



Programación

DESCOMPOSICIÓN

División en subtarear

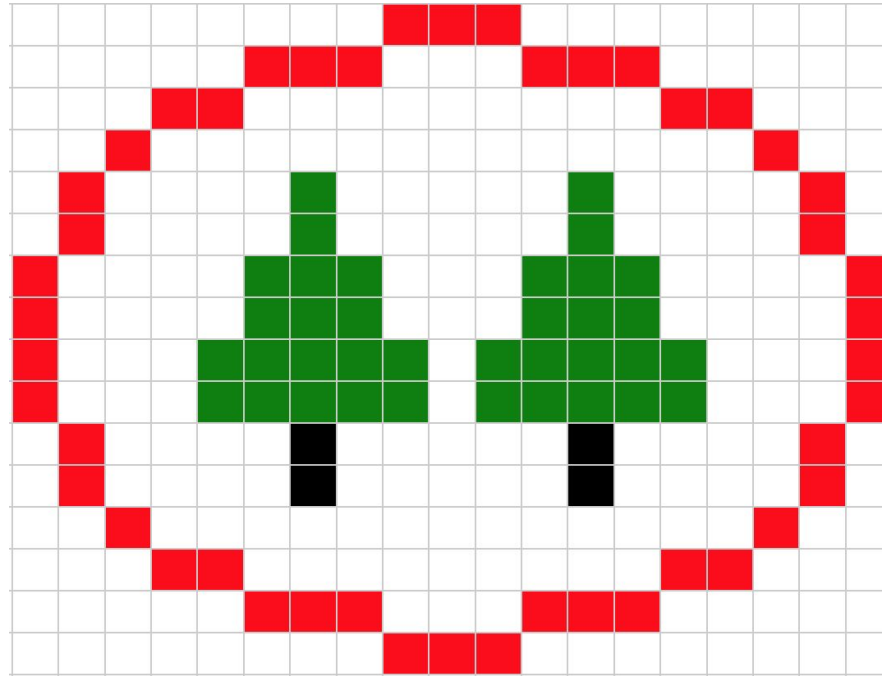
Universidad Nacional de Quilmes

Precalentando motores

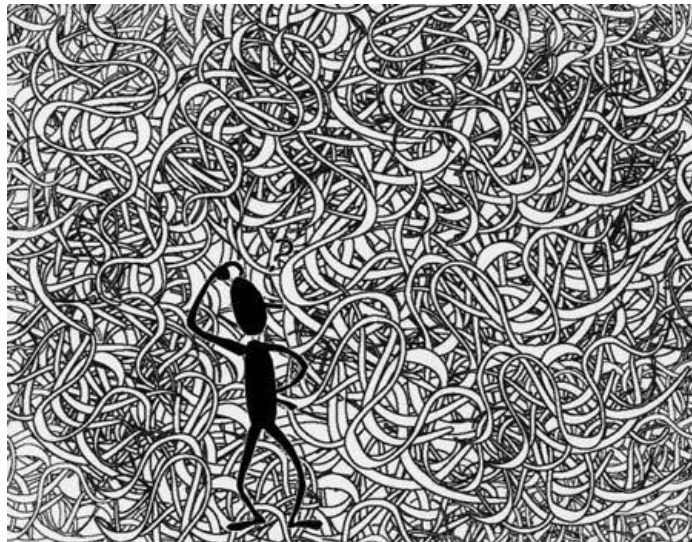


Ejercicio: Logo con árboles

Realicemos el siguiente dibujo en QDraw.



¿Por dónde arrancamos?





=




Recordemos siempre que **Programar**
es comunicar

Intentando comunicar la solución

Si le intentamos contar a alguien que es lo que hay que hacer, terminaremos diciendo:

arriba, arriba, arriba, pinta, arriba, pinta... etc.

Esta solución presenta varios problemas:

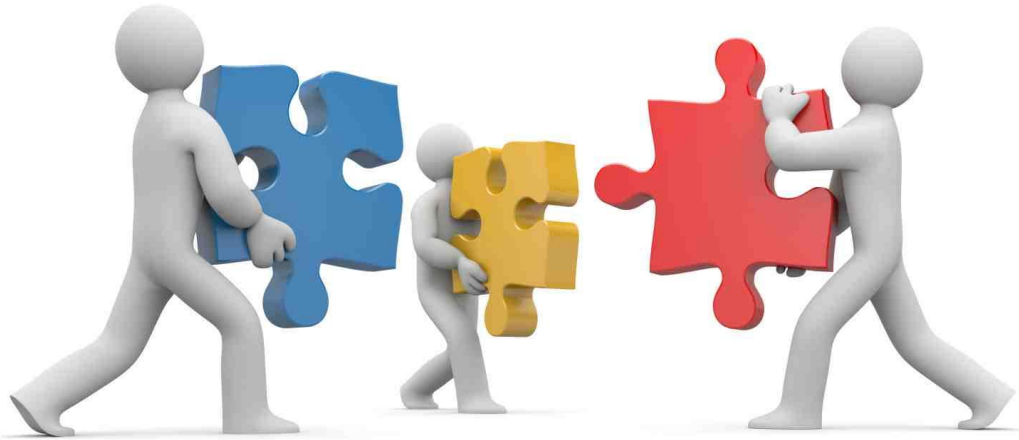
- El código es confuso hasta para nosotrxs mismos.
 - La otra persona no tiene idea de qué le estamos hablando.
 - Los comentarios pueden ayudar a entender el código, pero no solucionan el problema de fondo.
 - Uno quisiera poder transmitir la idea de una forma más sencilla.
- 

Lo que quisiéramos transmitir es algo del estilo:

```
programa {  
    dibujar óvalo externo en rojo  
    dibujar árbol derecho  
    dibujar árbol izquierdo  
    volver el cabezal a la posición inicial  
}
```



Divide y vencerás



Divide y vencerás

Al dividir el problema general en problemas más pequeños, podemos centrarnos en resolver cosas más sencillas, que requieren menos código y son más fáciles de razonar.

De esta forma es más fácil atacar problemas grandes, para arribar a una solución integral.



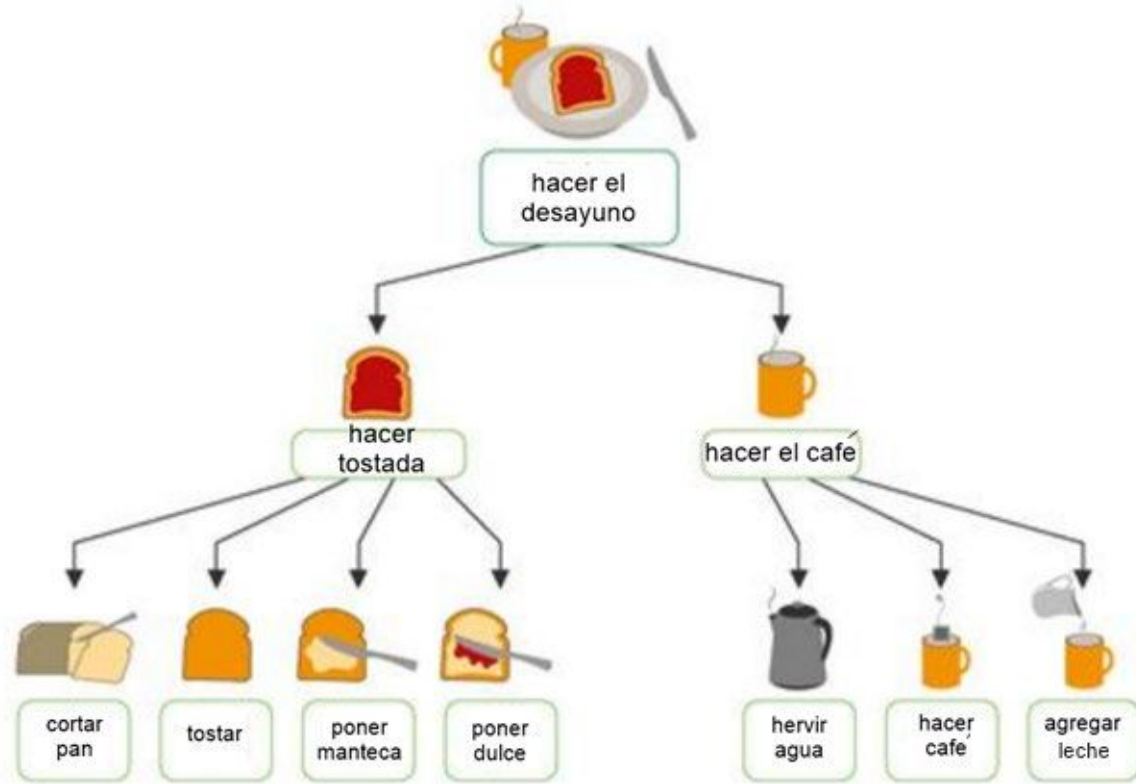
Técnica Top-Down

Una manera de dividir el problema en partes más pequeñas, es mediante la técnica **Top-Down**, que consiste en un diagrama (cajitas) que grafica las distintas partes del problema en forma de árbol, donde cada cajita representa una porción del problema, el cual se vuelve a dividir en otras partes aún más pequeñas, y así sucesivamente.

