Práctica 6 Parámetros

> **Objetivo** Descubrir las técnicas existentes para pasaje de parámetros entre unidades y sus diferencias esenciales de acuerdo al lenguaje que lo implementa

Ejercicio 1

Explique brevemente los siguientes conceptos

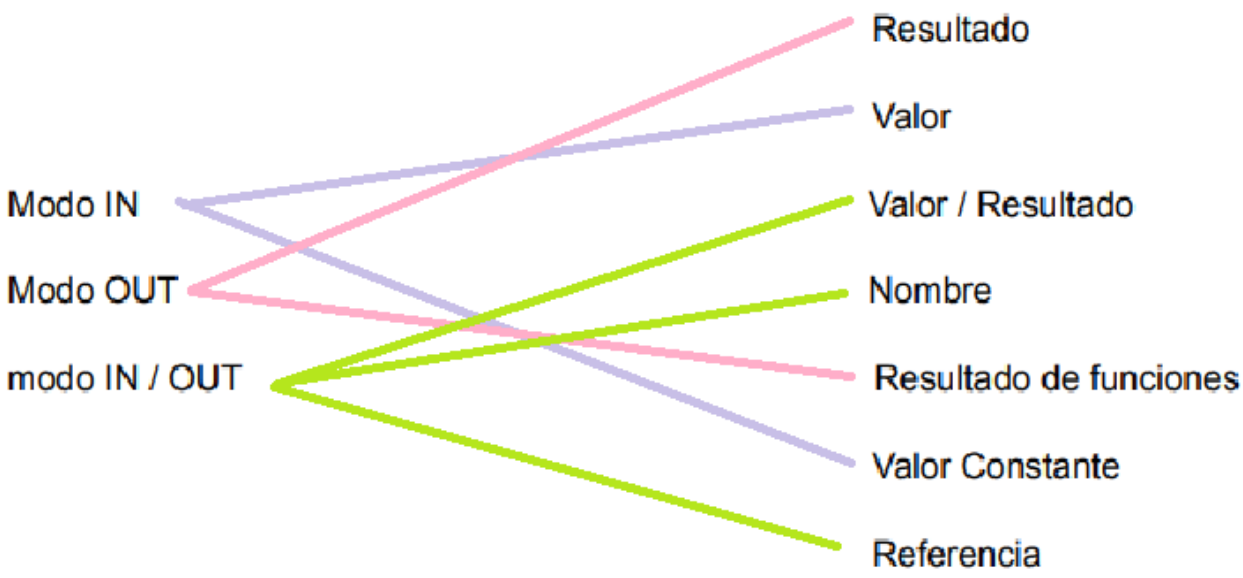
- **parámetro** es una variable que se utiliza para pasar información a una función o procedimiento. Los parámetros se ubican por lo general dentro de un paréntesis situado al lado del nombre del procedimiento o función.
- **parámetro real** es el valor real que se le pasa a la función o parámetro cuando se la llama y que se corresponde con el parámetro formal.
- **parámetro formal** es el nombre que se utiliza en la definición de una función o método para representar al parámetro que se espera se pase como argumento.
- **ligadura posicional** se corresponde con la posición que ocupan en la lista.
- **ligadura por palabra clave o nombre** se corresponden con el nombre por lo tanto pueden estar colocados indistintamente en la lista.

Ejercicio 2

Unir los siguientes puntos según corresponda y de una definición y un ejemplo
de cada par.

- **El modo IN** indica que el parámetro formal recibe el dato desde el real. Puede ser por valor (se “copia” el valor en el parámetro real) o por valor constante (en caso de que no se envíe o tome el parámetro real, el formal tiene un valor por default).
- **El modo OUT** indica que el parámetro real le envía el dato al parámetro formal. Puede ser por resultado (se devuelve el valor mediante el parámetro) o resultado de funciones (funciones)

- **En el modo IN/OUT** el parámetro formal recibe el dato del parámetro real y el parámetro formal le envía el dato al parámetro real. Puede ser por valor/resultado (el parámetro real obtiene el dato una vez finalizada la tarea), referencia (están ligadas al mismo espacio de memoria) o por nombre.



