Clase 2

Recursión

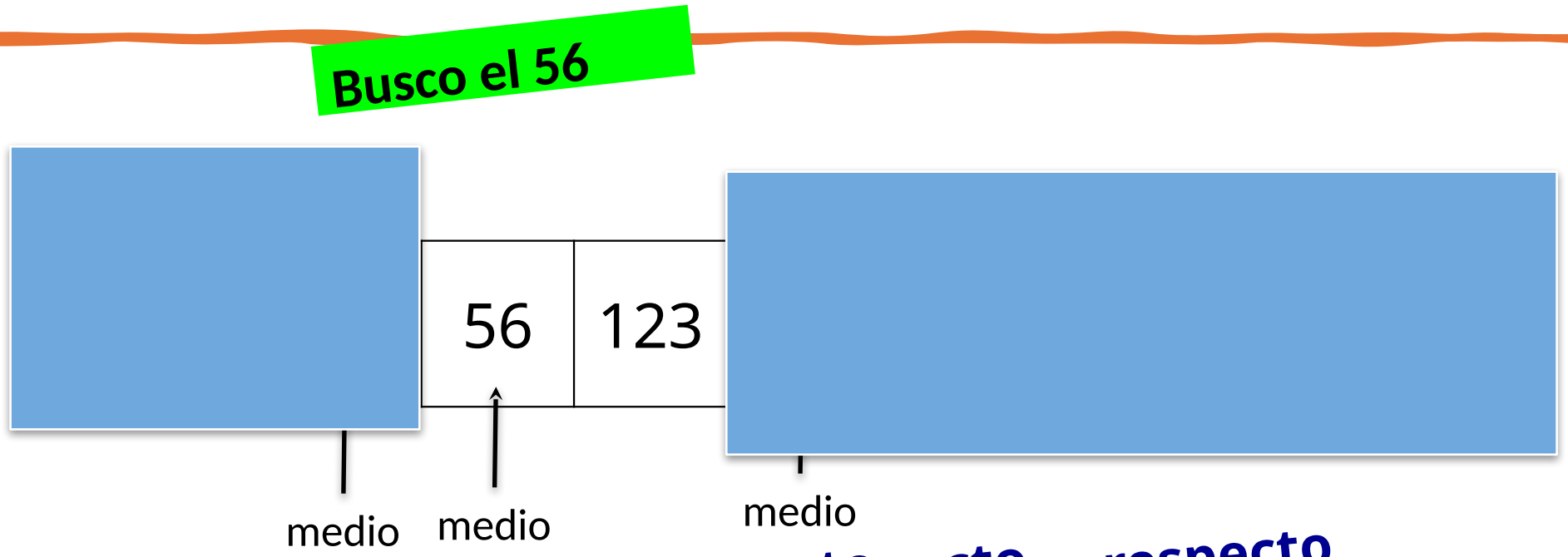
Temas de la clase

RECURSIÓN. DEFINICIÓN Y CARACTERÍSTICAS

EJEMPLOS DE RECURSIÓN

MÉTODO DE BÚSQUEDA DICOTÓMICA EN VECTORES. UNA APLICACIÓN

Motivación búsqueda dicotómica



Motivación búsqueda dicotómica

Observemos que :

1. La primera vez se trabaja con el vector completo para determinar el punto medio
2. La siguiente vez, el vector se reduce a la mitad
3. La siguiente vez, el vector se reduce a la mitad de la mitad

Motivación búsqueda dicotómica

```
Buscar (vector, datoABuscar)
 Si el vector "no tiene elementos"  entonces
  no lo encontré y termino la búsqueda
 sino
  Determinar el punto medio del vector
  Comparar datoABuscar con el contenido del punto medio
  Si coincide entonces "Lo encontré"
  sino
   Si datoABuscar < contenido del punto medio entonces
     Buscar (en la 1era mitad del vector, datoABuscar)
    sino
     Buscar (en la 2da mitad del vector, datoABuscar)
```

Motivación búsqueda dicotómica

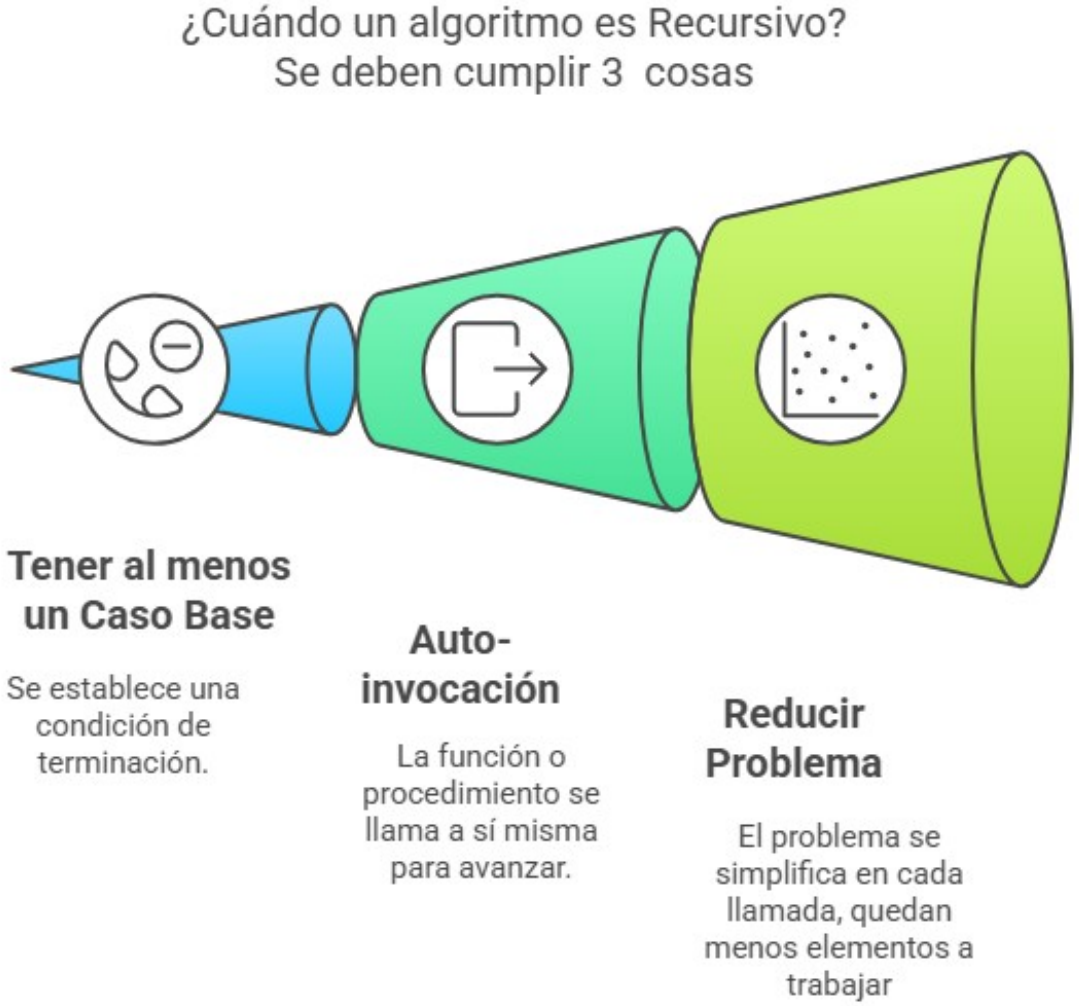
Observaciones importantes de esta solución :

1. El módulo realiza invocaciones a si mismo. En cada llamada, el tamaño del vector se reduce a la mitad.

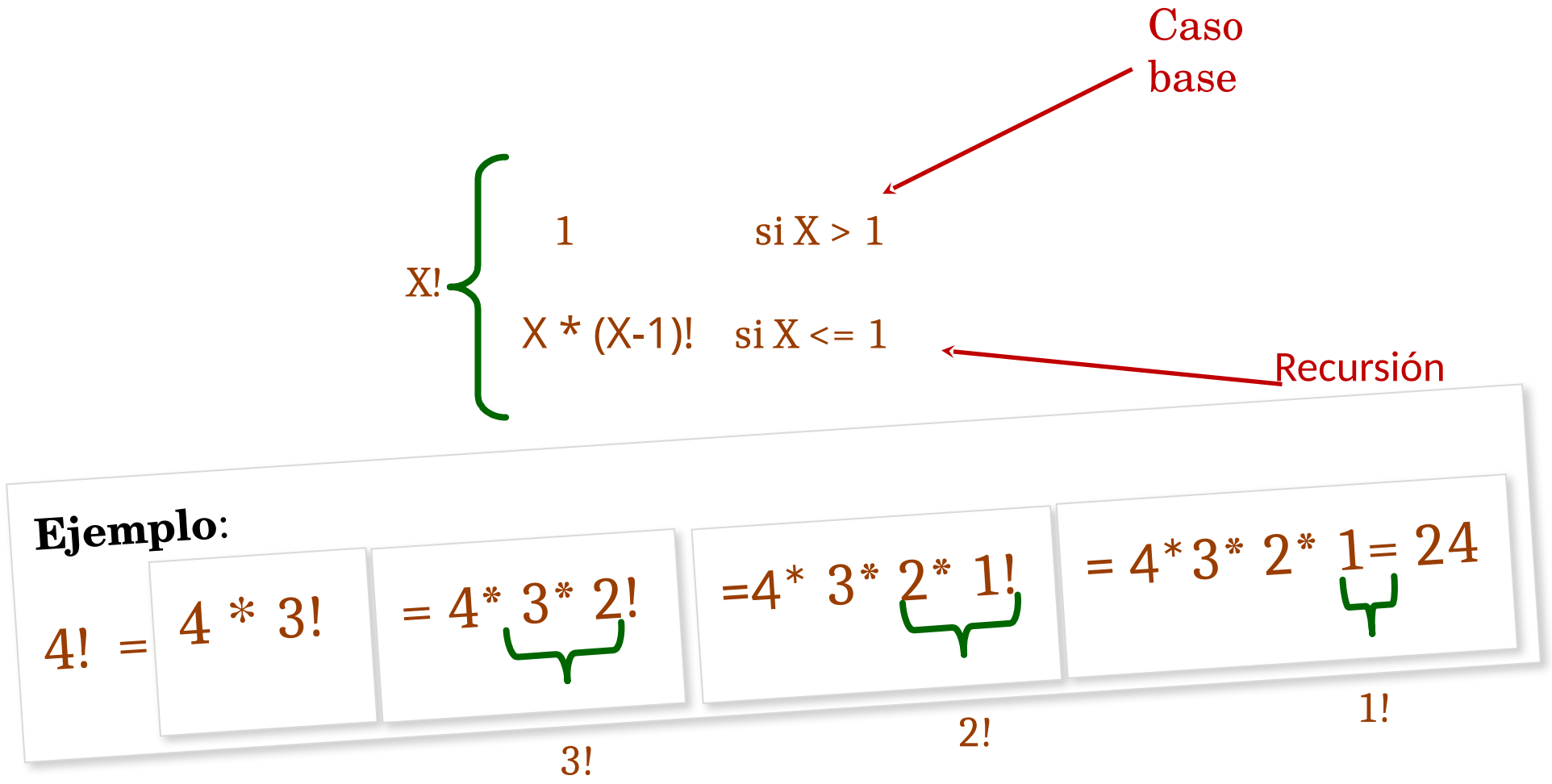
2. Existen 2 casos distintos que se resuelven de manera particular y directa:

a) Cuando el vector "no contiene elementos"
b) Cuando encuentro el datoABuscar

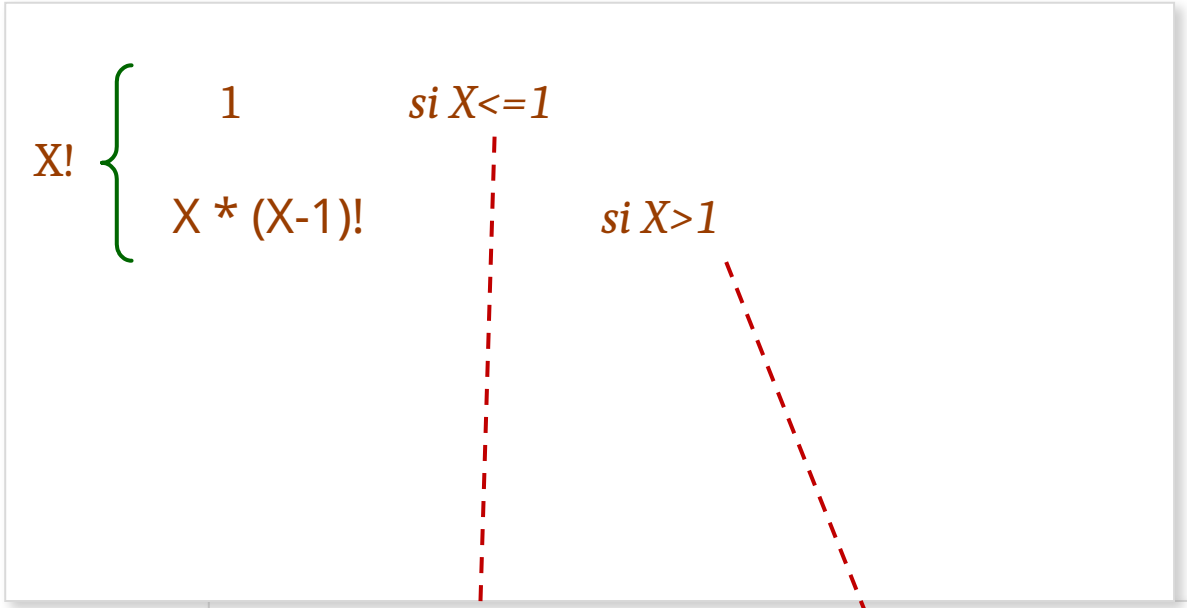
Algoritmo recursivo



Ejemplo 1 – Factorial de un número



Ejemplo 1 – Factorial de un número




```
Function factorial(x: integer): real;
begin
  if (x <= 1) then
    factorial:= 1
  else
    factorial := x * factorial(x-1)
end;
```

Ejemplo 2 – Suma N números

Escribe un programa en Pascal que calcule la suma de los primeros N números utilizando una **función recursiva**

¿Cuál es el caso base?

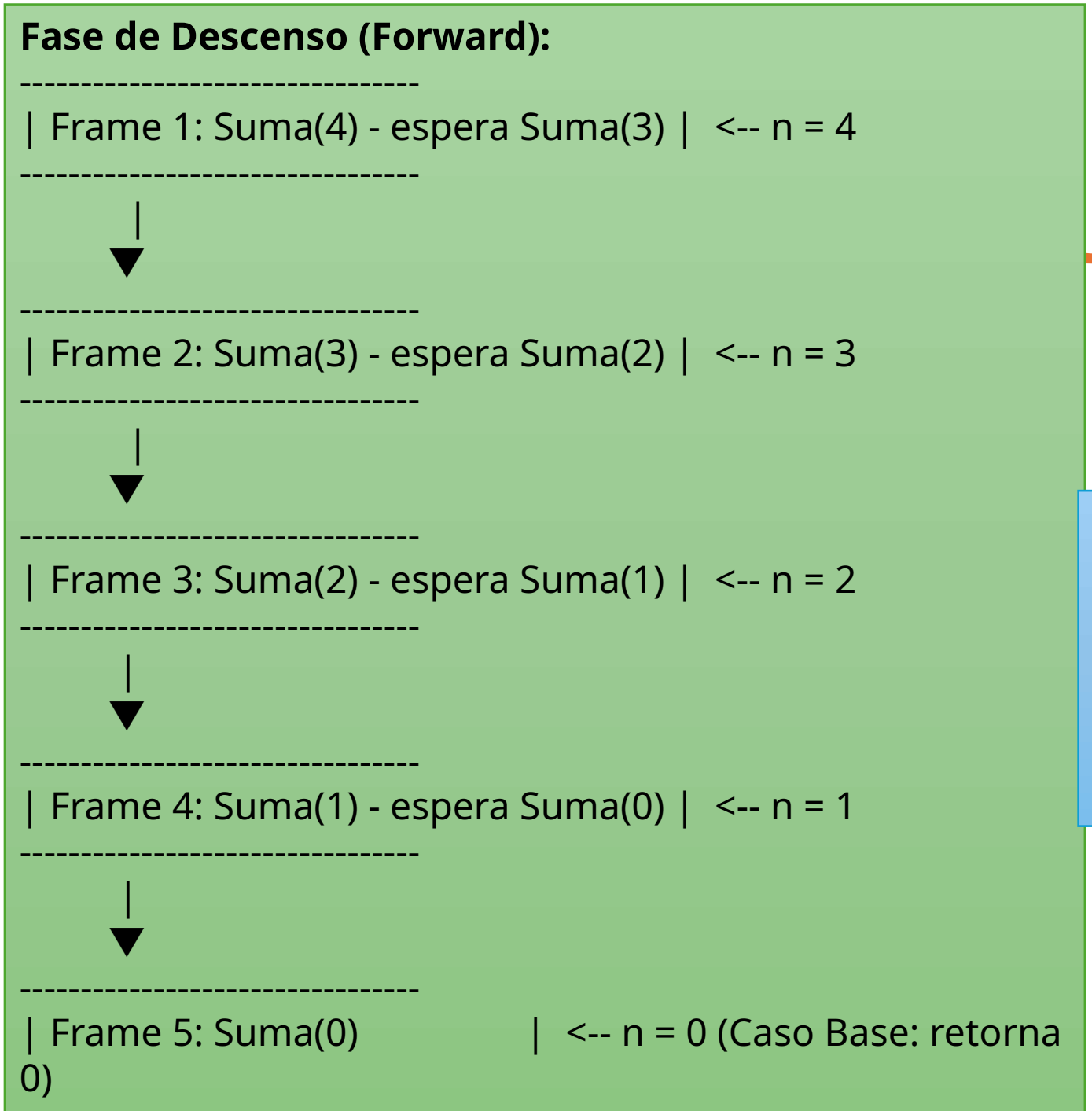
¿Cómo realiza la suma?