Veamos un ejemplo

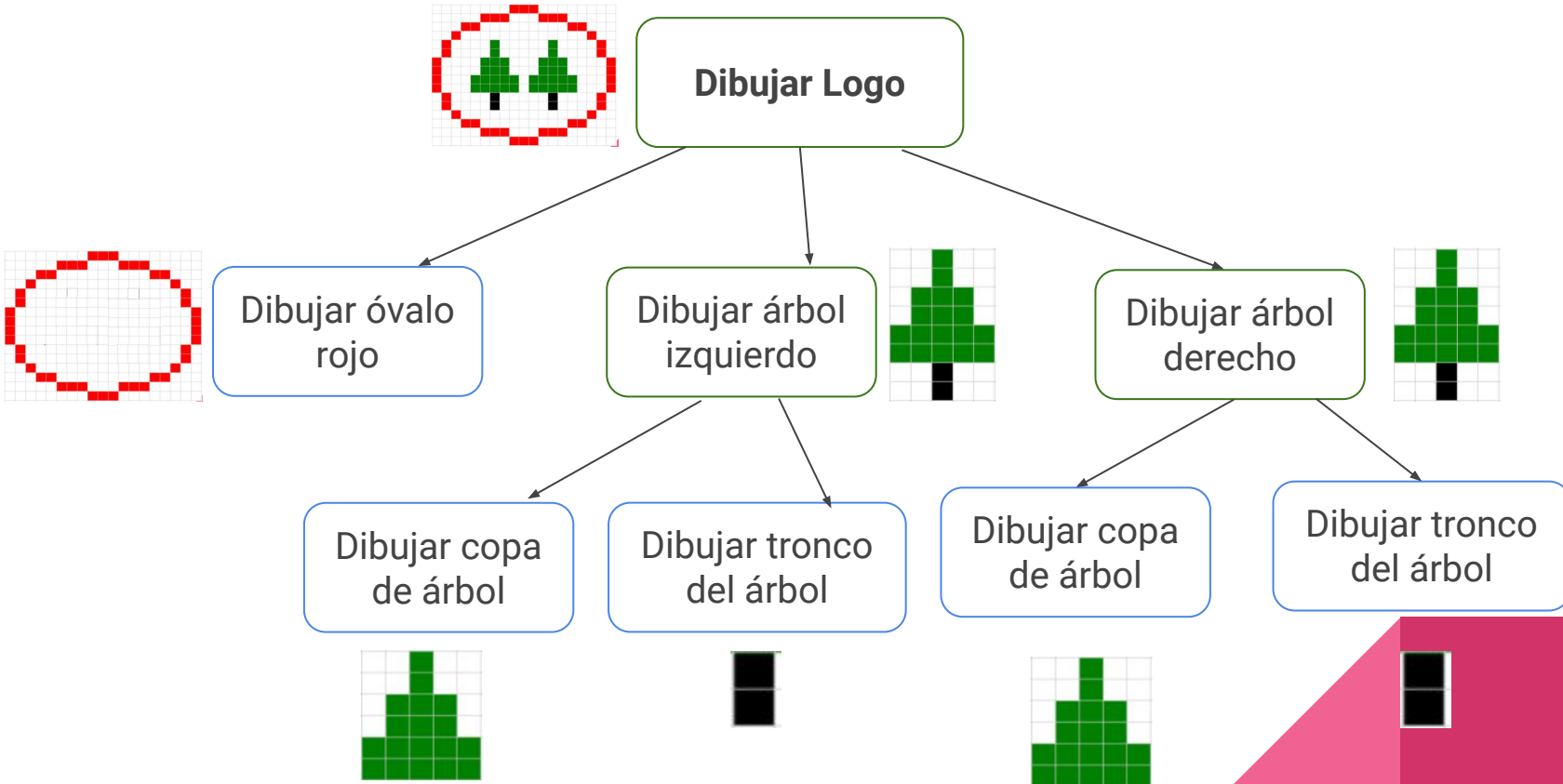


Hacer el desayuno



Fuente: Imágen perteneciente al material proporcionado por [UNIFE](#)