Ejercicio 3

a) Complete el siguiente cuadro según lo correspondiente a cada lenguaje:

Tipo de pasaje de parámetros	Lenguaje
Por valor, referencia, valor/resultado y por nombre.	ADA
Por valor y por referencia (usando punteros).	C
Por referencia y por valor (clonando el objeto).	Ruby
Por valor y por referencia (cuando se pasan objetos).	Java
Por referencia, sin embargo los enteros y las cadenas de caracteres se pasan por valor.	Python

b) Ada es más seguro que Pascal, respecto al pasaje de parámetros en las funciones. Explique por qué.

ADA es considerado más seguro en cuanto al pasaje de parámetros en las funciones debido a que ADA proporciona un sistema de tipos más riguroso y una mayor flexibilidad en la forma en que los parámetros se pasan y manipulan en las funciones. En Ada, es posible especificar el modo de pasaje de cada parámetro en la definición de la función, lo que permite controlar exactamente cómo se manejan los datos pasados a la función. Además, Ada ofrece la posibilidad de utilizar tipos abstractos y tipos protegidos, que permiten una mayor encapsulación y protección de los datos. Por otro lado, en Pascal, el pasaje de parámetros se hace principalmente por valor, y aunque también es posible pasar parámetros por referencia mediante el uso de punteros, esto puede aumentar el riesgo de errores de programación y corrupción de datos si no se manejan adecuadamente.

c) Explique cómo maneja Ada los tipos de parámetros in-out de acuerdo al tipo de dato

En ADA el manejo de los parámetros in-out depende del tipo de dato de los mismos. Para tipos simples se utiliza `in out` , mientras que para tipos complejos se utilizan punteros y para tipos de datos protegidos se utiliza la referencia del dato junto a `in out` .

Ejercicio 4

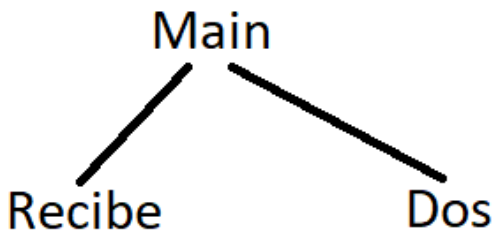
Sea el siguiente programa escrito en Pascal-like

```
Procedure Main;
var
    j, m, i: integer;
Procedure Recibe (x:integer; y:integer);
begin
    m:= m + 1 + y;
    x:=i + x + j;
    y:=m - 1;
    write (x, y, i, j, m);
end;
```

```
Procedure Dos;
var
    m:integer;
begin
    m:= 5;
    Recibe(i, j);
    write (i, j, m);
end;

begin
    m:= 2;
    i:=1; j:=3;
    Dos;
    write (i, j, m);
end.
```

a) Arme el árbol de anidamiento sintáctico y el registro de activación de cada una de las unidades.

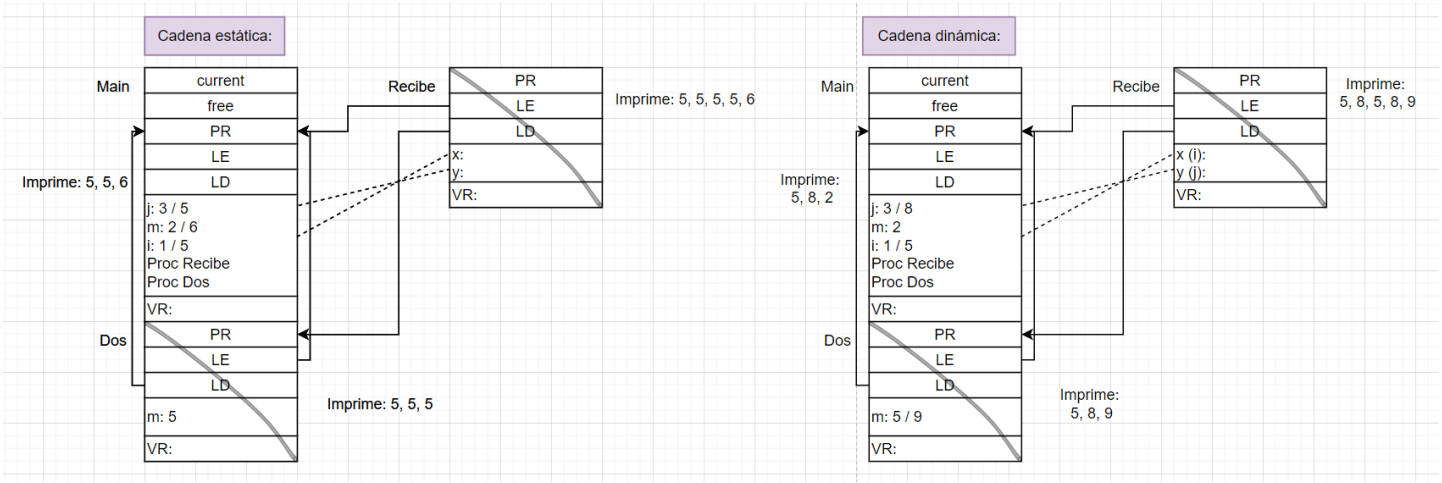


Main	Recibe	Dos
PR	PR, LE, LD	PR, LE, LD
j	x	m
m	y	
i	VR	VR
Recibe		
Dos		
VR		

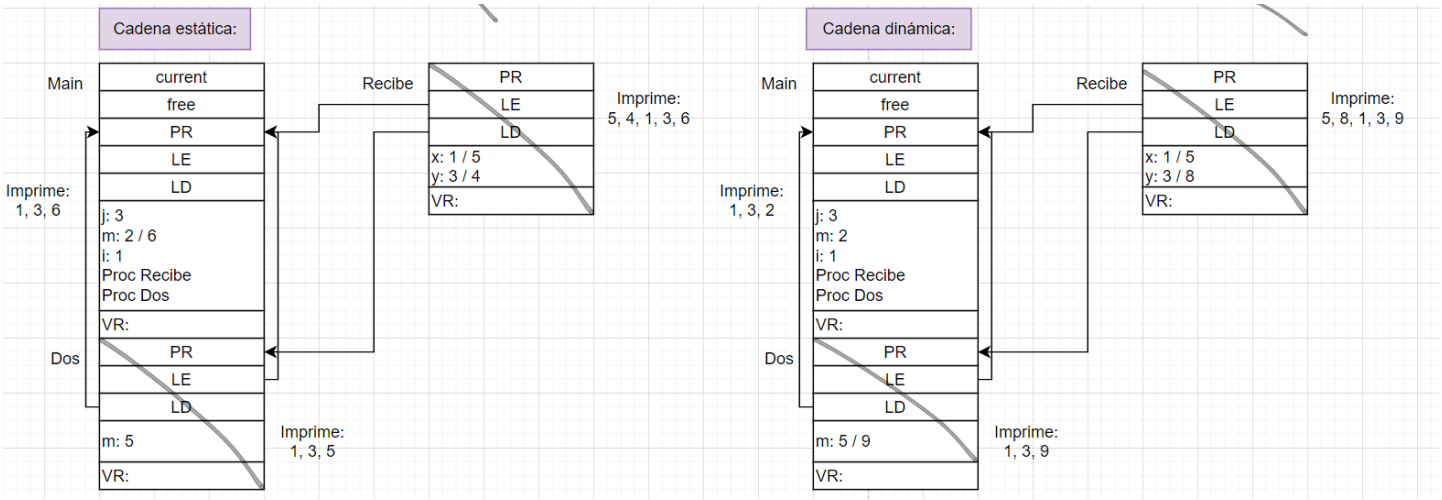
Main	Recibe	Dos

b) Decir qué imprime el programa suponiendo que para todas las variables que se pasan el pasaje de parámetros es por: (Deberá hacer la pila estática y dinámica para cada caso)