Ejemplo 2 – Suma N números

```
function Suma(n: Integer): Integer;
begin
  if n = 0 then
    Suma := 0          { Caso base }
  else
    Suma := n + Suma(n - 1);  { Caso recursivo }
end;
```

Ejemplo 2 – Suma N números

Fase de Descenso (Forward):



Fase de Retroceso (Backtracing - backward):

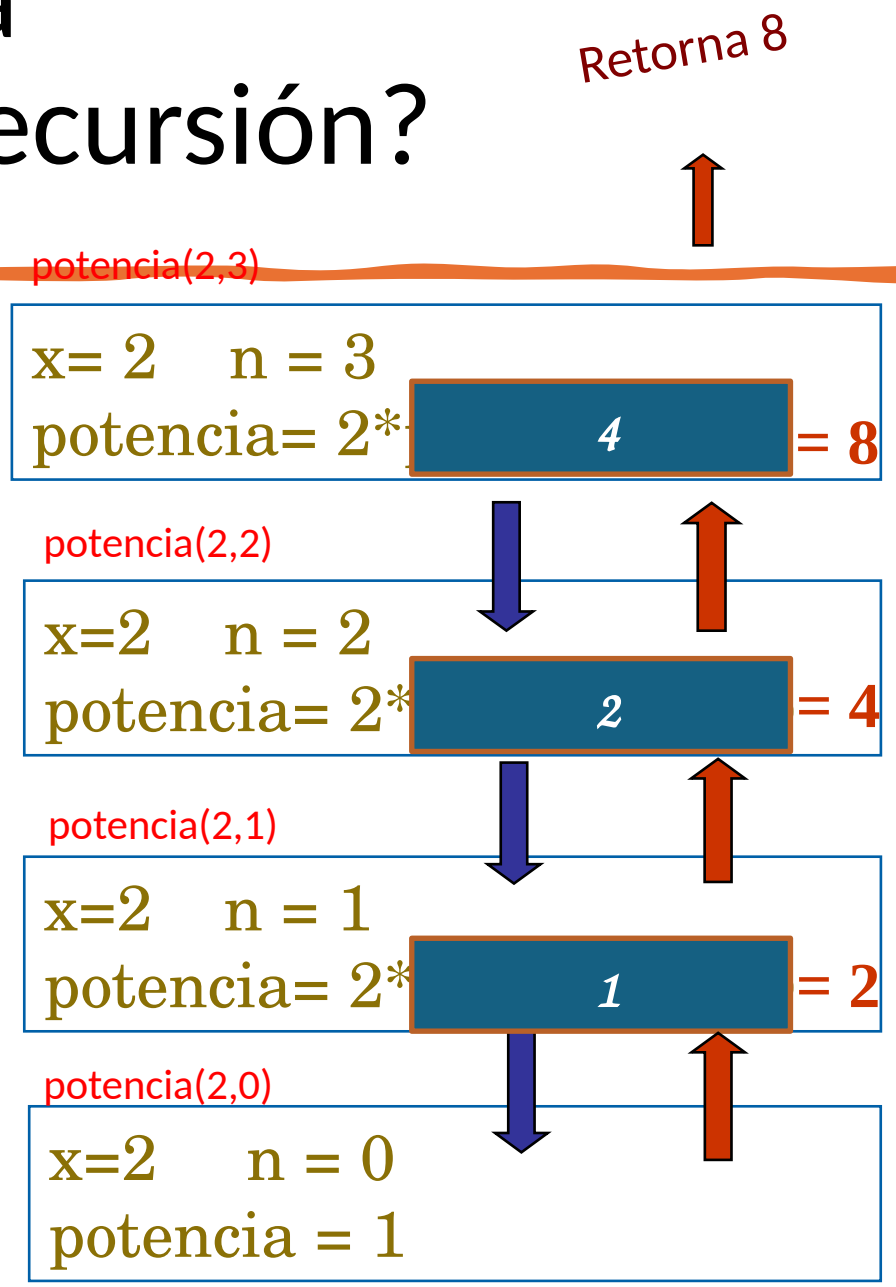
Frame 5 retorna 0 y se libera
Frame 4: calcula 1 + 0 = 1, retorna 1 y se libera
Frame 3: calcula 2 + 1 = 3, retorna 3 y se libera
Frame 2: calcula 3 + 3 = 6, retorna 6 y se libera
Frame 1: calcula 4 + 6 = 10, retorna 10 y se libera

Ejemplo 3 – Potencia
¿Cómo funciona la recursión?

```
Program ejemplo

Function potencia (x,n:integer):
real;
begin
  if (n = 0) then
    potencia:= 1;
  else
    potencia := x * potencia(x,n-1)
end;

Begin
  read (x,n);
  write(potencia(x,n))
End.
```



Ejemplo 4 – Número de Fibonacci

Escribe un programa en Pascal que genere la secuencia de **Fibonacci** hasta el enésimo término utilizando recursión.

1. Buscar en la Web que es el número de fibonacci

2. ¿Cómo se genera el Número de Fibonacci?

3. ¿Cúal es el caso base?

4. ¿Qué tipo de recursión es?

