Objetos

1) Abstracción de elementos del mundo real en la programación

Perro:
- Color
- Numero de patas
- Altura
- Peso
- Ancho
- Largo

} Características

ladrar()
morder()
moverse(x,y)
comer(x)

} Acciones

Estado

2) Encapsulación. Capacidad del objeto de definir cómo se muestran y modifican sus campos internos
Hacer de cuenta que el objeto es una caja cerrada (negra) controla cómo se accede o modifica la información

Acceso: public, privated, protected

Clases: Especificación de los campos y métodos en un paradigma orientado a objetos
Templates / Plantillas de los objetos

Clases, clases abstractas, interfaces

public abstract class pepito {} public interface pepito {}

Plantilla

Definición

Relaciones entre clases: herencia, uso y composición

Polimorfismo: Capacidad de tener varias versiones de un instanciación de una clase
- Sobreescritura de métodos
- Instanciación a través de otra clase

1. Programa ahora va ser una declaración de clases seguido una expresión
2. Vamos a tener un objeto generico llamado object
3. super que para llamar metodos de una clase padre
4. self que es una apuntador del instancia del objeto funciona como this

```
class c1 extends object
  field i
  field j
  method initialize (x)
    begin
      set i = x;
      set j = -(0,x)
    end
  method countup (d)
    begin
      set i = +(i,d);
      set j = -(j,d)
    end
  method getstate () list(i,j)
```

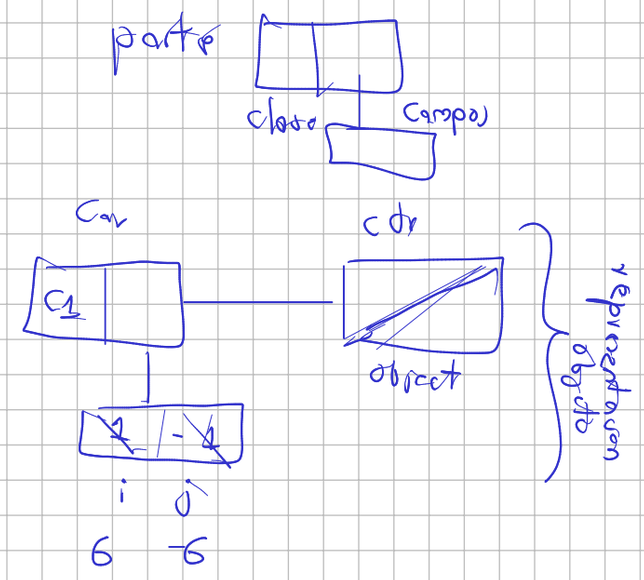
0
1
-1
6
-6

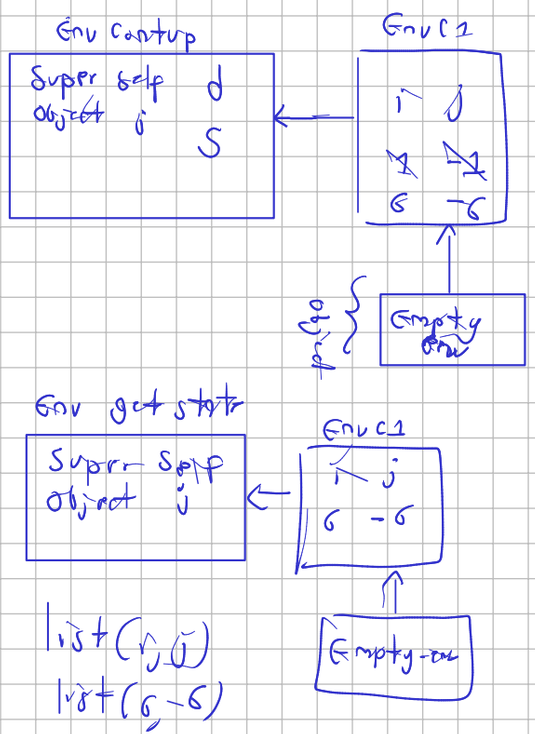
```
let
  j = new c1(1)
in
  begin
    send j countup(5);
    send j getstate()
  end
```

(-6, -6)