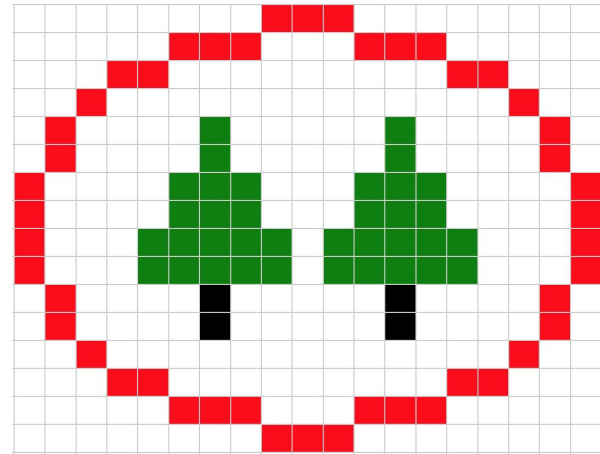
Volviendo al ejercicio del dibujo del logo



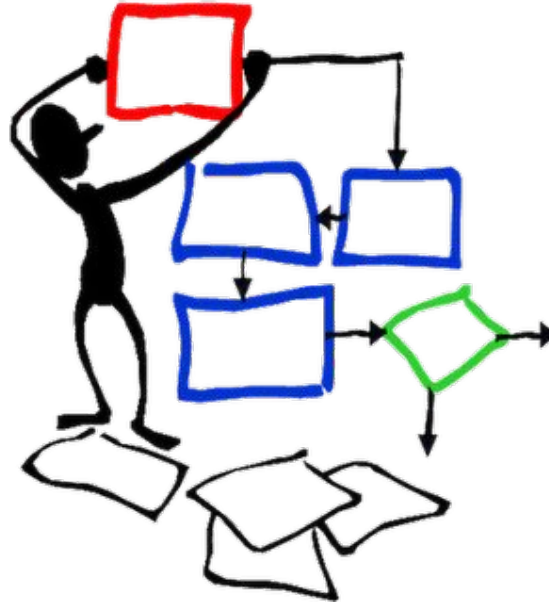
Escribiendo las acciones de cada problema

Dibujar logo

- ↳ Dibujar óvalo rojo
 - ↳ Arriba, arriba, derecha, pintar de rojo ...
- ↳ Dibujar árbol izquierdo
 - ↳ Dibujar copa del árbol
 - ↳ Pintar de verde, arriba, pintar
 - ↳ Dibujar tronco del árbol
 - ↳ Pintar de negro, arriba, pintar
- ↳ Dibujar árbol derecho
 - ↳ Dibujar copa del árbol
 - ↳ Pintar de verde, arriba, pintar
 - ↳ Dibujar tronco del árbol
 - ↳ Pintar de negro, arriba, pintar



Procedimientos



Procedimientos

¿Qué son?

Los procedimientos son una **forma de estructurar el código** para reflejar estos esquemas mentales que hemos comentado.

Un **procedimiento** es una **nueva instrucción definida por el usuario** que representa la solución de una parte del problema.

Sintaxis:

Un procedimiento se define mediante la palabra **“procedimiento”**, seguida de un **nombre** (no puede contener espacios y comienza con mayúscula), paréntesis vacíos, la sección de **documentación** (sintaxis de **comentarios**) y su correspondiente **bloque de código** entre llaves.

Ejemplo

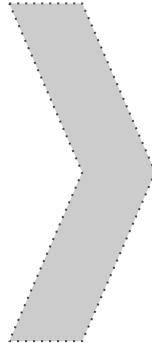


procedimiento DibujarTroncoDeArbol () {

/* **PROPÓSITO:** Dibujar el tronco de un árbol de color negro. El cabezal inicia y finaliza en el extremo inferior del tronco.

PRECONDICIÓN: Existe al menos 1 celda hacia arriba desde el inicio del cabezal.*/

```
PintarNegro  
MoverArriba  
PintarNegro  
MoverAbajo  
}
```



Recordar que consiste en **definir una nueva instrucción**, a partir de las instrucciones primitivas del lenguaje Qdraw, donde la secuencia de estas primitivas cumplen con un propósito. Por lo cual es importante que el nombre **comience con un verbo**.

Ahora tenemos Instrucciones y Procedimientos

Primitivas: set limitado de instrucciones

- MoverDerecha
- MoverArriba
- MoverIzquierda
- MoverAbajo
- PintarNegro
- PintarRojo
- PintarVerde
- Limpiar

Procedimientos: instrucción definida por el usuario

- DibujarTroncoDeArbol ()

Llamar (invocar) procedimientos

Cada **procedimiento** tiene dos etapas:

- Definición. Ya vimos como es la sintaxis para definir un procedimiento (diapo 15)
- Invocación / llamado. Cómo usamos a los procedimientos que definimos.

Definición: se define con un nombre, la sección de documentación y el bloque de código correspondiente al algoritmo que resuelve el problema.

Invocación: se llama (**invoca**) a través de su nombre, y desde cualquier bloque del programa o desde otro procedimiento.