Ejemplo 4 – Número de Fibonacci

```
program FibonacciRecursivo;

function CalcularFibonacci(n: integer): integer;
begin
  if (n = 0) or (n = 1) then
    CalcularFibonacci := n
  else
    CalcularFibonacci := CalcularFibonacci(n - 1) + CalcularFibonacci(n - 2);
end;
var
  numero, i: integer;
begin
  write('Ingrese un número para generar la secuencia de Fibonacci hasta ese término: ');
  readln(numero);

  writeln('Secuencia de Fibonacci hasta el término ', numero, ':');
  for i := 0 to numero do
    write(CalcularFibonacci(i), ' ');
  writeln;
end.
```

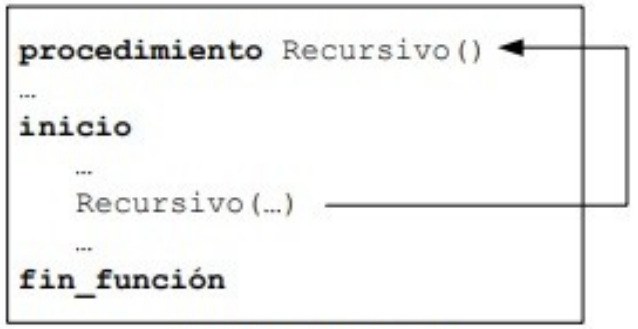
¿Qué ocurre en esta función con los llamados recursivos?

¿Que hay de diferente?

Tipos de recursión

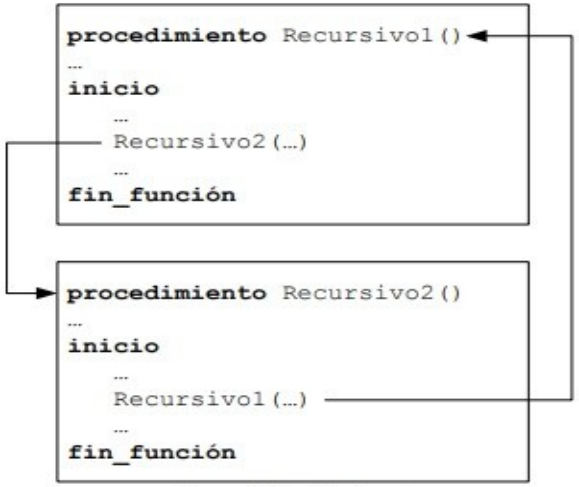
Según el subprogram a al que se llama

Recursividad directa. La función incluye una referencia explícita a sí misma.



Recursividad directa

Recursividad indirecta. El módulo llama a otros módulos de forma anidada y en la última llamada se llama al primero.



Recursividad indirecta

Tipos de recursión

Según el modo en que se hace la llamada recursiva

```
procedimiento f(valor entero: n)
…
inicio
   si n>0 entonces
      f(n-1)
   fin_si
   instrucción A
   instrucción B
fin_procedimiento
```

Recursividad de cabeza

```
procedimiento f(valor entero: n)
…
inicio
   instrucción A
   instrucción B
   si n>0 entonces
      f(n-1)
   fin_si
fin_procedimiento
```

Recursividad de cola

```
procedimiento f(valor entero: n)
…
inicio
   instrucción A
   si n>0 entonces
      f(n-1)
   fin_si
   instrucción B
fin_procedimiento
```

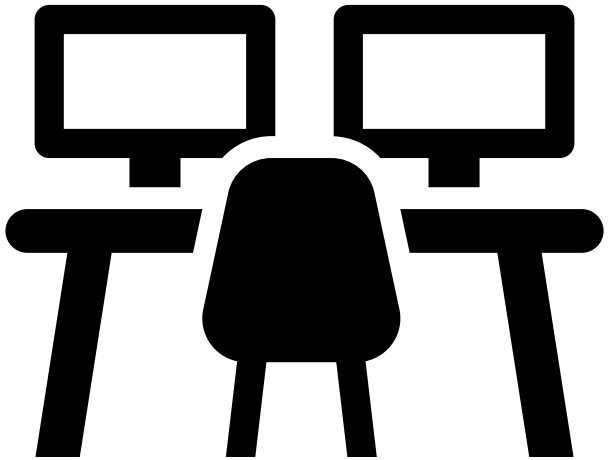
Recursividad de intermedia

```
procedimiento f(valor entero: n)
…
inicio
   ...
   si n>0 entonces
      f(n-1)
   fin_si
   si n<5 entonces
      f(n-2)
   fin_si
   ...
fin_procedimiento
```

Recursividad múltiple

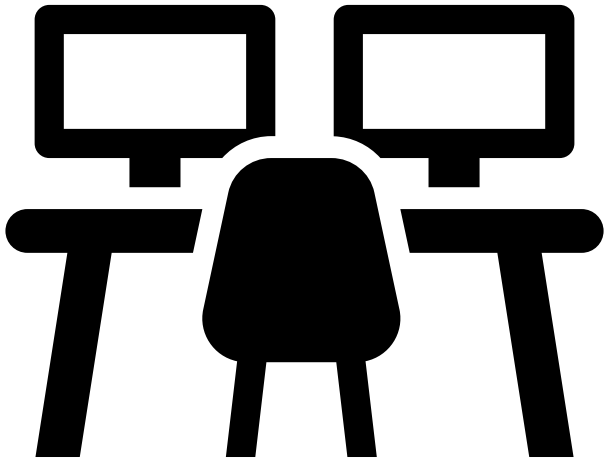
```
entero función f(valor entero: n)
…
inicio
   ...
   si n>0 entonces
      devolver(f(n-1)+f(n-2))
   fin_si
fin_función
```

Recursividad anidada



Actividad 1

- **Descargue de Asignaturas: programaCalculoDeFactorial**
  a) Implementar el módulo **factorial** como parte del programa **CalculoDeFactorial**
  b) Completar el programa **CalculoDeFactorial** para que lea un valor X, invoque a la función **factorial** para calcular X! y muestre el resultado.
  c) Compilar y ejecutar



Actividad 2

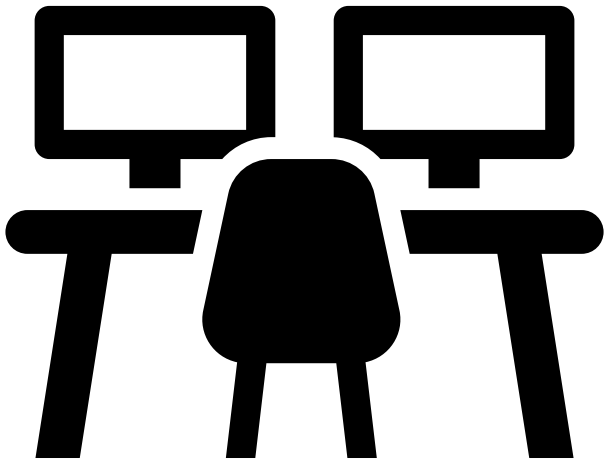
Descargar de Asignaturas:
`programaCalculoDePotencia`

a) Implementar en el programa CalculoDePotencia, la función **`potencia1`**.

```
Function potencia1 (x,n: integer): real;
begin
  potencia1 := x * potencia1(x,n-1)
end;
```

b) Invocar a la función **`potencia1`** para calcular 5^3 .

c) Compilar y ejecutar. ¿Qué ocurre? ¿Por qué?



