



INSTITUTO POLITÉCNICO
NACIONAL



ESCUELA SUPERIOR DE CÓMPUTO

ESTRUCTURAS DE DATOS

Práctica 3: Recursividad

Alumno:

Ramírez Cotonieto Luis Fernando.

Grupo: 1CV3

Profesor: Yaxkin Flores Mendoza

CDMX, México

15/Diciembre/2020

Practica 3

Estructura de Datos

1. Introducción

Se llama recursividad a un proceso mediante el que una función se llama a sí misma de forma repetida, hasta que se satisface alguna determinada condición. El proceso se utiliza para computaciones repetidas en las que cada acción se determina mediante un resultado anterior. Se pueden escribir de esta forma muchos problemas iterativos.

En matemáticas se da el nombre de recursión a la técnica consistente en definir una función en términos de sí misma. Puesto que en C una función puede llamar a otras funciones, se permite que una función también pueda llamarse a sí misma.

Toda función definida recursivamente debe contener al menos una definición explícita para alguno de sus argumentos. De no ser así la función puede caer en un bucle infinito.

2. Código

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <math.h>
4
5  void binario(int tam){
6
7      int c;
8      char num[tam];
9
10     fflush(stdin);
11
12     printf("Numero en binario:\n");
13     scanf("%s",num);
14
15     c = conv_binario(num,tam-1,tam-1);
16     printf("Numero convertido: %d",c);
17 }
18
19 void convertir_binario(int num){
20     int divid,resto;
21
22     divid = num/2;
23     resto = num%2;
24
25     if(divid == 0)
26         printf("%d",resto);
27     else{
28         convertir_binario(divid);
29         printf("%d",resto);
30     }
31 }
32
33 int conv_binario(char* cadena,int potencia, int tam){
34     int dec = 0, posicion = tam-potencia;
35     if(potencia == 0){
36         if(cadena[posicion] == '1')
```

```

37         return 1;
38     else
39         return 0;
40 }
41 else{
42     if(cadena[posicion] == '1')
43         dec = pow(2,potencia);
44     --potencia;
45
46     return dec += conv_binario(cadena,potencia,tam);
47 }
48 }
49
50 int factorial(int numero){
51     if(numero == 1 || numero == 0)
52         return numero;
53     else
54         return numero*factorial(numero-1);
55 }
56
57 long fib( unsigned long pos){
58     printf("f(%d) ",pos);
59     if (pos == 0 || pos == 1)
60         return pos;
61     return fib(pos-2)+fib(pos-1);    //Recurividad
62 }
63
64
65 long fibo( unsigned long p,long *A){
66
67     printf("f(%d) ",p);
68     if (A[p]>-1)
69         return A[p];
70     A[p]=fibo(p-1,A)+fibo(p-2,A);    //Recurividad
71     return A[p];
72 }
73
74 long fibm( unsigned long p){
75     int i;
76     long Z [p+1];
77
78     Z[0]=0;
79     Z[1]=1;
80
81     for(i=2;i<=p;i++){
82         Z[i]=-1;
83     }
84     return fibo(p,Z);
85 }
86
87 void torres_hanoi(unsigned int disco, char disc_origen , char disc_anterior
88 , char disc_destino ){
89
90     if ( disco == 1 )
91         printf("Mover el Disco 1 de %c a %c \n", disc_origen , disc_destino
92 );
93     else {
94         torres_hanoi(disco-1,disc_origen,disc_destino,disc_anterior);
95         printf("Mover el Disco %d de %c a %c \n", disco , disc_origen ,
96             disc_destino);
97         torres_hanoi(disco-1,disc_anterior,disc_origen,disc_destino);
98     }
99 }

```

```

95     }
96 }
97
98
99 int main(int argc, char const *argv[])
100 {
101     char opcion, valid = 1, bin[6];
102     int numF;
103
104
105     do{
106         printf("Eliga una opcion (a - f):\n");
107         printf("a) Factorial\nb) Fibonacci\nc) Fibonacci con Arreglo\n");
108         printf("d) Torres de Hanoi\ne) Conversion Decimal a Binario\nf)
109             Conversion de Binario a Decimal\nOpcion: ");
110         scanf("%c",&opcion);
111
112         switch (opcion)
113         {
114             case 'a':
115                 system("cls");
116                 valid = 0;
117                 fflush(stdin);
118                 printf("Inserta el numero del que deseas obtener el
119                     factorial:\n");
120                 scanf("%d",&numF);
121                 printf("El factorial de %d es %d",numF,factorial(numF));
122                 break;
123             case 'b':
124                 system("cls");
125                 valid = 0;
126                 printf("Escribe el numero de la posicion en la serie de
127                     Fibonacci que quieras saber:\n");
128                 scanf("%d",&numF);
129                 printf("\nEl numero en la posicion %d es: %d\n",numF,fib(
130                     numF));
131                 break;
132             case 'c':
133                 system("cls");
134                 printf("Escribe el numero de la posicion en la serie de
135                     Fibonacci que quieras saber:\n");
136                 scanf("%d",&numF);
137                 printf("\nEl numero en la posicion %d calculado con arreglo
138                     es: %d\n",numF,fibm(numF));
139                 valid = 0;
140                 break;
141             case 'd':
142                 system("cls");
143                 printf("Indica el numero de discos de Hanoi (Del 1 al 5): \n
144                     ");
145                 scanf("%d",&numF);
146                 torres_hanoi(numF,'A','B','C');
147                 valid = 0;
148                 break;
149             case 'e':
150                 system("cls");
151                 valid = 0;
152                 printf("Escribe el numero que desee convertir a binario:\n")
153                     ;
154                 scanf("%d",&numF);
155                 printf("El numero %d en binario es: ",numF);

```

```

148         convertir_binario(numF);
149         break;
150     case 'f':
151         system("cls");
152         valid = 0;
153         printf("Ingrese la cantidad de bits:\n");
154         scanf("%d",&numF);
155         binario(numF);
156         break;
157     default:
158         printf("Opcion no valida, intente de nuevo.");
159         system("PAUSE");
160         system("cls");
161         break;
162     }
163
164 }while(valid == 1);
165
166
167 return 0;
168 }

```

3. Extensión de Funcionalidades

- Se podría ampliar una interfaz más didáctica para visualizar las traslaciones de Hanoi de forma explícita
- Podríamos implementar una calculadora a pasos de Binario a Decimal y de Decimal a Binario

4. Programas donde se emplea la recursividad

- En calculadoras donde se necesitan soluciones recursivas de un programa.
- Para un programa que busque hacer demostraciones matemáticas recurrentes
- Para simplificar cálculos de sistemas numéricos

5. Capturas de Pantalla

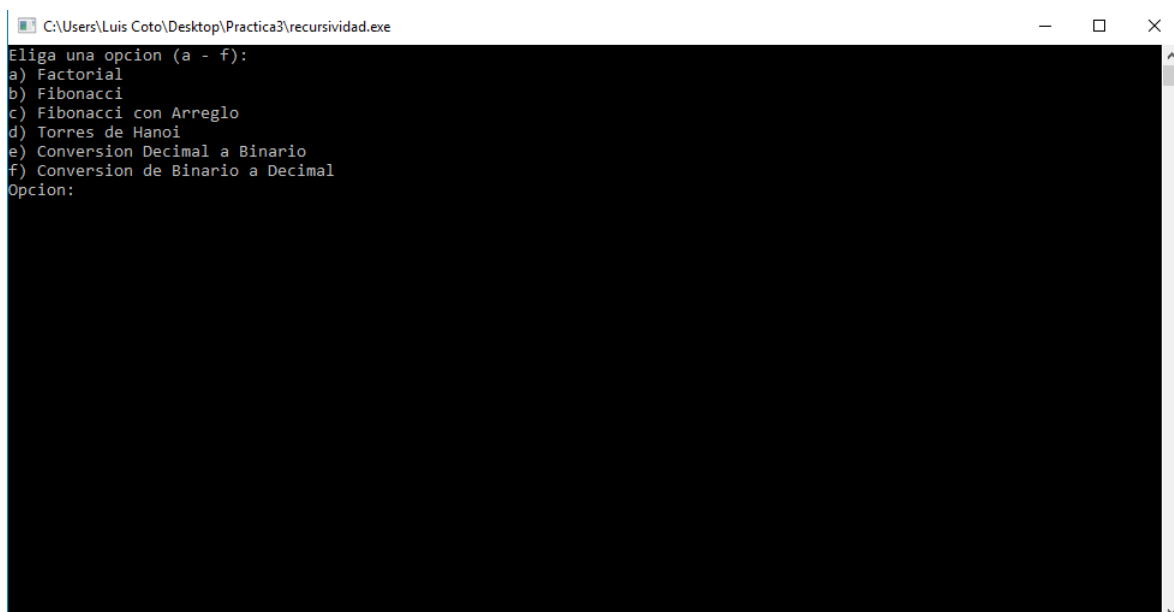


Figura 1: Pantalla inicial de ejecución

```
C:\Users\Luis Coto\Desktop\Practica3\recursividad.exe
Inserta el numero del que deseas obtener el factorial:
8
El factorial de 8 es 40320
-----
Process exited after 5.241 seconds with return value 0
Presione una tecla para continuar . . .
```

Figura 2: Opción a)Factorial