Ejecución: al momento de **ejecutar el programa**, en el lugar desde donde se invoca al procedimiento, se ejecuta el **bloque de código** correspondiente en su definición.

IMPORTANTE: Un procedimiento **se define una sola vez**, pero se puede **invocar** tantas veces como sea necesario.



Llamar a procedimientos - Sintaxis - Ejemplo 1

Para llamar a un procedimiento basta utilizar el nombre del mismo seguido de paréntesis.

Invocación:

```
programa {  
    DibujarTroncoDeArbol()  
    ...  
}
```



Definición:

```
procedimiento DibujarTroncoDeArbol () {  
    /* */  
    PintarNegro  
    MoverArriba  
    PintarNegro  
    MoverAbajo  
}
```



Estado inicial tablero

(antes de ejecutarse
el procedimiento)



Estado final tablero

(después de ejecutarse
el procedimiento)

Llamar a procedimientos - Sintaxis - Ejemplo 2

Pueden llamarse más de una vez, aunque se define una sola vez.

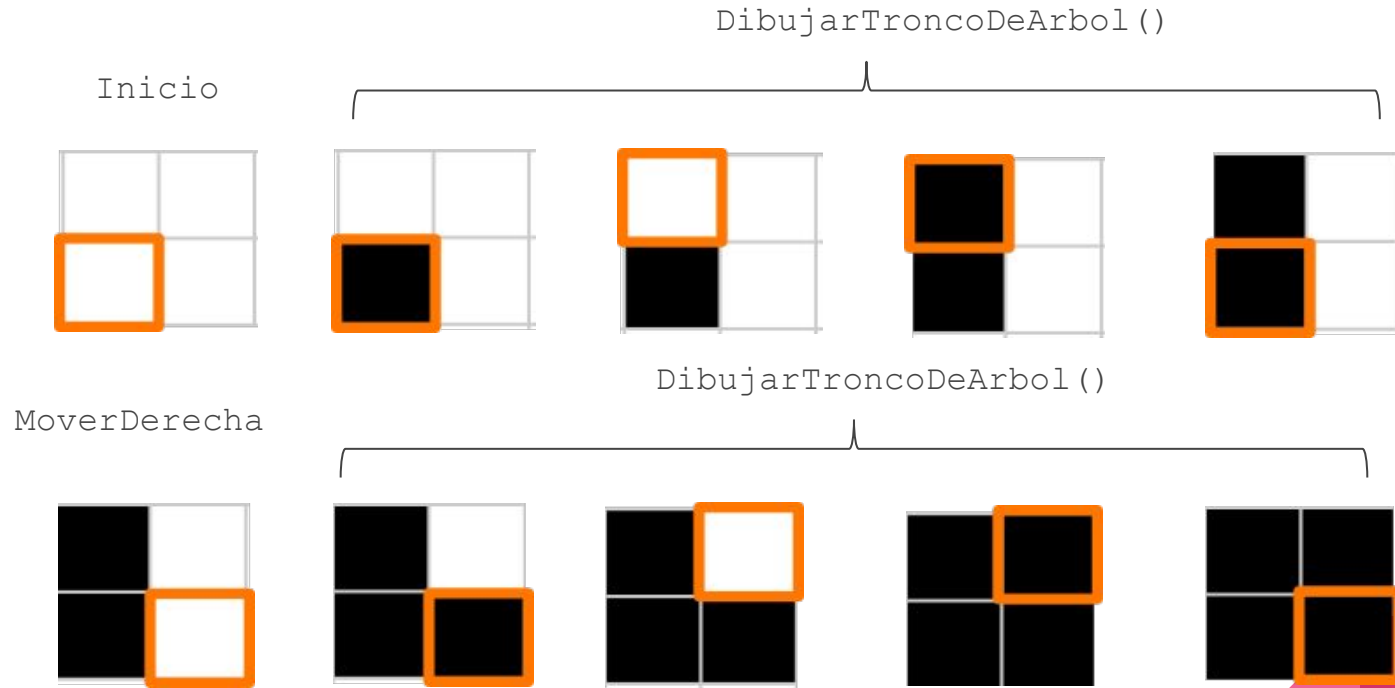
Invocación desde el programa:

```
programa {  
    DibujarTroncoDeArbol()  
    MoverDerecha  
    DibujarTroncoDeArbol()  
}
```