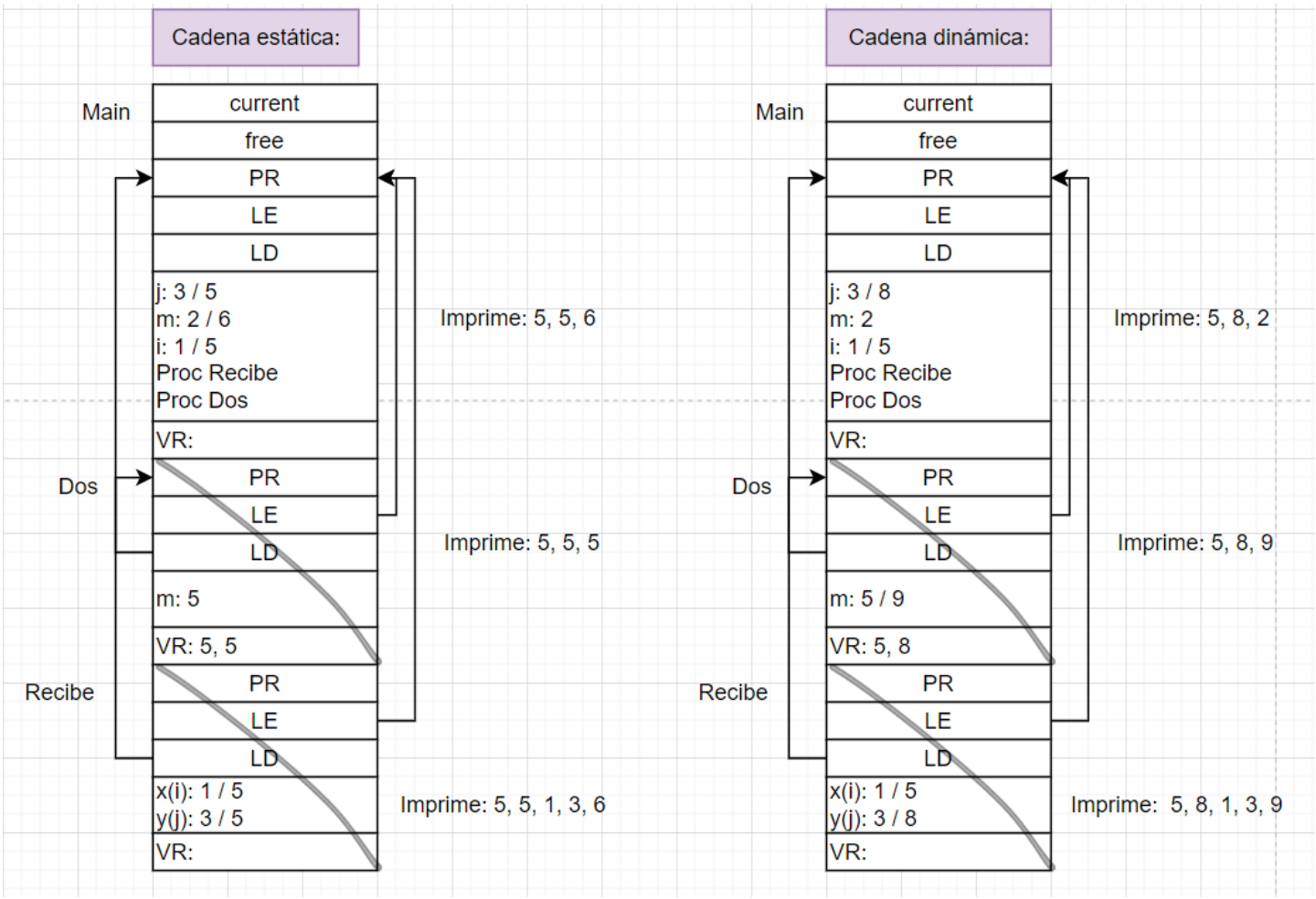
1) Referencia



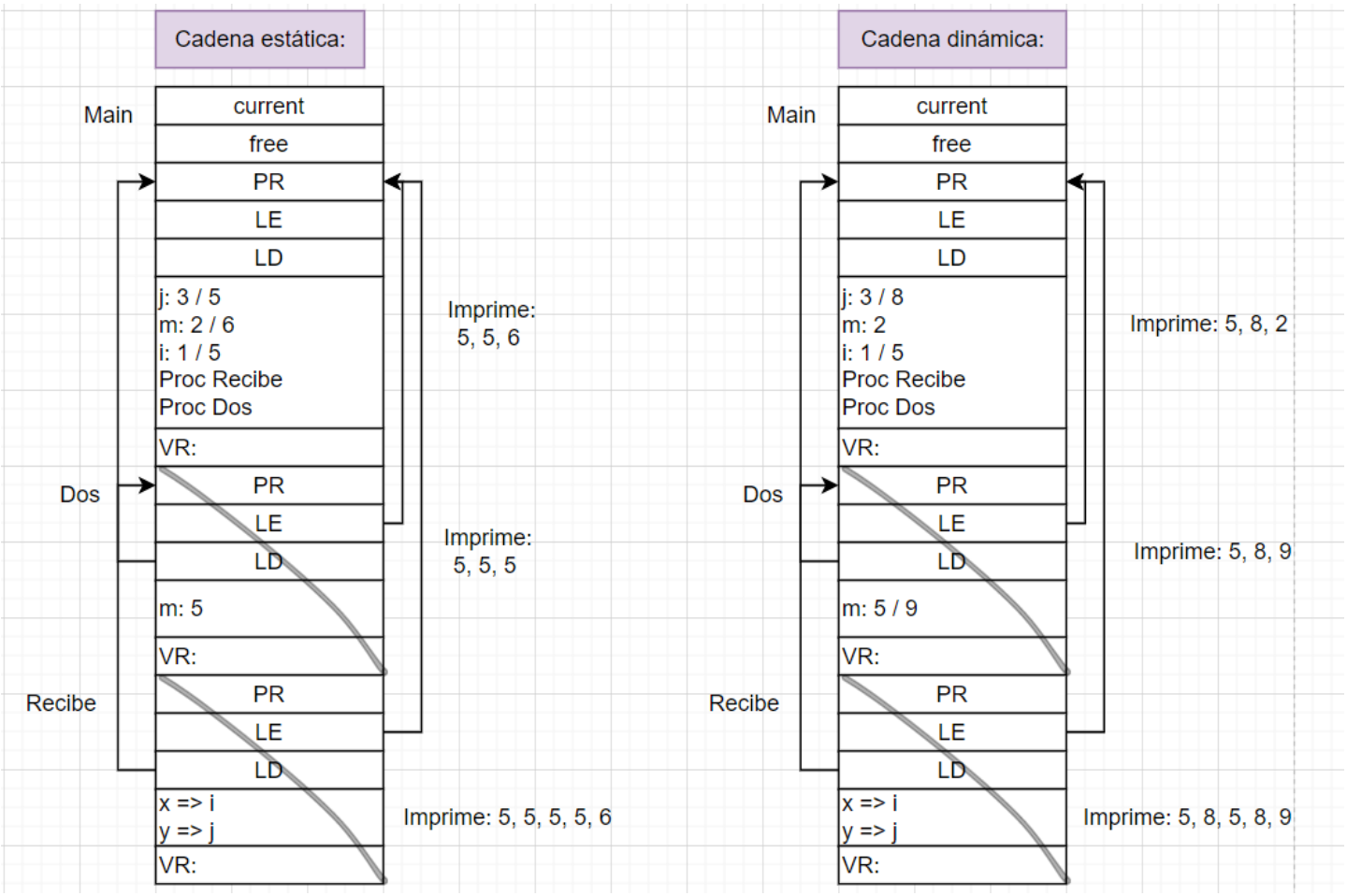
2) Valor



3) Valor resultado



4) Nombre



5) Resultado: No se puede realizar. Lo explico abajo

c) ¿Existió algún caso que no pudo realizarlo porque saltó algún tipo de error? Diga cuál y por qué.

