Funciones recursivas.

- Son funciones que llaman a ellas mismas

Ejemplo de la función que calcula factorial de un número entero no negativo.

n(n-1)(n-2)…1

```
factorial = 1;

for (int contador = numero; contador >= 1; contador --)
    factorial *= contador;
```

Definición recursiva de la función factorial:

$$n! = \begin{cases} 1 & si\ n = 0,1 \\ n(n-1)! & \end{cases}$$

```
double factRec(int n)
{
double i;

if (n<0){
   printf(" n=%d inadecuado.\n", n);
   return 0.0;
         }
if (n<2) return 1.0;
else return(n*factRec(n-1) );
}
```

Nota: para calcular factorial pueden usar tipo de variables "unsigned long" para los resultados dado que esta se almacena numeros de 0 a 429 4967 295.

Ej: cambiar el código para que en cada llamado de la función recursiva nos imprima cada vez "n" al llamar función a ella misma..

Ejercicio:

Averiguar en Internet sobre serie de Fibonacci y hacer un programa que al introducir un numero entero n le genera enésimo numero de Fibonacci.

Comparación de uso de recursividad y iteración:

	Recursividad	Iteración
Base	Estructuras de control: selección	Estructuras de control: repetición
Repetición	si	si
Prueba de terminación	si (base)	Si (condición)
Infinitas	Pueden ser	Pueden ser
Desventajas	sobrecarga de llamadas a función	

- Evite utilizar la recursividad en situaciones de rendimiento
- Cualquier problema que se resuelve de manera recursiva se puede resolver también mediante iteraciones.

Funciones con listas de parámetros vacías.

void imprime();

Ejemplo: vacio.cpp

Averiguar sobre torres de Hanoi y hacer un programa con función recursiva hanoi(n), donde n es el número de discos que calcula la cantidad mínima de movimientos para pasar los discos de una torre a otra.