UNER Facultad de Ciencias de la Administración

Algoritmos y Programación

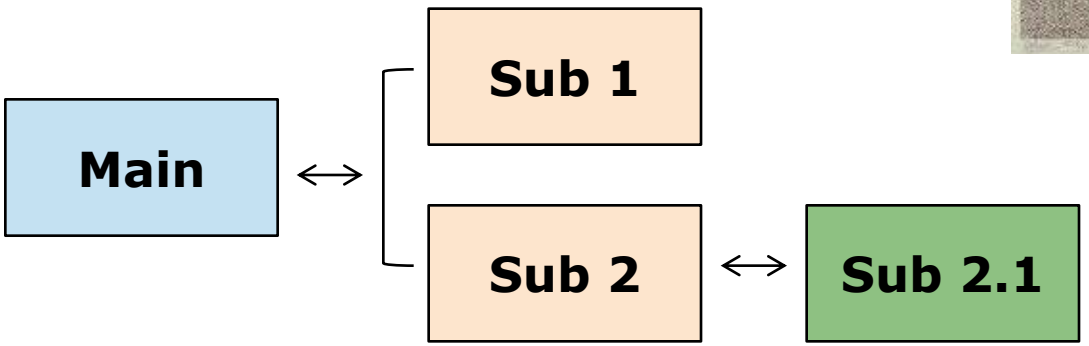
Recursividad Parte 1

Recursividad

- Desde el punto de vista algorítmico:
  es un método alternativo a los esquemas repetitivos.
- El concepto de recursividad está ligado, en los lenguajes de programación, al concepto de procedimiento o función. Un procedimiento o función es recursivo cuando durante una invocación a él puede ser invocado a su vez él mismo.
- Una acción es recursiva cuando se define en función de sí misma.

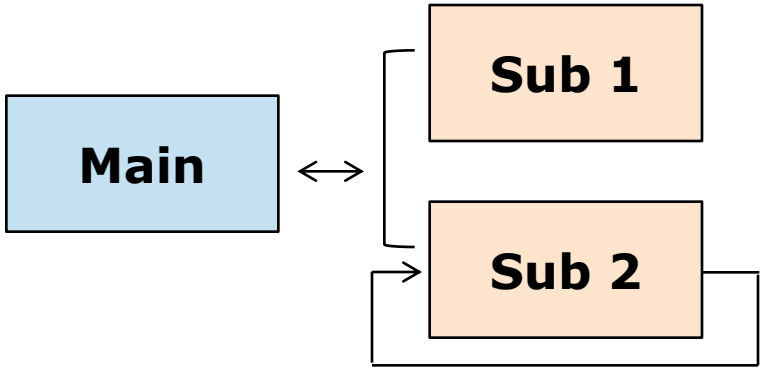
Recursividad

Sabemos que: Un subprograma puede llamar a otro subprograma:



Ahora:

Un subprograma puede llamarse a sí mismo:



Universidad Nacional de Entre Ríos

Recursividad

Ejemplo: Calcular el resultado de una operación de potencia mediante multiplicaciones sucesivas.

```
Program Potencia
Var
   R, Bas, Exp : Integer;
Procedure Calculo
Inicio
 Si  Exp > 0
       R := R * Bas
       Exp := Exp – 1
       Calculo
 FinSi
FIN
Fin-Procedimiento CALCULO
INICIO
   R := 1
   Ingresar Bas, Exp
   Calculo
   Mostrar R
FIN
```

MEMORIA RAM	
Dirección	DATOS
31	R
30	Bas
29	Exp
28	R := 1
27	Ingresar Bas, Exp
26	Calculo
25	Mostrar R

Registro De Activación

Recursividad

```
Program Potencia
Var
   R, Bas, Exp : Integer
Procedure Calculo
Inicio
  Si  Exp > 0
      R := R * Bas
      Exp := Exp – 1
      Calculo
  FinSi
FIN
Fin-Procedimiento Calculo
INICIO
   R := 1
   Ingresar Bas, Exp
   Calculo
   Mostrar R
FIN
```

Variables

MAIN

Código

H45	R	1	2	4	8
H40	Bas	2	2	2	2
H35	Exp	3	2	1	0
H30	R := 1				
H25	Ingresar Bas, Exp				
H20	Calculo				
H15	Mostrar R				
H10	FIN				

8

Recursividad

Condición de Final (Stop):

Al igual que en los esquemas repetitivos, debe preverse una condición de fin (evitar bucles infinitos)

En el ejemplo anterior, la condición es:

Exp = 0

Algoritmo Divergente:

- Cuando un subprograma no contempla la condición de Final.
- Genera error en ejecución (“overflow” o desbordamiento de pila)

Recursividad

Ventajas:

– Soluciona problemas recurrentes
– Permite generar programas menos extensos.
– Facilita la comprensión al modularizar el problema.

Desventajas:

– Ineficiencia en el uso de recursos (tiempo y memoria).
- Creación de muchas variables.
– Puede necesitar mucha memoria.
– Puede generar problemas de “overflow”.

Recursividad

- Calcular la sumatoria de los números naturales desde el 1 hasta el número 5. Imprimir el resultado.

Con esquema repetitivo

```
Program SUMA
VAR
      Res : Integer;
      i : Integer;
BEGIN
     Res := 0
     Para i = 1, 5, 1
          Res := Res + i
     FinPara
     Imprimir Res
FIN
```

Con Recursividad

```
Program SUMA
VAR
      Res, Con : Integer;
Procedure UNO
     Si Con < 5
          Con := Con + 1
          Res:= Res + Con
          UNO
     FinSi
Fin-Procedimiento UNO
BEGIN
     Res, Con := 0
     UNO
     Imprimir Res
FIN
```

Recursividad

- Calcular la sumatoria de los números naturales desde el 1 hasta un número que ingresa el operador por teclado. Imprimir el resultado.

```
Program SUMA
VAR
      Res, Con, Nro : Integer;
Procedure UNO
      Si Con < Nro
          Con := Con + 1
          Res := Res + Con
          UNO
      FinSi
Fin-Procedimiento UNO
BEGIN
      Ingresar Nro
      Res, Con := 0
      UNO
      Imprimir Res
FIN
```

Universidad Nacional de Entre Ríos

UNER Facultad de Ciencias de la Administración

Algoritmos y Programación

FIN Recursividad Parte 1