
13 */
14
15 #include <stdio.h>
16
17 int main (void){
18
19     unsigned int n, suma=0;
20
21     printf("Sumatoria de N numeros\n");
22     printf("Ingrese el N valor del numero: ");
23     scanf("%d",&n);
24
25     for (int i=1; i<=n; i++)
26     {
27         suma = suma + i;
28     }
29
30     printf("La sumatoria de los primeros (%d) numeros naturales es: %d",n, suma);
31
32     return 0;
33 }

```

Para comenzar, debemos declarar dos variables “unsigned int” al comienzo de nuestra función main, las cuales son:

- “n” para poder almacenar el número que el usuario digite.
- “suma” inicializada en ceros, donde almacenaremos la suma de cada numero

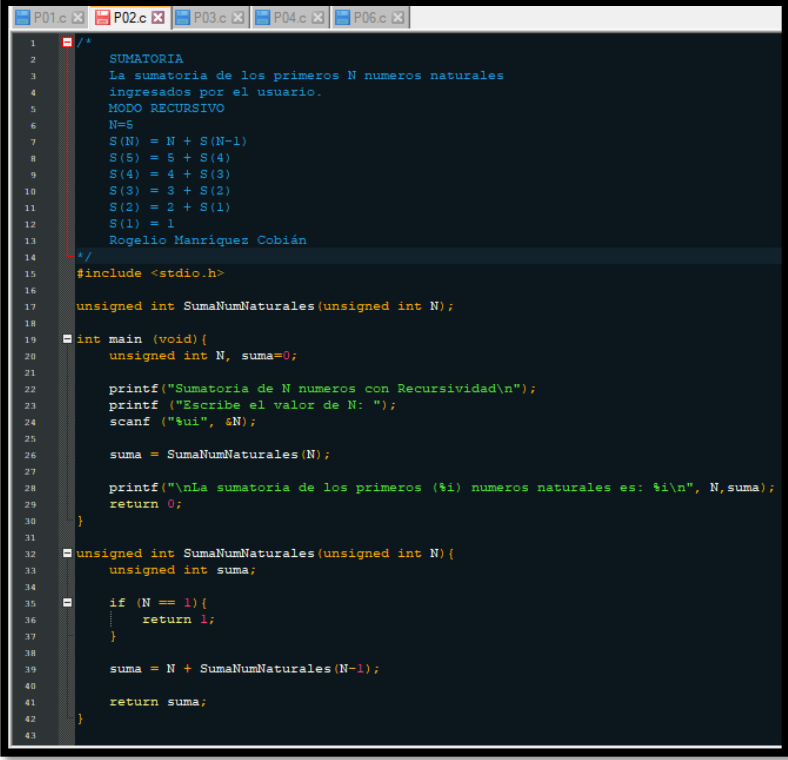
Ya de una vez que tenemos estas variables, debemos pedir al usuario que ingrese el valor de “n” para calcular la suma de estos, y poder asignárselos a la variable “n”. Ya que guardamos este valor, haremos un ciclo for sencillo, donde el valor comienzo sería “int i=0”, e indicar que se detenga cuando “i<n”, para eso también debemos de incrementar “i++”. Cada ciclo de nuestra iteración estará realizando la operación de “suma = suma + i” para cuando el ciclo termine, se obtendrá la suma total de los “n” números ingresados por el usuario.

Para finalizar nuestro código, debemos imprimir estos valores en pantalla de la suma que hicimos en nuestro ciclo, por lo cual se debe hacer una referencia de “n” que se ingresó y el resultado de suma para poder ver el resultado final.

NOTA: El ejemplo de este código viene comentado en la imagen.

PROBLEMA 2

Para la realizar la forma recursiva de este problema de sumatoria de N números naturales, se explicará en el siguiente código.