Simple
Planos

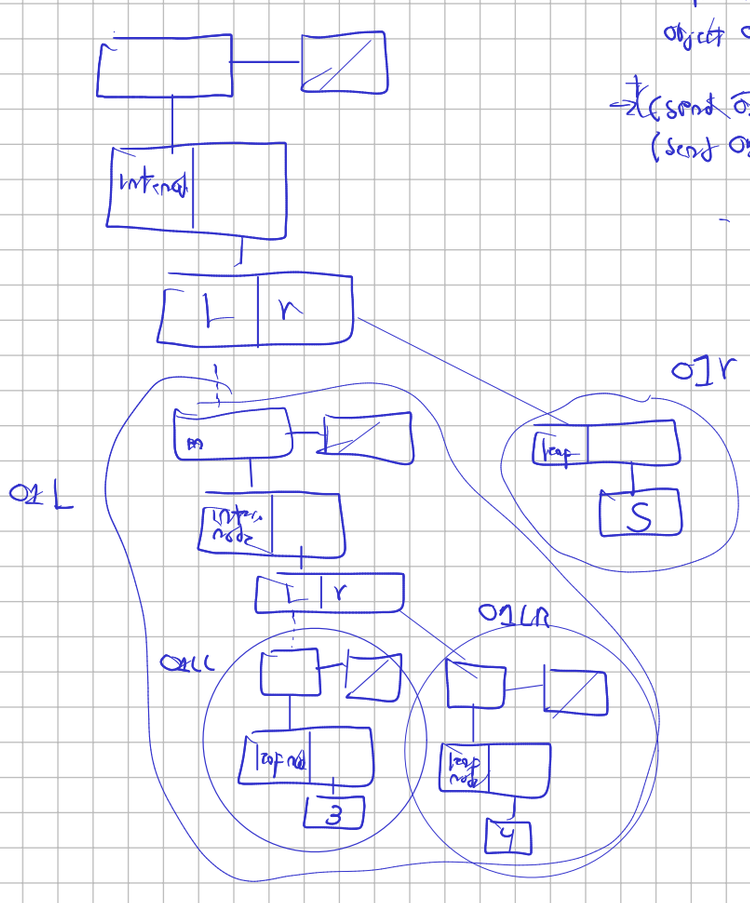
Simplificar objeto lista de partes

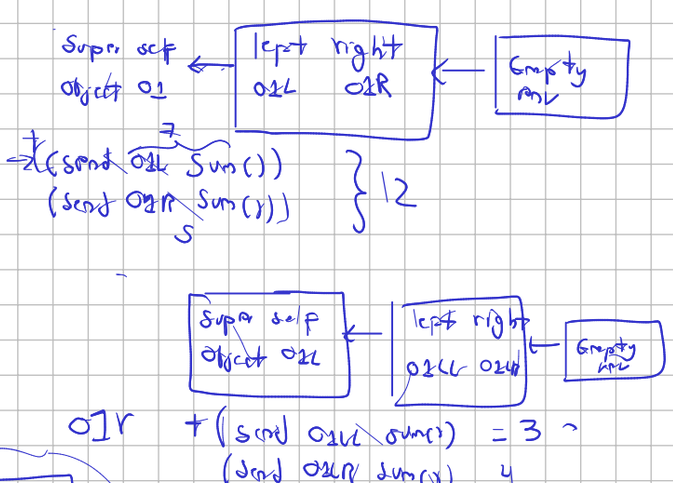




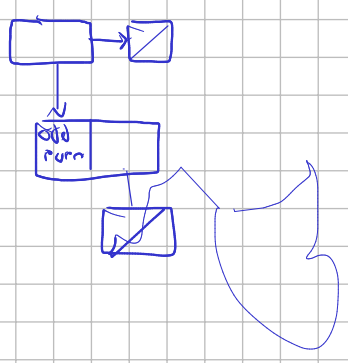

```
class interior_node extends object
  field left
  field right
  method initialize (l, r)
    begin
      set left = l;
      set right = r
    end
  method sum () +(send left sum(),send right sum())

class leaf_node extends object
  field value
  method initialize (v) set value v
  method sum () value
```

```
let
  o1 = new interior_node(
         new interior_node(new leaf_node(3),
                           new leaf_node(4)),
         new leaf_node(5))
in
  send o1 sum()
```

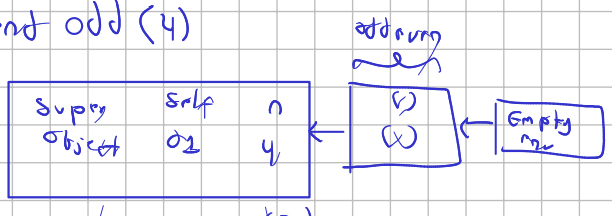




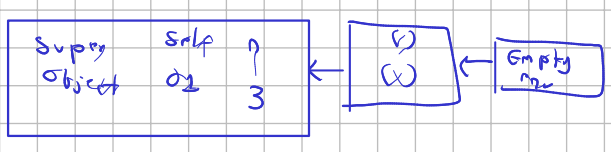
```
class oddeven extends object
  method initialize () 1
  method even (n)
    if zero?(n) then 1 else send self odd(sub1(n))
  method odd (n)
    if zero?(n) then 0 else send self even(sub1(n))

let o1 = new oddeven()
in send o1 odd(4)
```