Si, cuando el tipo de parámetro es resultado. Esta ejecución va a tener error puesto que cuando quiero realizar la operación `m:=m + 1 + y` el procedimiento Recibe no inicializa en ningún momento `y` por lo tanto tendrá un error.

d) ¿Dará el mismo resultado si se trata de un lenguaje que sigue la cadena dinámica? Justifique la respuesta realizando las pilas de activación

Si se refiere a todas las pilas de ejecución: varían los valores según si es estática o dinámica en la mayoría de los casos. Si se refiere al último caso, en ambas situaciones ocurrirá error.

Ejercicio 5

Suponiendo que se está ejecutando un programa con el siguiente registro de activación en memoria y se llama al procedimiento rutina(iter,vec,a). Determine el tipo de parámetro que se deben utilizar en el llamado para que los resultados sean los siguientes:

- **1)** `(4,6,7),(4,6,7), 2, 2` : todos por referencia
- **2)** `(3,5,6),(4,6,7), 2, 2` : **vec** por valor y **a** por referencia
- **3)** `(3,5,6),(5,5,6), 0, -1` : todos por valor

PR
LD
LE
Iter: true
Vec:[3,5,6]
a: -1
Rutina()
VR

```
......
procedura rutina(tipoParam iteracion,tipoParam vector,tipoParam vit):
    while iteracion begin
        vit = a+1
        vector[vit] = vector[vit]+1
        iteracion = (vector[vit] mod 2)==0
    end
    print vec
    print vector
    print vit
print a
.....

rutina(iter,vec,a)
```

Ejercicio 6

Indique con un ejemplo el comportamiento del parámetro por nombre (en el parámetro formal) para los siguientes casos de parámetros reales:

- **Un valor entero** Se comporta exactamente igual que el pasaje por referencia.
- **Una constante** Es equivalente al tipo por **valor**
- **Un elemento de un arreglo** Puede cambiar el suscripto entre las distintas referencias.
- **una expresión**

Valor entero

En este ejemplo se imprime 1.

Constante

El main imprimirá 5.

```
Program main
var i:integer;
Procedura A(nombre a:integer)
Begin
  a++;
end;
Begin
  i=0;
  Print(i);
End.
```

```
Program main
const int i = 5;
Procedura A(nombre a:integer)
Begin
  a = a + 6 * 7;
end;
Begin
  A(i);
  Print(i);
End.
```

Elemento de un arreglo

En este ejemplo se imprimirá 2.

```
Program main
var i:integer;
Procedura A(nombre a:integer)
var vec[1..3] of integer;
Begin
  vec[1]=0;
  a=a-1;
  vec[i]=a;
  vec[a+1]=1;
end;
Begin
  i=3;
  A(i);
  Print(i);
End.
```

Expresión: Se evalúa cada vez

Ejercicio 7

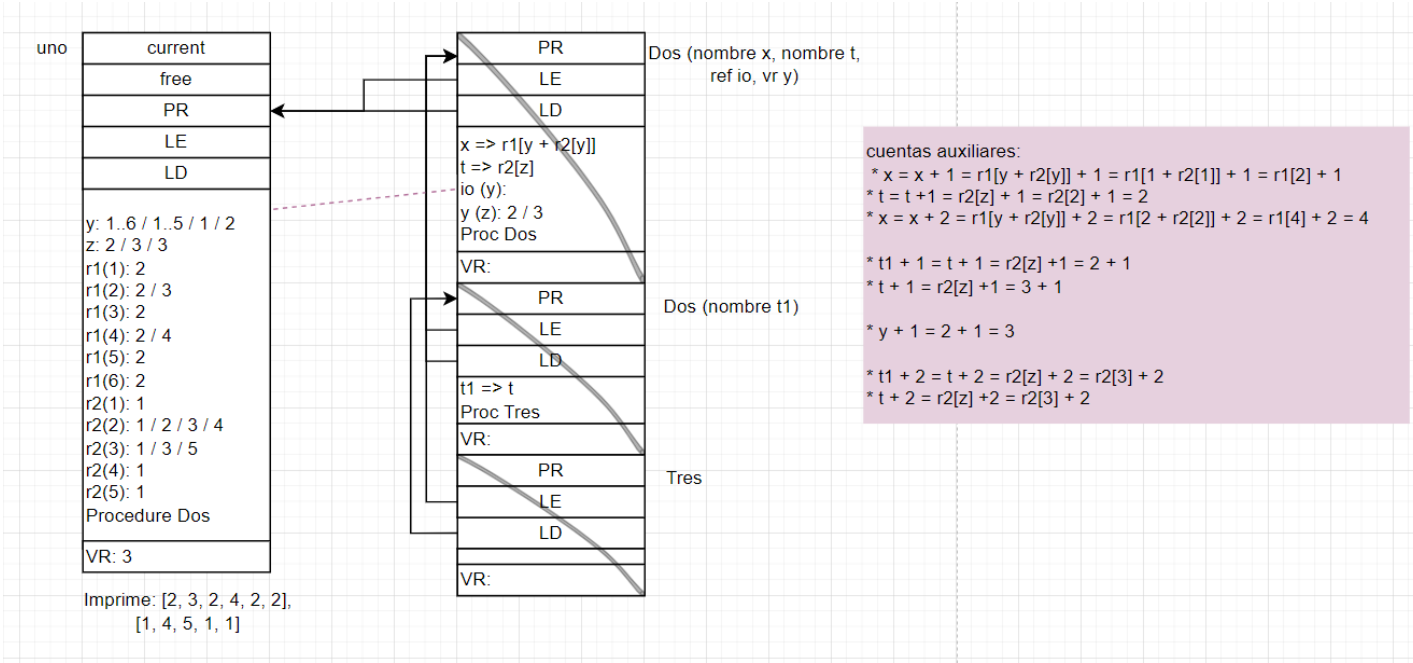
Realice la pila de ejecución del siguiente programa

```
Procedure Uno;
    y, z: integer;
    r1:array[1..6] of integer;
    r2:array[1..5] of integer;
Procedure Dos( nombre x, t:integer; var io:integer; valor-resultado y:integer);
    Procedure Dos( nombre t1:integer);
        Procedure Tres;
        begin
            y:= y + 1;
            z:= z + 1;
        end;
    begin
        t1:= t1 + 1;
        t:= t + 1;
        Tres;
        t1:= t1 + 2;
        t:= t + 2;
    end
begin
    x:= x + 1;
    t:= t + 1;
    io:= io + 1;
    x:= x + 2;
    if z =2 then Dos ( t );
end

begin
    for y:= 1 to 6 do r1(y):= 2;
    for y:= 1 to 5 do r2(y):= 1;
    z:= 2;
    y:= 1;
    Dos( r1( y + r2( y )), r2( z ), y, z);
    for y:= 1 to 6 do write (r1(y));
    for y:= 1 to 5 do write (r2(y));
end
```

a) Siguiendo la cadena estática

Cuando es por nombre tengo que tomar SIEMPRE los valores del contexto de la variable original, de su ambiente.



b) Siguiendo la cadena dinámica

Creo que la dinámica es igual así que ni ganas de hacerla.

Ejercicio 8

a) Indique las diferencias entre los pasaje de subprogramas como parámetros deep y shallow

Deep busca por los parámetros de la cadena estática y shallow por dinámica.

b) Realice la pila estática y dinámica tanto con el pasaje de parámetros deep y shallow para el siguiente código

```
Program A
    Var x:integer;
    Var y: char;

Procedure B;
    Var h:integer;
Begin
    h:=1+x;
    Write (y);
    C(D);
    Write (y);
End;

Procedure C (Subrutina S);
    Var x:integer;
    Var y: char;
Begin
    x:=3;
    y:= "b";
    x:=S(x,y)
    y:= "j";
    Write (x,y);
End
```

```
Function D (j:integer, k:char);
Begin
    j:=j+x;
    k:=y;
    Write (k);
    Return j;
End;
BEGIN
        x:=0;
        y:="a";
        B();
        Write (x,y);
END
```

Ejercicio 9

Sea el siguiente código escrito en Pascal like

```
Procedure main
    a: array(1..5) of integer;
    x: integer;
    i;integer;
Procedure Uno (tipo_pasaje m:integer)
Begin
    x:=0;
    x:=x+1;
    m:=m+x + a(3);
    x:=x*2;
    a(3):=a(3) - 1;
    m:=m+1;
End
```

```
Begin
    For i:=1 to 5 a(i):=1;
    x:=3;
    Uno(a(x));
    For i:=1 to 5 write (a(i));
End
```

a) Plantee diferencias, relacionada con la forma de implementación de cada uno y los resultados sobre este ejemplo considerando los siguientes tipos de pasajes parámetros nombre, referencia y valor resultado.

Si se pasa por `nombre` los valores impresos serían: `[3,2,0,1,1]` . En este tipo de pasaje de parámetros, el parámetro formal es sustituido textualmente por el parámetro real, se hace siempre alusión al parámetro real y todo su contexto.

Si se pasa por `referencia` los valores serían: `[1,1,3,1,1]` . En este tipo de pasaje de parámetros, se pasa la referencia o dirección de memoria de un objeto o variable en lugar de su valor. Los cambios realizados en la función a los parámetros pasados por referencia afectarán el objeto o variable original fuera de la función.

Si se pasa por `valor resultado` los valores serían: `[1,1,4,1,1]` . En este tipo de pasaje de parámetros, se pasa el valor de un parámetro a la función y la función devuelve un valor resultante al finalizar. Los cambios realizados en la función no afectarán el valor original fuera de la función, a menos que se asigne explícitamente el valor de retorno.

b) ¿Qué sucede si en Uno se agrega la siguiente declaración: x: integer? Indique el resultado para cada uno de los tipos de pasajes de parámetros (nombre, referencia y valor resultado)

Si se agrega la declaración de `x` dentro de Uno los valores serían:

- Por nombre: `[1,1,3,1,1]` .
- Por referencia: `[1,1,3,1,1]` .
- Por valor resultado: `[1,1,4,1,1]` .

Ejercicio 10

Sea el siguiente un programa escrito en Pascal

```
Program Uno;
    var x:integer;
Function Dos:integer;
begin
    x:= x + 1;
    return (x);
end;
```

```
Procedure Tres (pasaje x:integer);
begin
    x:= x + 5;
    x:= Dos + 10;
end;
begin
    x:= 8; Tres(x);
    write (x);
end.
```

a) Explique cómo simularía en Pascal el pasaje por valor-resultado y hágalo sobre este ejemplo.

> Nota: No se pueden agregar más variables, ni cambiar el nombre de las que están.

En este punto consideré si era por link estático y por link dinámico, pero tengo que corregirlo porque capaz está mal pensado.

Variable	Valores LE	Valores LD
x (Uno)	8, 19	8
x (Tres, apunta a x de Uno)	8, 13, 19	8, 13, 24
Dos	9	14

b Transcriba este ejemplo en Ada de manera tal que el resultado de la ejecución sea diferente si el pasaje de parámetros es por referencia y luego por valor – resultado

🥳😚