```

1  /*
2  SUMATORIA
3  La sumatoria de los primeros N numeros naturales
4  ingresados por el usuario.
5  MODO RECURSIVO
6  N=5
7  S(N) = N + S(N-1)
8  S(5) = 5 + S(4)
9  S(4) = 4 + S(3)
10 S(3) = 3 + S(2)
11 S(2) = 2 + S(1)
12 S(1) = 1
13 Rogelio Manriquez Cobián
14 */
15 #include <stdio.h>
16
17 unsigned int SumaNumNaturales(unsigned int N);
18
19 int main (void){
20     unsigned int N, suma=0;
21
22     printf("Sumatoria de N numeros con Recursividad\n");
23     printf("Escribe el valor de N: ");
24     scanf ("%i", &N);
25
26     suma = SumaNumNaturales(N);
27
28     printf("\nLa sumatoria de los primeros (%i) numeros naturales es: %i\n", N,suma);
29     return 0;
30 }
31
32 unsigned int SumaNumNaturales(unsigned int N){
33     unsigned int suma;
34
35     if (N == 1){
36         return 1;
37     }
38
39     suma = N + SumaNumNaturales(N-1);
40
41     return suma;
42 }
43

```

Se observa claramente que se han declarado variables “unsigned int” para poder realizar las acciones de nuestro código. De nuevo se tiene que pedir que el usuario ingrese el número “N” y poder guardarlo en nuestra variable; esto para que se pueda guardar en nuestra variable de suma y vaya a la función que se creó, llamada “SumaNumNaturales”

NOTA: En este punto ya no se utilizará el ciclo for (manera iterativa)

La función “SumaNumNaturales” recibe el valor de “N”, que se había mandado desde un principio en la función main, estando dentro de la función se debe de declarar una variable suma (diferente de main) para poder realizar nuestro algoritmo de recursividad. Entonces si nuestro valor de “N” es igual a “1” debemos de regresar este mismo valor, de lo contrario este valor se le asigna a nuestra línea de código ($\text{suma} = N + \text{SumaNumNaturales}(N-1)$) para que pueda hacer la recursividad, este proceso se realizará varias veces hasta que la función regrese “1”. Cuando la condición se cumple todas las funciones recursivas deben de tener algún valor y se empieza a realizar una suma de esto; en la cual todo se guarda en la variable “suma” y se retorna a nuestra función main y poder imprimir en pantalla la sumatoria.

NOTA: El ejemplo de este código viene comentado en la imagen.

PROBLEMA 3

Ahora se analizará la solución de un problema de factoriales de los primeros N números naturales ingresados por el usuario de manera ITERATIVA.

```

1  /*
2  FACTORIAL
3
4  El factorial de los primeros N numeros naturales
5  ingresados por el usuario.
6
7  MODO ITERATIVO
8
9  N=5
10 1*2*3*4*5=120
11
12 Rogelio Manríquez Cobián
13 */
14
15 #include <stdio.h>
16
17 int main (void){
18     unsigned int N, fact=1;
19
20     printf("Factorial de N numeros\n");
21     printf ("\nEscribe el valor de N: ");
22     scanf ("%ui", &N);
23
24     for (int i=2; i<=N; i++)
25     {
26         fact = fact * i;
27     }
28
29     printf("\nEl factorial de (%i) es: %i\n", N,fact);
30     return 0;
31 }

```

Llegando en este punto, se puede notar que de igual manera es la plantilla que se utilizó cuando se realizó el ejercicio de sumatoria de forma iterativa.

Como se ve, se ha declarado una variable “unsigned int” llamada “fact” inicializada en “1” ya que matemáticamente el factorial de “1 o 0” siempre es igual a “1”.

De nueva manera, pedimos que el usuario ingrese el valor de “N” y poder guardarlo. Ahora comenzamos con nuestro ciclo for iniciado en “2” ya que como se mencionó el factorial de “1 o 0” es uno, nuestra variable de control es “i<=N” y en cada iteración incrementará en uno.

Entonces, dentro del ciclo se empezará a realizar la operación de “fact = fact * i”, lo que en pocas palabras realiza la acumulación del número ingresado y se multiplique por “i” en cada iteración.

Finalmente, se detendrá el ciclo, se guardará y se imprimirá en pantalla este valor del factorial.

NOTA: El ejemplo de este código viene comentado en la imagen.

PROBLEMA 4