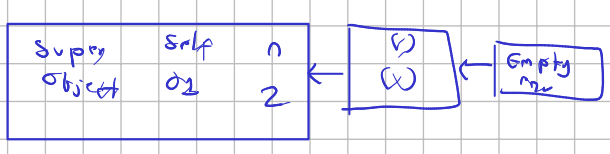
Send odd (4)



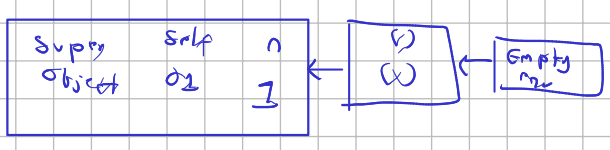
Send o1 even(3)



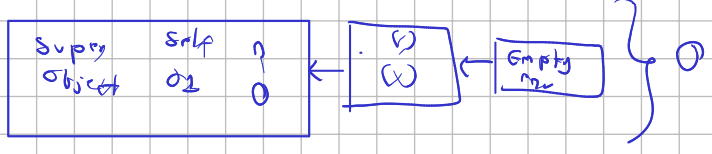
Send o1 odd(2)



Send o1 even(1)

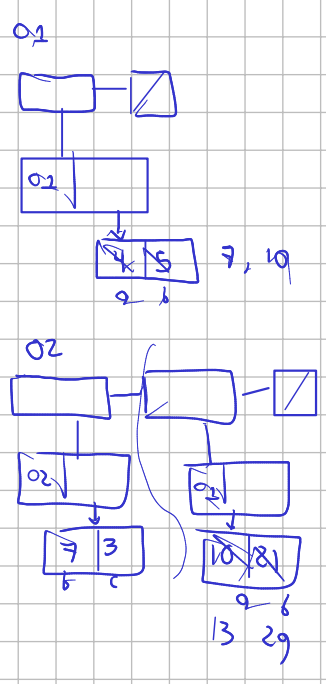


Send o1 odd(0)



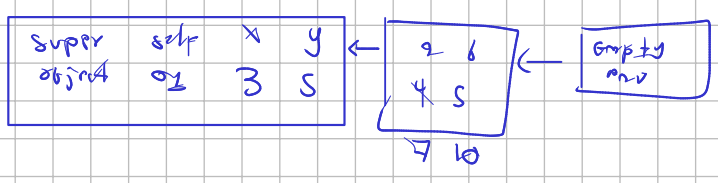

```
class m1 extends object
    field a
    field b

    method initialize(x,y)
        begin
            set a = x;
            set b = y
        end

    method k(x,y)
        begin
            set a = +(x,a);
            set b = +(y,b);
            send self q(+(a,x)
                ,+(b,y))
        end

    method q(x,y)
        +(x,y,a,b)
```

```
class m2 extends m1
    field b
    field c

    method initialize(x,y)
        begin
            super initialize(+(x
                ,y),*(x,y));
            set b = x;
            set c = y
        end

    method q(x,y)
        begin
            set b = +(x,a);
            set c = +(y,b);
            +(a,b)
        end

let

    o1 = new m1(4,5)
    o2 = new m2(7,3)
in
    let

        u = send o1 k(3,5)
        v = send o2 k(3,8)
        in
            +(u,v)
```

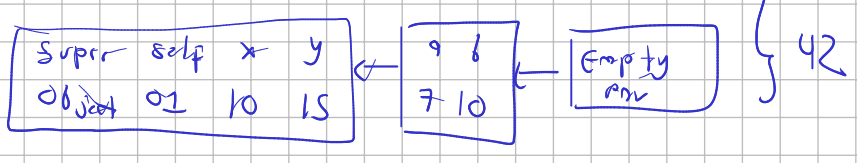


84

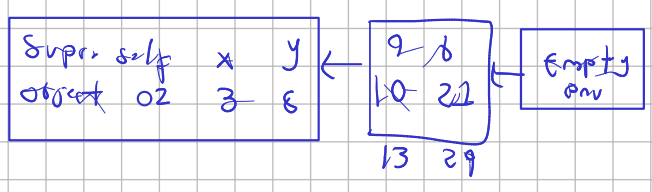
Send o1 k(3,8)



Send o1 q(10, 15)



Send o2 k(3,8)



Send o2 q(16, 37)