



# Programación

Clase 3

División en subtareas

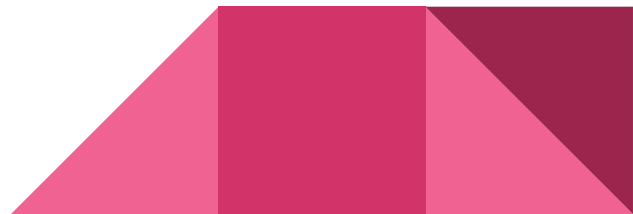
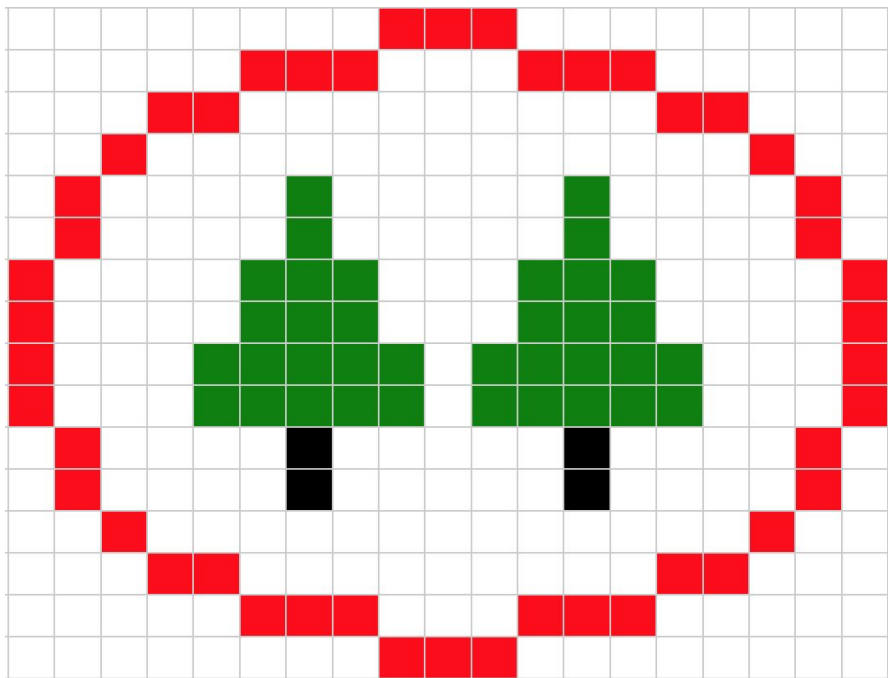
Universidad Nacional de Quilmes

Ejercitamos un poco



# Ejercicio

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:

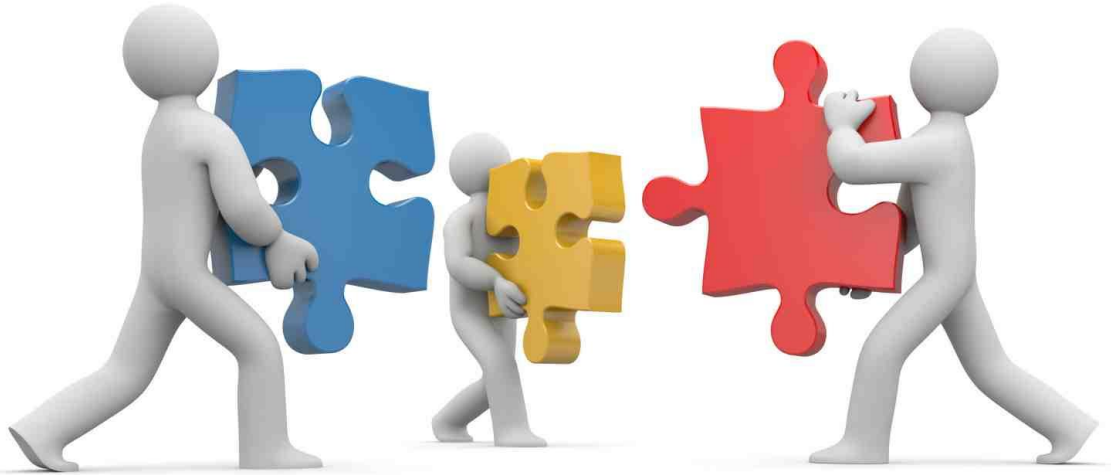
- El código es confuso hasta para nosotros mismos.
  - La otra persona no tiene idea de qué le estamos hablando.
  - Los comentarios pueden ayudar a entender el código, pero no solucionan el problema de fondo.
  - Uno quisiera poder transmitir la idea de una forma más sencilla.
- 