```
C:\Users\Luis Coto\Desktop\Practica3\recursividad.exe
Escribe el numero de la posicion en la serie de Fibonacci que quieras saber:
10
f(10) ,f(8) ,f(6) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(5) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(4) ,f(2)
,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(7) ,f(5) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1)
,f(2) ,f(0) ,f(1) ,f(6) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(5) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(4)
,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(9) ,f(7) ,f(5) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(4) ,f(2) ,f(0) ,f(1)
,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(6) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(5) ,f(3) ,f(1) ,f(2) ,f(0)
,f(1) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(8) ,f(6) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0)
,f(1) ,f(5) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(7) ,f(5) ,f(3) ,f(1)
,f(2) ,f(0) ,f(1) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(6) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2)
,f(0) ,f(1) ,f(5) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(4) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2) ,f(0) ,f(1) ,f(3) ,f(1) ,f(2)
El numero en la posicion 10 es: 55
-----
Process exited after 4.641 seconds with return value 0
Presione una tecla para continuar . . .
```

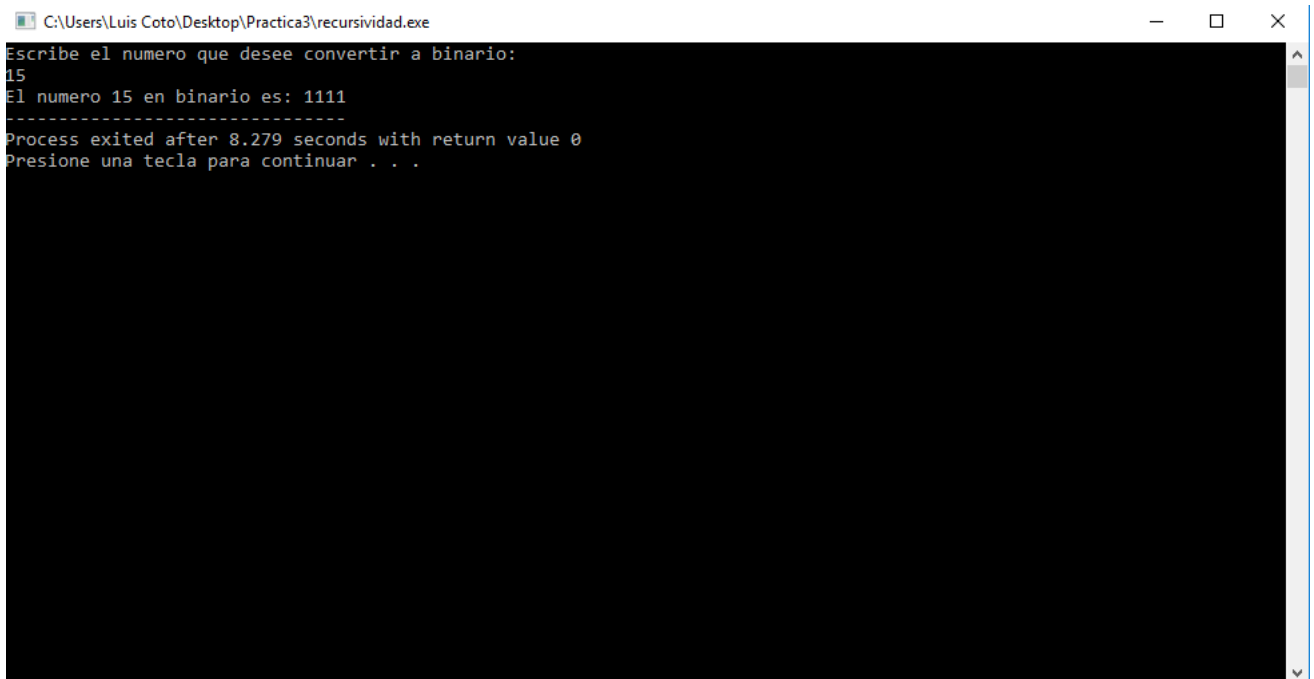
Figura 3: Opción b)Fibonacci