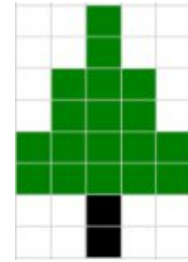
Estado final tablero

Simulación de la ejecución del programa anterior



Invocar a un procedimiento desde otro procedimiento

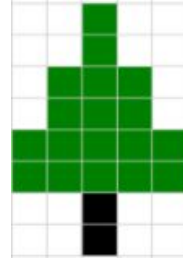
```
procedimiento DibujarArbol () {  
    DibujarTroncoDeArbol()  
    IrACopa()  
    DibujarCopaDelArbol()  
}
```



Cada procedimiento debe tener su correspondiente documentación

Dibujamos el árbol

```
programa{  
    /* ... */  
    DibujarArbol ()  
}
```



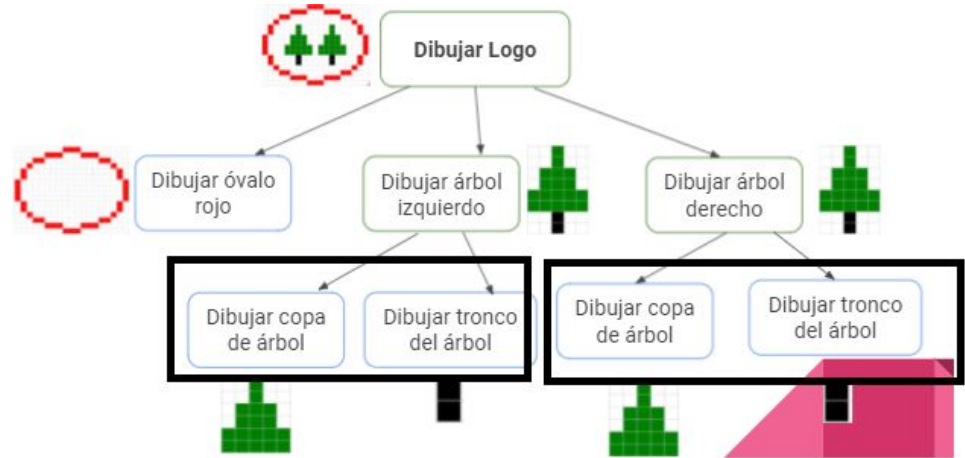
Estado final tablero

Nota: este programa se resume en llamar al procedimiento principal, y nada más

Dibujamos el logo

Si volvemos nuevamente al dibujo del logo, podemos obtener la siguiente solución final del problema:

```
programa {  
    DibujarLogo()  
}  
procedimiento DibujarLogo(){  
    DibujarOvalo()  
    IrAlArbolIzaquierdo()  
    DibujarArbol()  
    IrAlArbolDerecho()  
    DibujarArbol()  
    VolverAlIncio()  
}
```



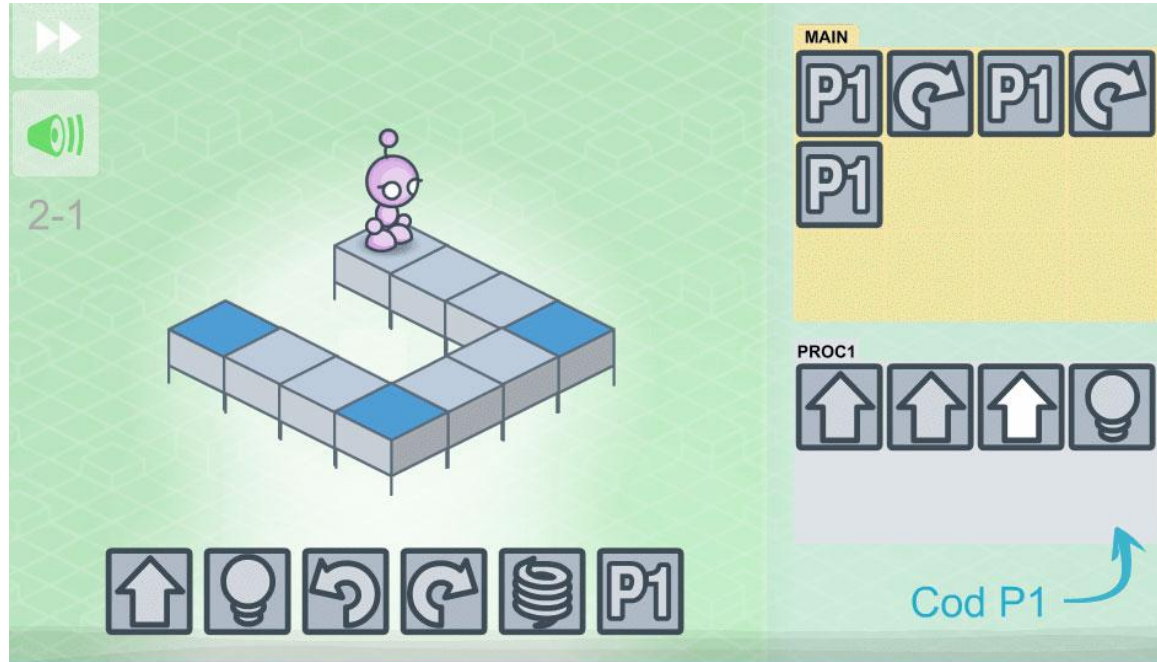
Dibujar el árbol izquierdo y derecho son iguales. Podríamos tener una única tarea que nos dibuje el árbol

Nota: es necesario definir todos los procedimientos, junto con su documentación.