```

1  /*
2  FACTORIAL
3  El factorial de los primeros N numeros naturales
4  ingresados por el usuario.
5  MODO RECURSIVO
6  N=5
7  F(N) = N * F(N-1)
8  F(5) = 5 * F(4) -> 120
9  F(4) = 4 * F(3)
10 F(3) = 3 * F(2)
11 F(2) = 2 * F(1)
12 F(1)
13 Rogelio Manríquez Cobián
14 */
15 #include <stdio.h>
16
17 unsigned FactNumNaturales(unsigned int N);
18
19 int main (void){
20     unsigned int N, fact=1;
21
22     printf("Factorial de N numeros con Recursividad\n");
23     printf ("Escriba el valor de N: ");
24     scanf ("%ui", &N);
25
26     int resultado = FactNumNaturales(N);
27
28     printf("\nEl factorial de (%i) es: %i\n", N, resultado);
29     return 0;
30 }
31
32 unsigned FactNumNaturales(unsigned int N){
33     unsigned int resultado;
34
35     if (N==1)
36     {
37         return 1;
38     }
39
40     resultado = N * FactNumNaturales(N-1);
41
42     return resultado;
43 }

```

Ahora chequearemos el mismo problema del factorial de la manera recursiva.

Como se observa, casi tiene la misma estructura de la forma iterativa del ejercicio anterior.

De igual manera se han declarado las mismas variables del ejemplo anterior, por consiguiente, se debe de introducir el valor de “N” por el usuario y poder guardarla en la variable.

Ahora este dato capturado, se debe de igualar a una función llamada “FactNumNatural” que se encuentra en nuestro main.

Al llegar a nuestra función, debemos de crear una variable “resultado” para poder almacenar nuestro algoritmo.

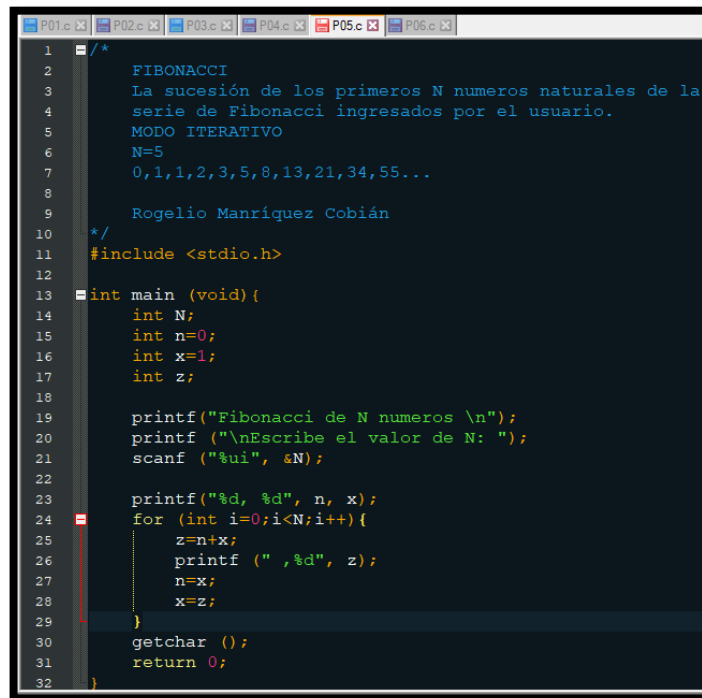
Ahora debemos de hacer una condición donde si el valor de “N” es idéntico a “1” debe de retornar este mismo valor, si el valor es mayor a uno dependiendo lo que haya ingresado el usuario, se hará la recursividad del problema con el siguiente algoritmo ($\text{resultado} = N * \text{FactNumNaturales}(N-1)$) ya que esta función estará de manera cíclica hasta que el valor de N sea igual a “1” para que después estas funciones trabajen como producto de manera inversa y así poder tener el valor de este factorial que se encuentra almacenado en la variable resultado y que irá directamente a la función main.

Por último, se tiene que imprimir este valor en pantalla para visualizar el resultado de nuestro programa.

NOTA: El ejemplo de este código viene comentado en la imagen.

PROBLEMA 5

Se analizará el siguiente problema de la sucesión/suma de la serie de Fibonacci.