```
C:\Users\Luis Coto\Desktop\Practica3\recursividad.exe
Escribe el numero de la posicion en la serie de Fibonacci que quieras saber:
10
f(10) ,f(9) ,f(8) ,f(7) ,f(6) ,f(5) ,f(4) ,f(3) ,f(2) ,f(1) ,f(0) ,f(1) ,f(2) ,f(3) ,f(4) ,f(5) ,f(6) ,f(7) ,f(8) ,
El numero en la posicion 10 calculado con arreglo es: 55
-----
Process exited after 11.05 seconds with return value 0
Presione una tecla para continuar . . .
```

Figura 4: Opción c)Fibonacci con Arreglo

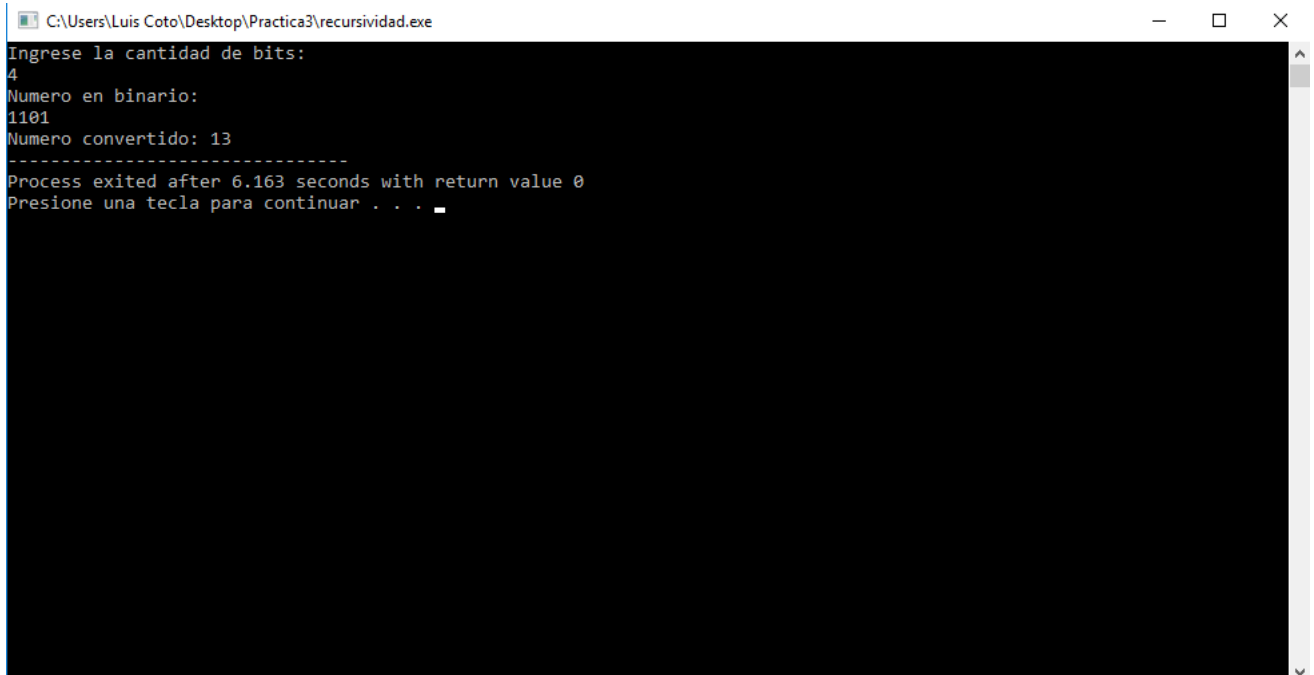
```
C:\Users\Luis Coto\Desktop\Practica3\recursividad.exe
Indica el numero de discos de Hanoi (Del 1 al 5):
4
Mover el Disco 1 de A a B
Mover el Disco 2 de A a C
Mover el Disco 1 de B a C
Mover el Disco 3 de A a B
Mover el Disco 1 de C a A
Mover el Disco 2 de C a B
Mover el Disco 1 de A a B
Mover el Disco 4 de A a C
Mover el Disco 1 de B a C
Mover el Disco 2 de B a A
Mover el Disco 1 de C a A
Mover el Disco 3 de B a C
Mover el Disco 1 de A a B
Mover el Disco 2 de A a C
Mover el Disco 1 de B a C
-----
Process exited after 5.026 seconds with return value 0
Presione una tecla para continuar . . .
```

Figura 5: Opción d)Hanoi



```
C:\Users\Luis Coto\Desktop\Practica3\recursividad.exe
Escribe el numero que desee convertir a binario:
15
El numero 15 en binario es: 1111
-----
Process exited after 8.279 seconds with return value 0
Presione una tecla para continuar . . .
```

Figura 6: Opción e)Decimal a Binario



```
C:\Users\Luis Coto\Desktop\Practica3\recursividad.exe
Ingrese la cantidad de bits:
4
Numero en binario:
1101
Numero convertido: 13
-----
Process exited after 6.163 seconds with return value 0
Presione una tecla para continuar . . .
```

Figura 7: Opción f)Binario a Decimal