Ejercitamos con Lightbot



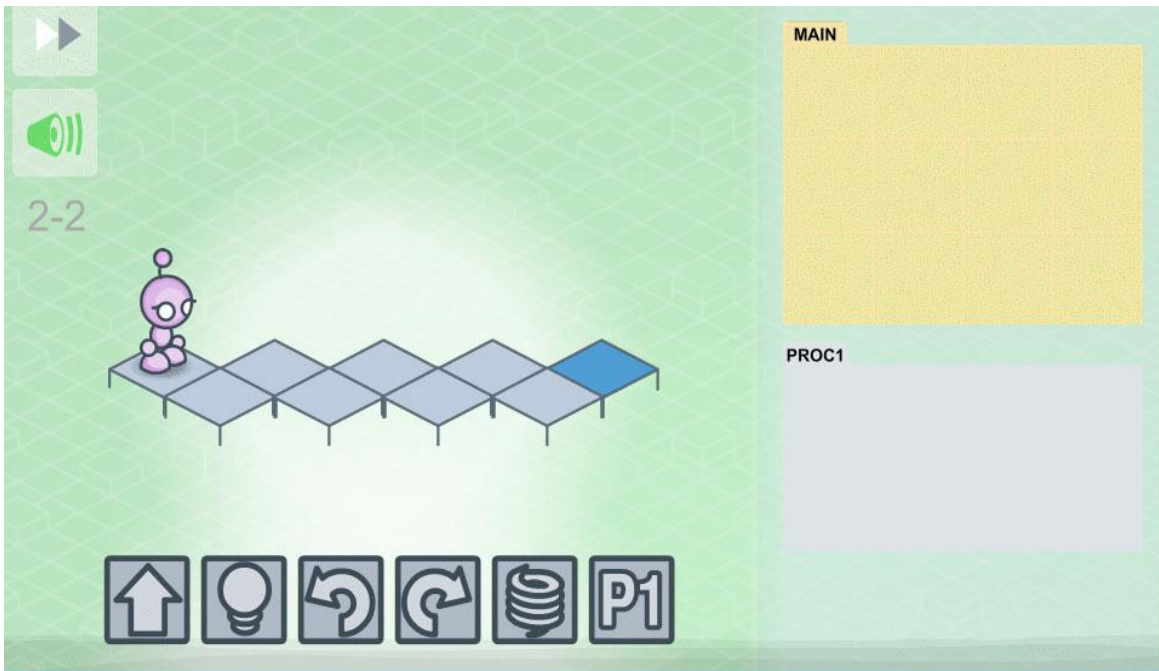
Nivel 1: Ejemplo



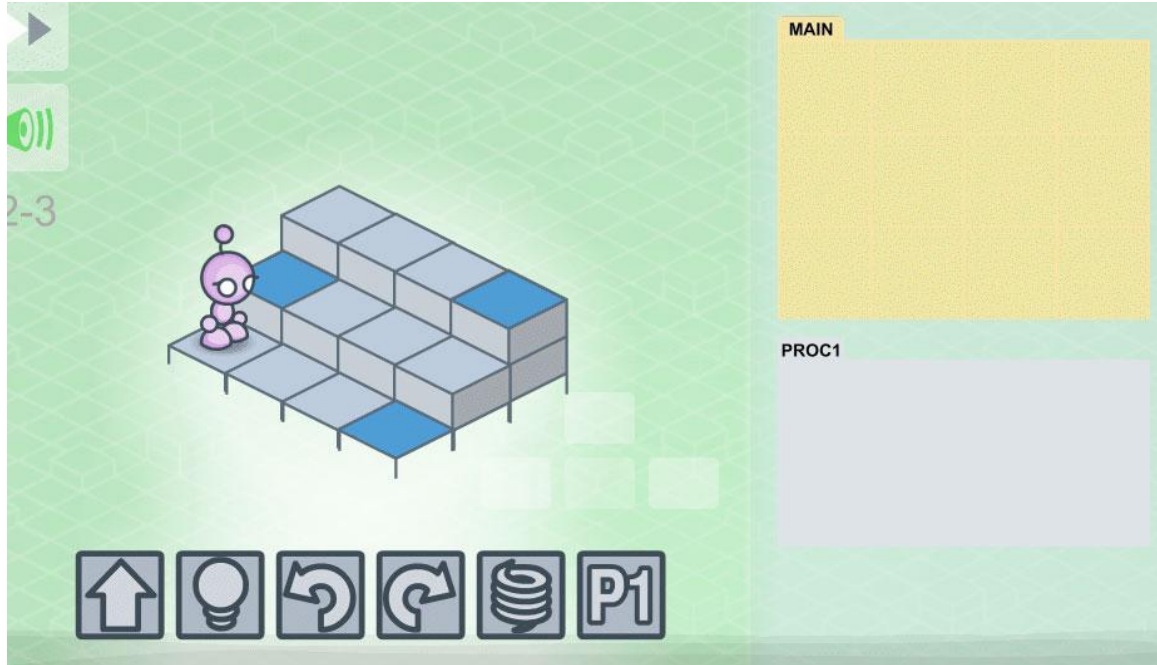
**Llamada al
procedimiento**

**Definición del
procedimiento**

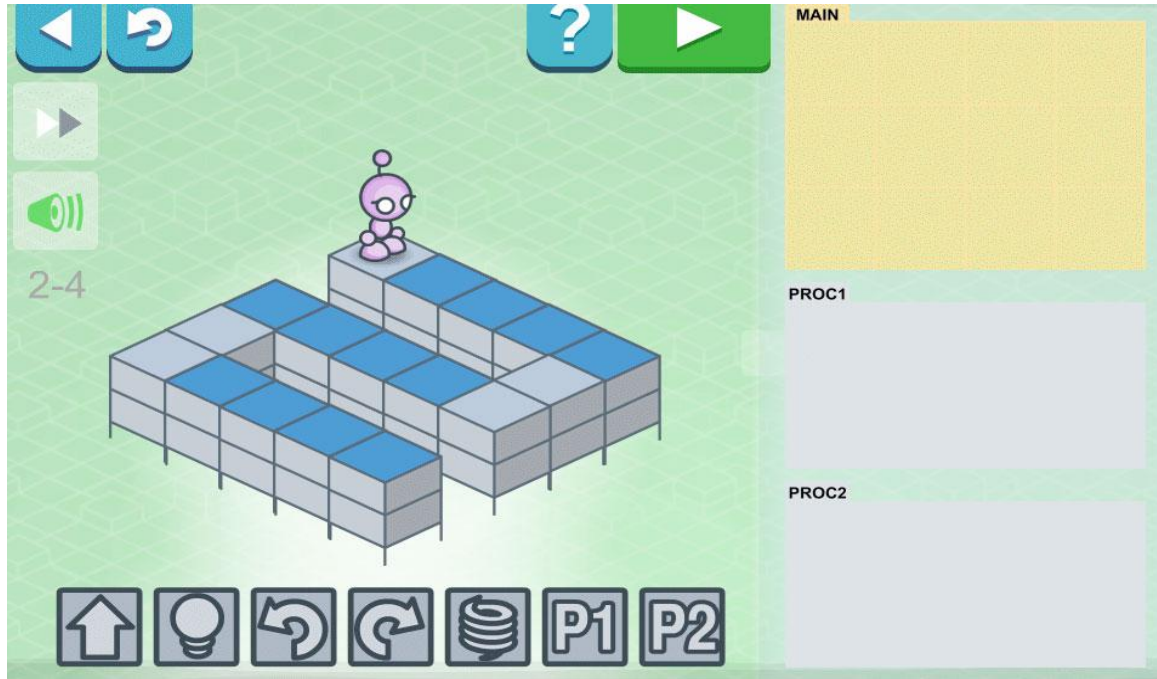
Nivel 2



Nivel 3

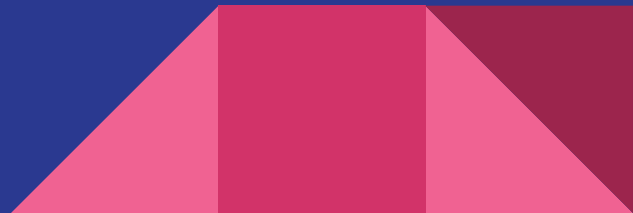


Nivel 4



Para reflexionar...

"Me lo contaron y me lo
olvidé, lo vi y lo entendí,
lo hice y lo aprendí"





Programación

Clase 3

División en subtareas

Universidad Nacional de Quilmes