Recursión

Programación I

Definición

Es una técnica de programación en la cual una función se llama a si misma.

Ejemplo: Cómo se calcula el factorial? El factorial del entero no negativo n es el producto de todos los enteros desde el n hasta 1.

• Formalmente:

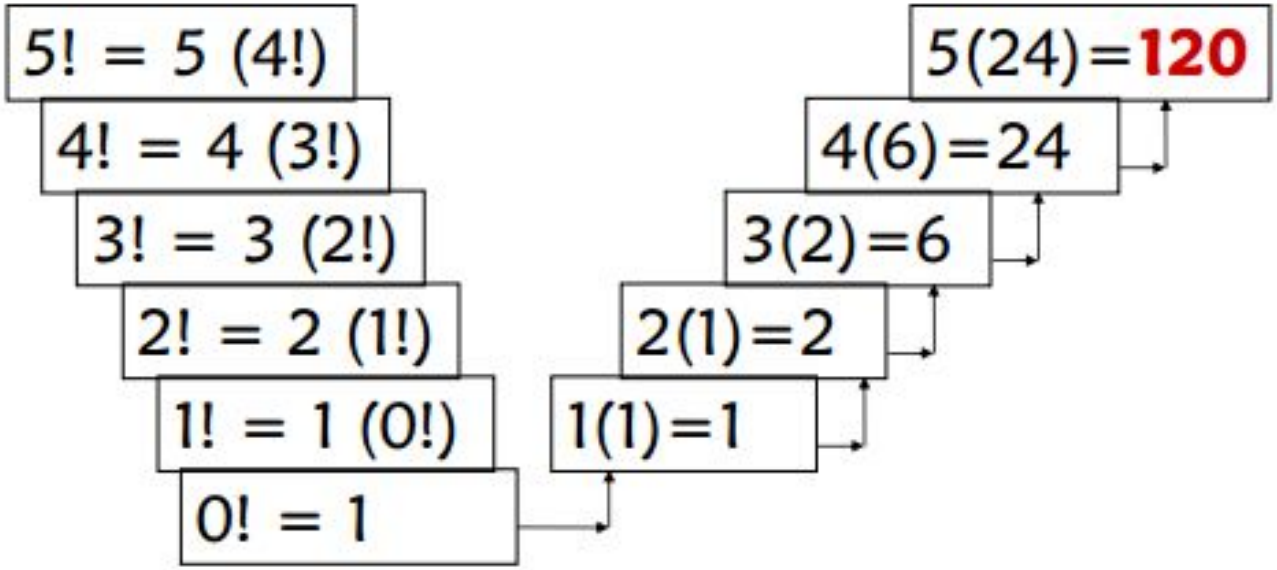
$$n! = \prod_{k=1}^{k=n} k$$

• De este producto es fácil ver que el factorial de n es también n! = n * (n - 1)!

• Caso Base: n = 1 → factorial(1) = 1;

Caso Recursivo: n > 1 → factorial(n) = n * factorial(n-1);

Ejecución



Ejecución

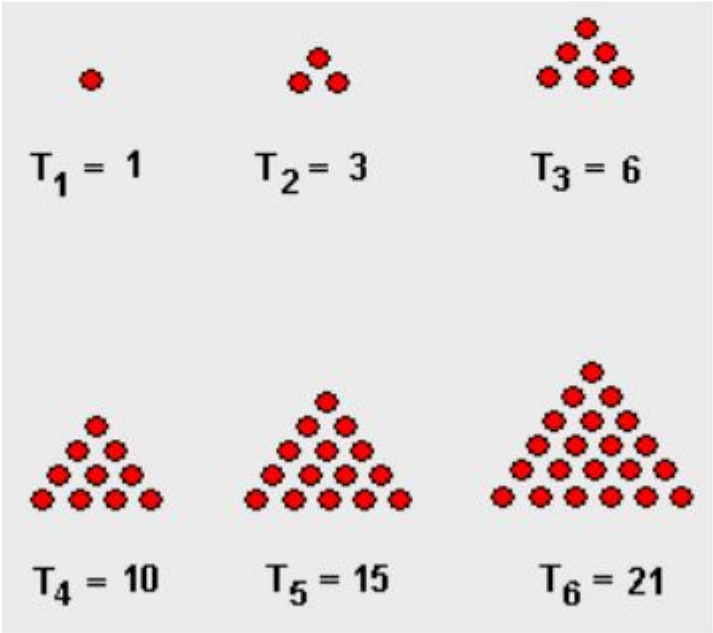
X = 3 //Queremos 3!, por lo tanto X inicial es 3
X >= 2 -> return 3*factorial(2);
    X = 2 //Ahora estamos solicitando el factorial de 2
    X >= 2 -> return 2*factorial(1);
        X = 1 // Ahora estamos solicitando el factorial de 1
        X < 2 -> return 1;
        [En este punto tenemos el factorial de 1 por lo que volvemos marcha atrás resolviendo todos los resultados]
    return 2 [es decir: return 2*1 = return 2*factorial(1)]
return 6 [es decir: return 3*2 = return 3*factorial(2)*factorial(1)] // El resultado devuelto es 6

Otro ejemplo

Dícese que los Pitagóricos sentían una conexión mística con la serie de números 1, 3, 6, 10, 15, 21, ...

Puedes encontrar el próximo miembro de esta serie?

El enésimo termino se consigue añadiendo n al termino previo



Los números en esta serie se llaman triangulares porque pueden ser visualizados de forma triangular

Cómo sería el código?

Código

- Si introducimos 4,
- 4 > 1 por lo tanto ejecuta 4 + triangulo(3)
- Entra la función con argumento 3
- 3 > 1 por lo tanto 3 + triangulo(2)
- Entra la función con argumento 2
- 2 > 1 por lo tanto 2 + triangulo(1)
- Entra la función con argumento 1
- 1 = 1 por lo tanto retorna 1
- Vuelve a la función con argumento 2
- triangulo(1) = 1, retorna 2 + 1 = 3
- Vuelve a la función con argumento 3
- triangulo(2) = 3, retorna 3 + 3 = 6
- Vuelve a la función original, con argumento 4
- triangulo(3) = 6, retorna 4 + 6 = 10

Caso Base?
Caso Recursivo?

Características de los métodos recursivos

Aunque triangulo() sea tan corta, posee las características comunes a todas las funciones recursivas:

– Se llama a ella misma
– Cuando se llama a ella misma, lo hace para resolver un problema mas pequeño
– Contiene una versión del problema tan simple que puede resolverla y retornar sin llamarse a ella misma

– Definen al menos un caso base y al menos un caso recursivo.

Características de los métodos recursivos

Ventajas
- Soluciones simples y claras
- Soluciones elegantes
- Soluciones a problemas Complejos

Desventajas
- INEFICIENCIA
- Sobrecarga
- Costo mayor en tiempo y memoria

La Ineficiencia de la recursión

El control se transfiere desde donde la función fue llamada al inicio de esta

Los argumentos y la dirección donde la función debe retornar se ponen en una pila

Con poca data quizá no haya problemas, pero una gran cantidad de data puede causar un *stack overflow*

La recursión se usa comúnmente porque simplifica un problema conceptualmente, no porque es inherentemente mas eficiente