UNIVERSIDAD NACIONAL DE HURLINGHAM

Estructuras de Datos

Profesor

Sergio Gonzalez

UNIVERSIDAD NACIONAL DE HURLINGHAM

Unidad 2: Recursividad, Arreglos uni y multidimensionales

Profesor

Sergio Gonzalez

Programación modular

- Descomposición en módulos independientes
  - Subprogramas o subalgoritmos
  - Independencia del programa principal
  - Diseño descendente
  - Reutilización de código

Programación modular

- Subprogramas se 'llaman' desde el programa principal
- Abstracción procedimental:
  - Nombre
  - Parámetros de entrada / salida
- Tipos de subprogramas:
  - Funciones
  - Procedimientos / Subrutinas

Programación modular

- Funciones
  - Tareas especificas
  - Lista de valores de entrada (argumentos)
  - Único valor de salida
- Procedimientos / Subrutinas
  - Similar a funciones
  - Pueden devolver ninguno o varios valores

Programación modular

- Recordatorio:
  - Ámbito de las variables (scope)
  - Paso de parametos
    - Por valor
    - Por referencia

Recursividad

- Alternativa a la repetición (soluciones iterativas)
- Problemas con ‘estructura de solución recursiva’

Estructura de solución recursiva

- Tenemos un problema A
- Se divide en dos partes B y C
- Si una de esas partes (por ejemplo B) es idéntica a A
- Entonces el problema es recursivo, porque la resolución de B se puede dividir igual que la de A

Ejemplo

- Calculo del factorial
  - N! = N (N - 1)!
  - (N - 1)! = (N – 1) (N - 2)!

Estructura de solución recursiva

- En programación, tenemos un procedimiento que se llama a si mismo
- Caso general: La autollamada resuelve una o mas versiones mas chicas del problema y algunas cosas mas.
- Caso base: Para que sea finito, debe haber una condición de corte (sin autollamada)

Método de las 3 preguntas

- Caso base: Existe una salida no recursiva o caso base? Es correcta la solución?
- Mas chico: Cada autollamada es un problema mas chico del original?
- Caso general: Es correcta la solución en los casos no base?

Escritura de programas recursivos

- Obtener definición exacta del problema
- Determinar el tamaño del problema general
- Resolver el/los casos triviales (no recursivos)
- Resolver el caso general en términos de uno mas chico (llamada recursiva)

Ejemplos

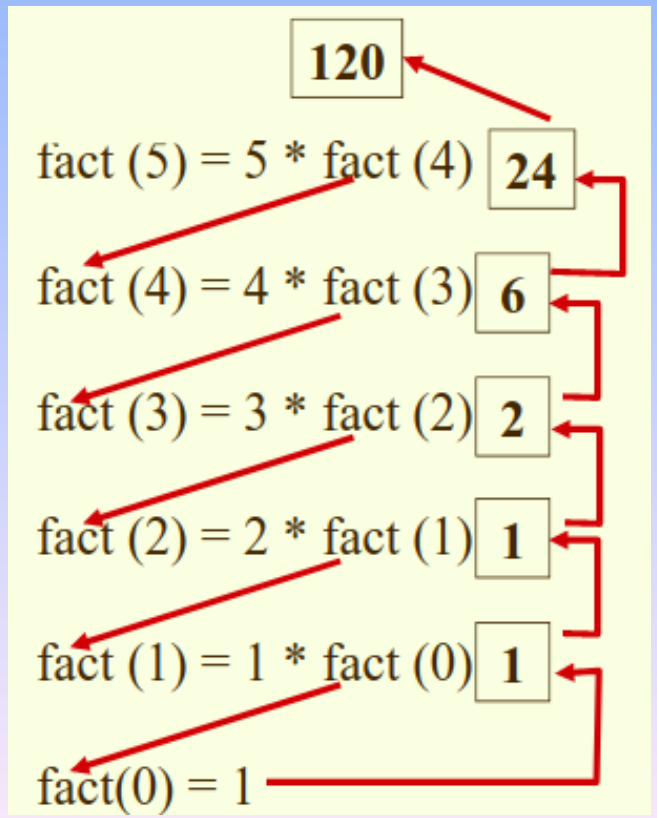
- Ejemplo 1: Calculo del factorial
- Solución en pseudocódigo

```
función Factorial (n)
inicio
    si n = 0
        entonces Factorial ←1
        sino Factorial ← n * Factorial (n-1)
    fin_si
fin
```

Ejemplos

- Ejemplo 1: Calculo del factorial de 3, descripción de llamadas apiladas:
  - factorial(3) -> 3 * factorial(2)
  - factorial(2) -> 2 * factorial(1)
  - factorial(1) -> 1 * factorial(0)
  - factorial(0) <- 1

- Ejemplo 1: Calculo del factorial de 5, descripción de llamadas apiladas:



Ejemplos

- Ejemplo 2: Serie de Fibonacci
  - Fibonacci(N) = Fibonacci(N - 2) + Fibonacci(N - 1)
    - Fibonacci(N - 2) = Fibonacci(N - 4) + Fibonacci(N - 3)
    - Fibonacci(N - 1) = Fibonacci(N - 3) + Fibonacci(N - 2)

Ejemplos

- Ejemplo 2: Serie de Fibonacci
- Solución en pseudocódigo

```
función FIBONACCI(n)
inicio
    si (n=1) o (n=2)
        entonces
            FIBONACCI ← 1
        sino
            FIBONACCI ← FIBONACCI (n-2) + FIBONACCI (n-1)
    fin_si
fin_función
```

- Ejemplo 2: Calculo del 4 numero de la serie de de Fibonacci, descripción de llamadas apiladas: