1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 0 1

</ Clase 5 - POO

- Repaso iteradores.
- Ejemplificación con iteradores y diccionario.
- Herencia.
- Ejercicios para resolver.

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</>

Repaso

} /> [

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Enunciado

Crear una farmacia, darle nombre y cargarle remedios. Luego:

Realizar un incremento del 20% a los remedios con stock menor a una cantidad determinada.
Eliminar remedios del laboratorio “Bagó”.
Modificar el precio del lotrial.

</Enunciado

```
"Crear una farmacia, cargarle nombre y remedios"
f:= Farmacia crear: 'Pato'.
s:=true.

[MessageBox confirm: 'Desea cargar más remedios?']whileTrue:[
    n:=Prompter prompt: 'Ingrese nombre del remedio'.
    p:=(Prompter prompt:'Ingrese precio del remedio')asNumber.
    st:=(Prompter prompt:'Ingrese stock del remedio')asNumber.
    lab:=Prompter prompt:'Ingrese laboratorio del remedio'.
    r:= Remedio crear: n con: p con: st con: lab.
    f agregar: r.
].
```

</Enunciado

"Realizar un incremento del 20% a los remedios con stock menor a una cantidad determinada"

```
stockMinimo:=[Prompter prompt: 'Ingrese el stock a validar']asNumber.
1 to: f tamanio do: [:i |
    rem := f recuperar: i.
    [rem verStock < stockMinimo] ifTrue: [
        rem modPrecio: rem verPrecio * 1.2.
    ].
].
```

</Enunciado

```
"Eliminar remedios del laboratorio “Bagó”"
i:=1.
[i <= f tamanio]whileTrue:[
	r:= f recuperar: i.
	[r verLab = 'Bagó']ifTrue:[
		f eliminar: r.
	] ifFalse:[ i := i+1. ].
].
```

</Enunciado

```
"Modificar el precio del lotrial"
pre := [Prompter prompt: 'Ingrese nuevo precio']asNumber.
1 to: f tamanio do: [:i |
	rem := f recuperar: i.
	[rem verNombre = 'Lotrial'] ifTrue: [
		rem modPrecio: pre.
	].
].
```

</>

¿Iterando?

} /> [

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</>

Diccionario

} /> [

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Diccionario

Generar un diccionario que guarde la cantidad de remedios por laboratorio.

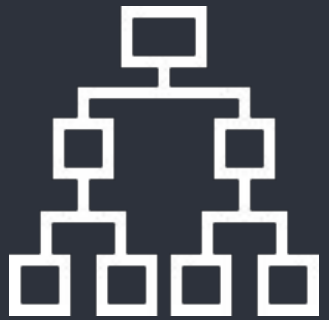
</Diccionario

```
dic:= Dictionary new.
col:= far verTodos.
colLab:= col collect:[:el | el verLab].  “tomo los nombres de todos los laboratorios”
sinRep:= colLab asSet. “saco los repetidos”

sinRep do: [: el| dic at: el put: ( colLab occurrencesOf: el)].
“recorro el diccionario por claves e imprimo Laboratorio y cantidad.”
dic     keysDo:[ :cla|   Transcript show: ‘Laboratorio: ', cla, ‘Cantidad de remedios:' , (dic at:cla) displayString. Transcript cr].
```

</>

Herencia



} /> [

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Herencia

En POO, la Herencia nos permite tener una jerarquía de objetos que puede tener varios niveles. Esta puede estar compuesta de clases abstractas [no pueden tener instancias] y clases concretas.

En la herencia llamamos **superclase** o ***clase padre*** a aquella que generaliza a las subclases agrupando comportamiento común de las mismas, y llamamos subclase a aquellas que ***especializan/especifican*** conceptos definidos de una superclase.

</Herencia: tipos

Hay ***dos tipos*** de herencia:

Estructural: en Smalltalk es total, las subclases heredan TODOS los atributos de la superclase. No hay forma de omitir alguno.

Comportamiento: es parcial, se puede no heredar un método de una superclase, redefiniéndolo en la subclase.

</>

Self y Super

} /> [

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Herencia: self y super

Para programar usando herencia se utilizan las pseudo variables self y super:

- **self:** hace referencia al objeto receptor del mensaje
- super: hace referencia a la superclase inmediata de la clase que contiene el método en el cual aparece dicha pseudovariable

</Herencia: self y super

Suponiendo la Jerarquía de clase A > B > C, y que cada una responde a los siguientes métodos

A	B	C
m1 ^3	m1 ^6	m1 ^self m4
m2 ^5	m4 ^self m2 + super m3	m2 ^9
m3 ^self m6 + self m2	m3 ^ 4	m7 ^super m6
	m6 ^self m2	

Evaluar las siguientes expresiones:

```
| unObjeto |
unObjeto := C new.
```

a] unObjeto m7

b] unObjeto m1

</Herencia: self y super

unObjeto := C new.
a] unObjeto m7
unObjeto Super m6
unObjeto Self m2
Resultado: 9

A	B	C
m1 ^3	m1 ^6	m1 ^self m4
m2 ^5	m4 ^self m2 + super m3	m2 ^9
m3 ^self m6 + self m2	m3 ^ 4	m7 ^super m6
	m6 ^self m2	

</Herencia: self y super

A	B	C
m1 ^3	m1 ^6	m1 ^self m4
m2 ^5	m4 ^self m2 + super m3	m2 ^9
m3 ^self m6 + self m2	m3 ^ 4	m7 ^super m6
	m6 ^self m2	

b] unObjeto m1

unObjeto Self m4

unObjeto Self m2 + Super m3

9 + (Self m6 + self m2)

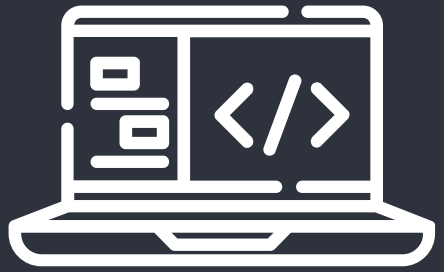
9 + (Self m2 + 9)

9 + 9 + 9

Resultado: 27

</>

Ejercicio Herencia



} /> [

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Ejercicio Herencia

CONSIGNA 1: Elaborar un diagrama de clases, con las siguientes clases:

1] Alumno.
2] Docente.
3] Ayudante.
4] Persona.
5] Titular de cátedra.
6] Docente adjunto.
7] Facultad.
8] Cátedra.

Pista: Un modelo PERFECTO de esta consigna estaría resuelto implementando composición y herencia.

</Ejercicio Herencia

CONSIGNA 2: ubique los atributos que considere importantes en su clase correspondiente, teniendo en cuenta que:

A] De cada individuo se quiere saber nombre,edad y nacionalidad.
B] De todos los que cursen en la facultad se quiere saber su legajo.

</Ejercicio Herencia

CONSIGNA 2: ubique los atributos que considere importantes en su clase correspondiente, teniendo en cuenta que:

C) De los docentes el nombre de la materia que dictan y de su obra social, si están a cargo de la materia se quiere saber el año en que se les dio el cargo y si no están a cargo de la materia se quiere saber el nombre del docente a cargo de esa materia. que dictan.
D) De los ayudantes se quiere saber la categoría (1,2) a la que pertenecen, y si son remunerados o no.

</Ejercicio Herencia

CONSIGNA 2: ubique los atributos que considere importantes en su clase correspondiente, teniendo en cuenta que:

E) Por cada cátedra, interesa saber: código, nombre, carga horaria, carrera, docente a cargo, tipo [Curricular o Electiva] y nivel al que pertenece [1,2,3,4 o 5]
F) De la facultad, es de interés saber el nombre de la universidad, y su localidad.