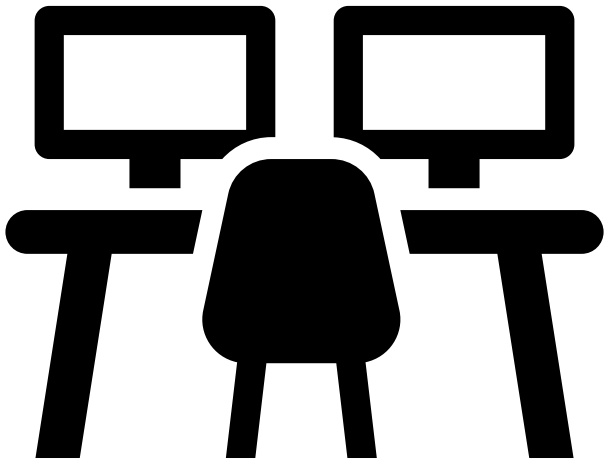
Actividad 3

a) Implementar en el programa CalculoDePotencia, la función **potencia2**.

```
Function potencia2 (x,n: integer):
real;
begin
  if (n = 0) then
    potencia2 := 1;
  else
    potencia2 := x * potencia2(x,n)
end;
```

b) Invocar a la función **potencia2** para calcular 5^3 .

c) Compilar y ejecutar. ¿Qué ocurre? ¿Por qué?



Actividad 4

- **Descargar de Asignaturas: `programaRecursion`**

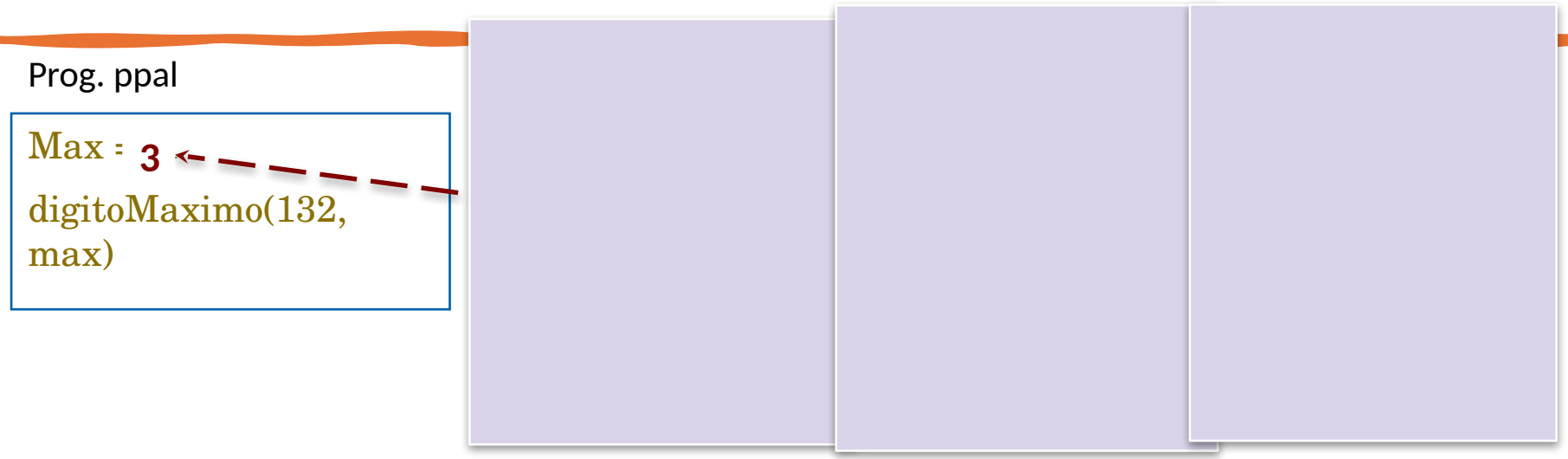
Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

a) Compilar y ejecutar.

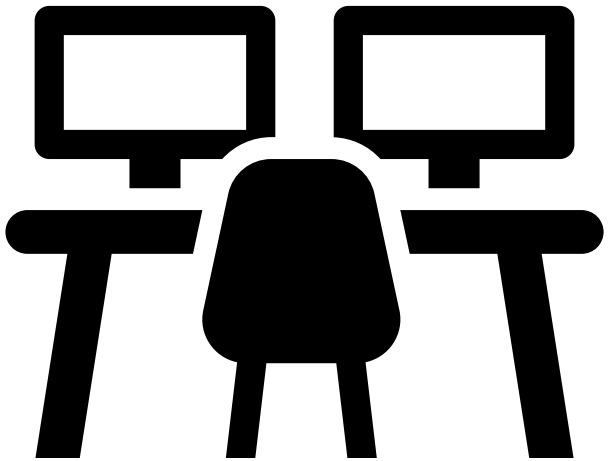
b) Responder:

- ¿Cuál es el caso base en el procedimiento `digitoMaximo`?
- ¿Cómo se acerca al caso base?

Actividad 4 – ¿Cómo funciona?




```
procedure digitoMaximo(n: integer; var max:
integer);
var
  dig: integer;
begin
  dig:= n mod 10;
  if ( dig > max ) then
    max:= dig;
  n:= n div 10;
  if (n <> 0) then
    digitoMaximo(n, max);
```



Actividad 5

Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

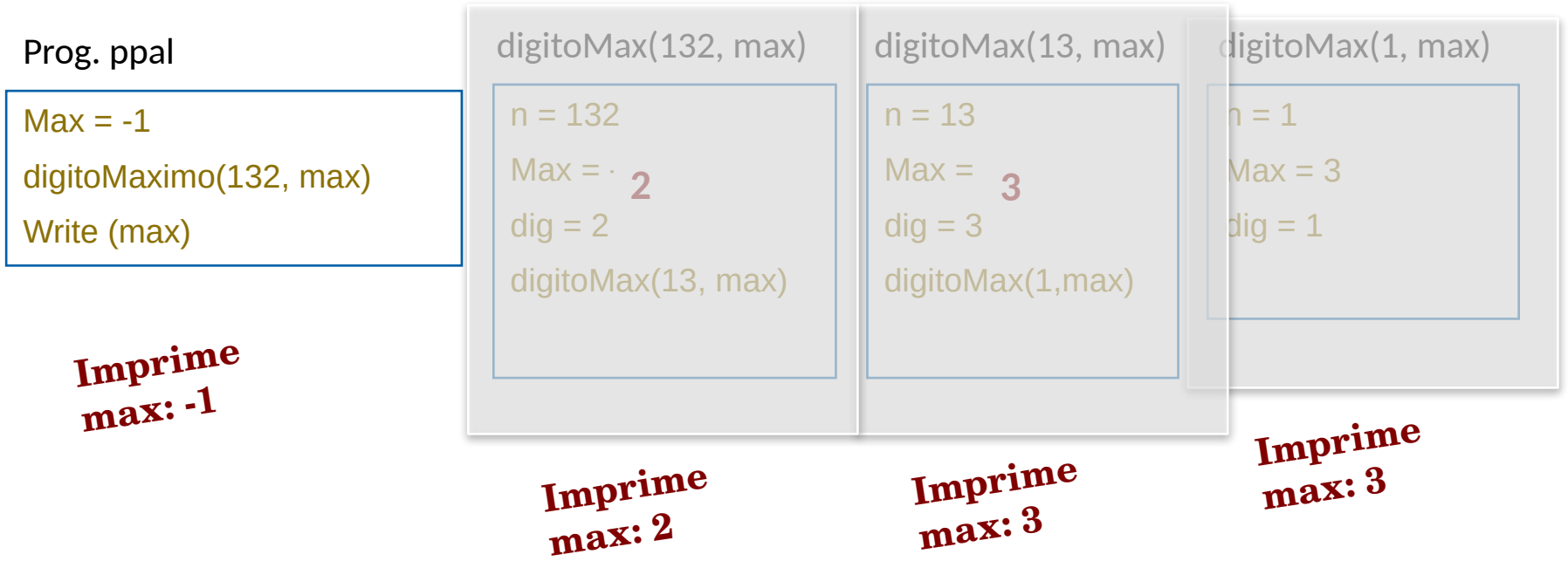
a) Modificar el procedimiento digitoMaximo. Debe colocarse la instrucción **`writeln ('max: ', max);`** como última instrucción del procedimiento.

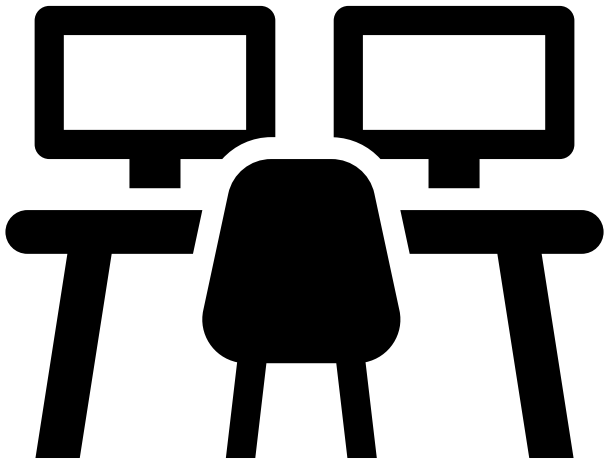
b) Compilar y ejecutar.

c) Responder:

- ¿Qué valor se muestra antes de finalizar cada módulo?
- ¿Qué valor se muestra en el programa principal?

Actividad 5 – ¿Cómo funciona?





Actividad 6

Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

a) Modificar el procedimiento `digitoMaximo`. Debe pasarse el parámetro `max` por valor.

b) Compilar y ejecutar.

c) Responder:

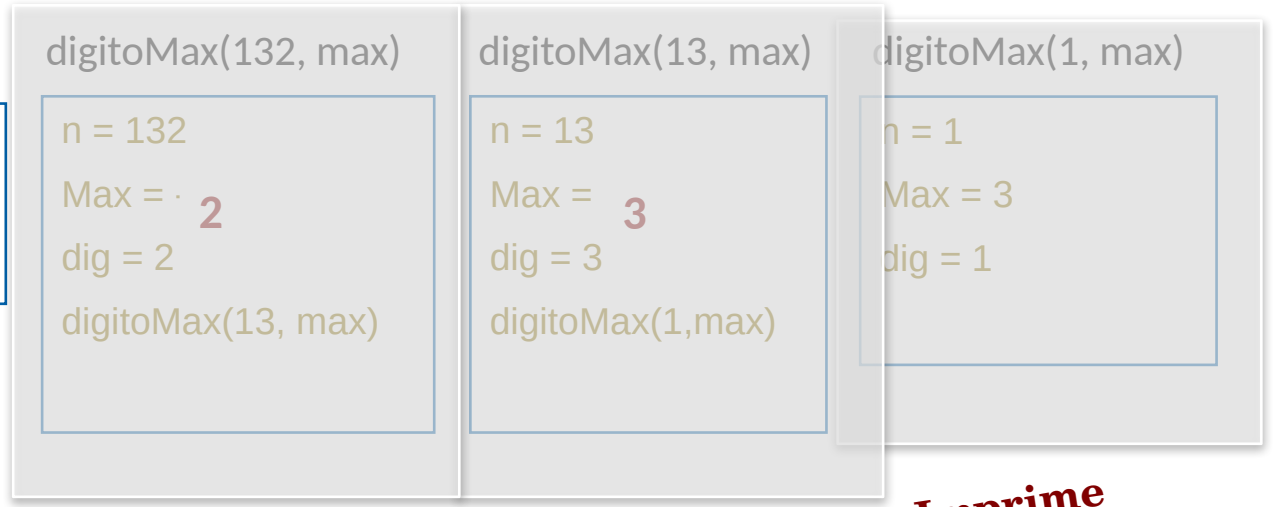
- ¿Qué valor se muestra antes de finalizar cada módulo?
- ¿Qué valor se muestra en el programa principal?