```

1  /*
2     FIBONACCI
3     La sucesión de los primeros N numeros naturales de la
4     serie de Fibonacci ingresados por el usuario.
5     MODO ITERATIVO
6     N=5
7     0,1,1,2,3,5,8,13,21,34,55...
8
9     Rogelio Manríquez Cobián
10  */
11  #include <stdio.h>
12
13  int main (void){
14      int N;
15      int n=0;
16      int x=1;
17      int z;
18
19      printf("Fibonacci de N numeros \n");
20      printf ("\nEscribe el valor de N: ");
21      scanf ("%ui", &N);
22
23      printf("%d, %d", n, x);
24      for (int i=0;i<N;i++){
25          z=n+x;
26          printf (" ,%d", z);
27          n=x;
28          x=z;
29      }
30      getchar ();
31      return 0;
32  }

```

Se han declarado 4 variables de tipo “int” ya que la primera variable será “N” la cual tendrá como función solicitar y guardar el dato que el usuario ingrese, el siguiente es “n=0” ya que esta nos servirá para tener un control en nuestro ciclo for e imprimir en pantalla la primera posición, ahora tenemos “x=1” el cual también nos ayudará en el ciclo for e imprimir la segunda posición de la sucesión y por último “z” donde se estará guardando la suma de los números anteriores de “n” y “x”.

Lo primero que hacemos es pedir el valor de “N” al usuario y guardarlo; después imprimiremos en pantalla las variables “n” y “x” que harán comienzo en nuestra sucesión. Luego se realizará un ciclo for iniciado en cero, con la variable de control “i<N” y en cada iteración esta se incrementará.

Ahora dentro de nuestro ciclo for, guardaremos en la variable “z” la suma entre los números de “n” y “x”, después de esto, se hace un cambio de valor en la variable “n” en la cual tomará el valor de “x” y “x” tomará el valor de la suma en “z”, esto se hará en cada iteración hasta que el ciclo pare.

Por último se imprimirá el valor nuevo que va obteniendo “z” en nuestro ciclo, mostrando así la sucesión pedida por el usuario.

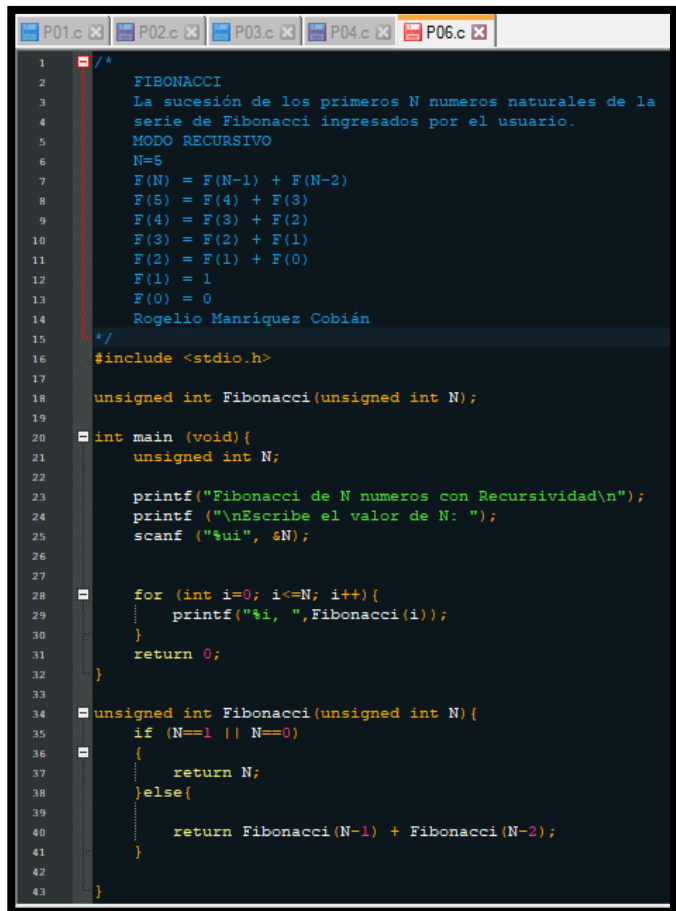
NOTA: El ejemplo de este código viene comentado en la imagen.

PROBLEMA 6

Ahora se analizará el siguiente problema de Fibonacci de la manera recursiva donde el usuario ingresa el valor de "N" números naturales para visualizar la sucesión.

Se tendrá que declarar una variable tipo "unsigned int" la cual será "N", para que el usuario ingrese el valor de esta y poder guardarla.

Este valor de "N" irá a nuestra función llamada "Fibonacci", en la cual se tendrá que realizar la condición de que ($N == 1$ || $N == 0$) este regresará la misma variable, ya que el primer valor de la serie de Fibonacci comienza en "0" y que la siguiente suma de esta sucesión es "1". Sino se cumple esta condición, el valor de "N" irá a nuestro algoritmo creado el cual no se declaró una variable para guardar, ya que directamente esta se puede retornar a nuestra función main.