Actividad 6 – ¿Cómo funciona?

Prog. ppal

```
Max = -1
digitoMaximo(132, max)
Write (max)
```

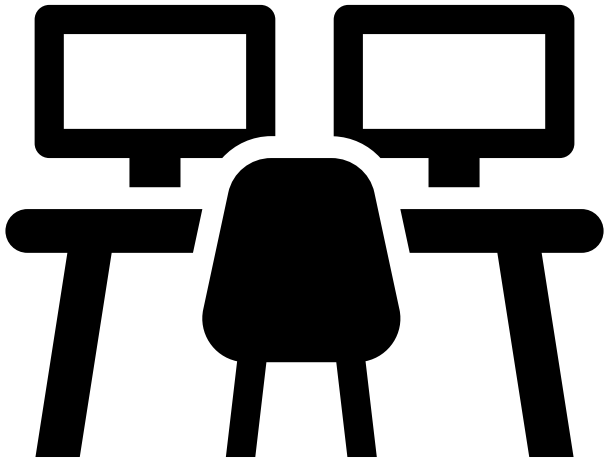
Imprime
max: -1



Imprime
max: 2

Imprime
max: 3

Imprime
max: 3



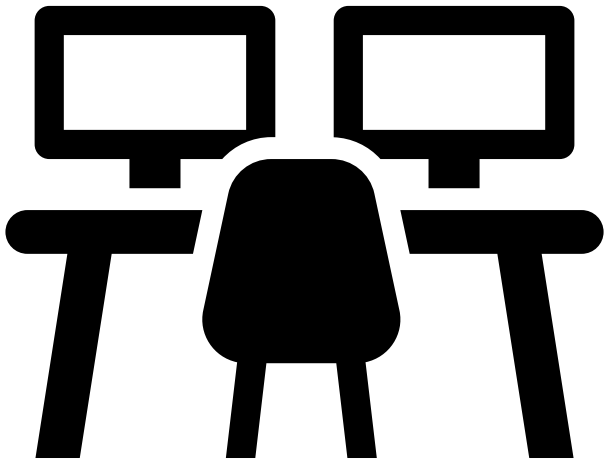
Actividad 7

Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

a) Escribir el procedimiento `digitoMaximo` como una función.

b) En el programa, leer un número, invocar a la función y mostrar el resultado.

c) Compilar y ejecutar.



Actividad 8

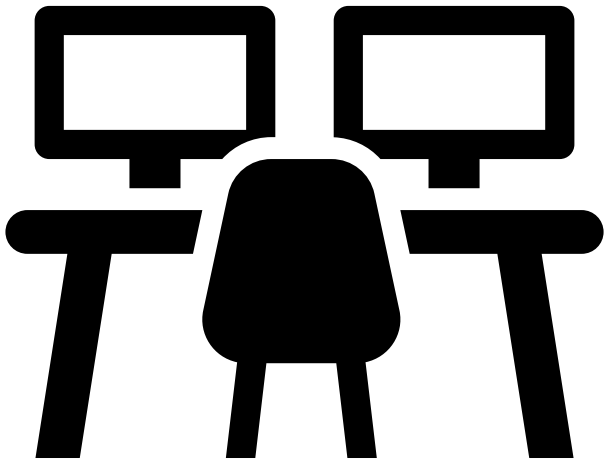
- **Descargar de Asignaturas:** `programaVectorOrd`

Utilizando ~~`programaVectorOrdenado`~~ realice las actividades:

a) Compilar y ejecutar.

b) Implementar el método de búsqueda dicotómica.

c) Utilizar el método implementado para buscar un val lee de teclado y mostrar el resultado.



Actividad 9

Descargar de Asignaturas: `programaVectores`

Utilizando **`programaVectores`** realice las siguientes actividades

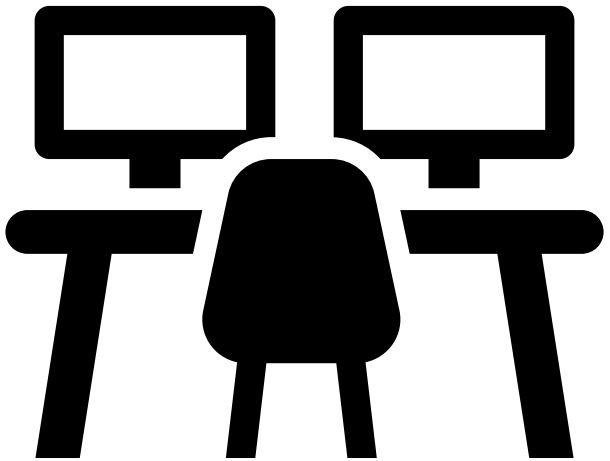
a) Compilar y ejecutar

b) Implementar un módulo recursivo **`Máximo`** que devuelva el máximo valor del vector.

c) Implementar un módulo recursivo **`Suma`** que devuelva la suma de los valores contenidos en el vector

d) Utilizar los módulos implementados para mostrar el máximo y la suma.

e) Compilar y ejecutar



Actividad 10

Descargar de Asignaturas: `programaLista`

Utilizando **`programaLista`** realice las siguientes actividades:

a) Compilar y ejecutar

b) Implementar un módulo recursivo **`Mínimo`** que devuelva el mínimo valor de la lista.

c) Implementar un módulo recursivo **`Imprimir`** que imprima los valores contenidos en la lista.

d) Utilizar los módulos implementados para mostrar el mínimo y la impresión.

e) Compilar y ejecutar