```

1  /*
2  FIBONACCI
3  La sucesión de los primeros N numeros naturales de la
4  serie de Fibonacci ingresados por el usuario.
5  MODO RECURSIVO
6  N=5
7  F(N) = F(N-1) + F(N-2)
8  F(5) = F(4) + F(3)
9  F(4) = F(3) + F(2)
10 F(3) = F(2) + F(1)
11 F(2) = F(1) + F(0)
12 F(1) = 1
13 F(0) = 0
14 Rogelio Manriquez Cobián
15 */
16 #include <stdio.h>
17
18 unsigned int Fibonacci(unsigned int N);
19
20 int main (void){
21     unsigned int N;
22
23     printf("Fibonacci de N numeros con Recursividad\n");
24     printf ("Escribe el valor de N: ");
25     scanf ("%ui", &N);
26
27
28     for (int i=0; i<=N; i++){
29         printf("%i ", Fibonacci(i));
30     }
31     return 0;
32 }
33
34 unsigned int Fibonacci(unsigned int N){
35     if (N==1 || N==0)
36     {
37         return N;
38     }else{
39         return Fibonacci(N-1) + Fibonacci(N-2);
40     }
41 }
42
43

```

Lo que hace el algoritmo ($Fibonacci(N-1) + Fibonacci(N-2)$) es evaluar el valor de N y restarle "1", cuando esto sucede se obtiene un valor nuevo de "N" el cual ahora se estará evaluando en "N-2" y retornar un valor de "N" a la función, hasta que recursividad pueda terminar en "0" y así poder terminar este ciclo, el cual se mandará a nuestra función main.

En la función main, se hará un ciclo for de manera ITERATIVA, inicializada en "0", nuestra variable de control será " $i \leq 5$ ", y cada iteración se incrementará en uno, después de esto, imprimiremos en pantalla todos los valores que se realizaron en la función "Fibonacci", pero la variable a utilizar será la misma función evaluada en "i" y así visualizar toda la sucesión que quiso el usuario.

NOTA: El ejemplo de este código viene comentado en la imagen.

Pruebas y Resultados

Con los resultados que obtuvimos con los códigos anteriores queda comprendido el tema de “*SOLUCIÓN DE PROBLEMAS USANDO ESTRATEGIAS ITERATIVAS Y RECURSIVAS*”, las cuales son muy funcionales para un programador.

Estas estrategias siempre son vistas en la gran mayoría de los códigos de las personas ya que sin estas estrategias sería algo difícil de programar. De igual manera estas estrategias dependen de ventajas y desventajas en proyectos muy grandes que un programador realice, ya que estas estrategias en ocasiones no son muy efectivas porque pueden ser algo lentas; pero siempre es bueno comprender esto, para poder realizar más algoritmos que nos ayuden a solucionar problemas de este modo.