# Lo que quisiéramos transmitir es algo al 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.



# Divide y vencerás. Ejemplo 1

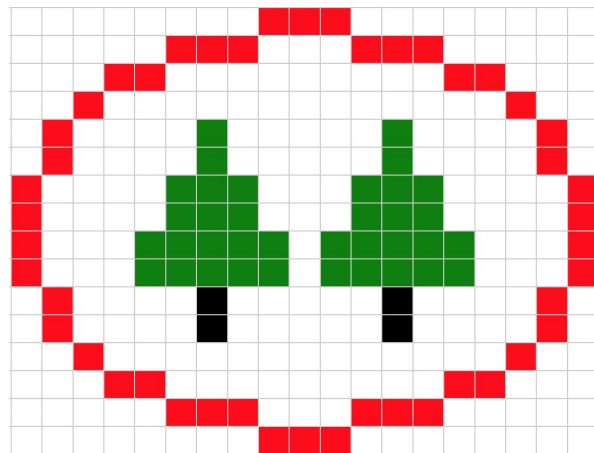
¿Qué pasos o acciones realizan cada vez que realizan la actividad de “cepillarse los dientes”?



# Divide y vencerás. Ejemplo 2

Dibujar logo

- ↳ Dibujar círculo 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

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**.

Un procedimiento se define mediante la palabra “**procedimiento**”, seguida de un **nombre** (El cual no puede contener espacios y comienza con mayúscula), paréntesis vacíos y un **bloque de código**.



# Procedimientos

```
procedimiento DibujarTroncoDeArbol () {
```

```
  /*
```

```
  PROPÓSITO: Dibuja el tronco de un árbol de color negro, de dos  
             celdas de alto, con la celda actual siendo la celda inferior del tronco.
```

```
  */
```

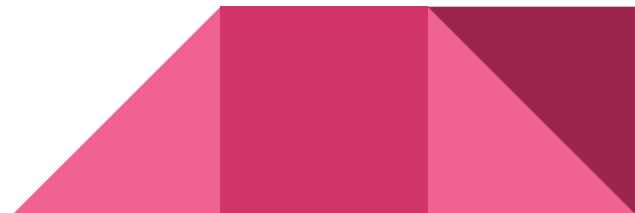
```
  PintarNegro
```

```
  MoverArriba
```

```
  PintarNegro
```

```
  MoverAbajo
```

```
}
```



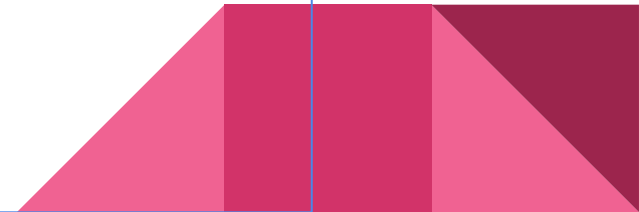
# Ahora tenemos Instrucciones y Procedimientos

Acciones :Set limitado de instrucciones:

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

Procedimientos

- DibujarTroncoDeArbol ()




# Llamar a procedimientos

Los **procedimientos definidos** pueden ser **llamados** luego en cualquier lugar del programa.

Al momento de **ejecutar el programa**, en los lugares en los que se llama a un procedimiento se ejecuta el **bloque de código con el cual fue definido**.

Un procedimiento puede ser **llamado en cualquier bloque de código**, ya sea el del programa o el de otro procedimiento.

Un procedimiento **se define una sola vez**, pero se puede **llamar tantas veces como se desea**.





# Llamar a Procedimientos

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

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



# Llamar a Procedimientos

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

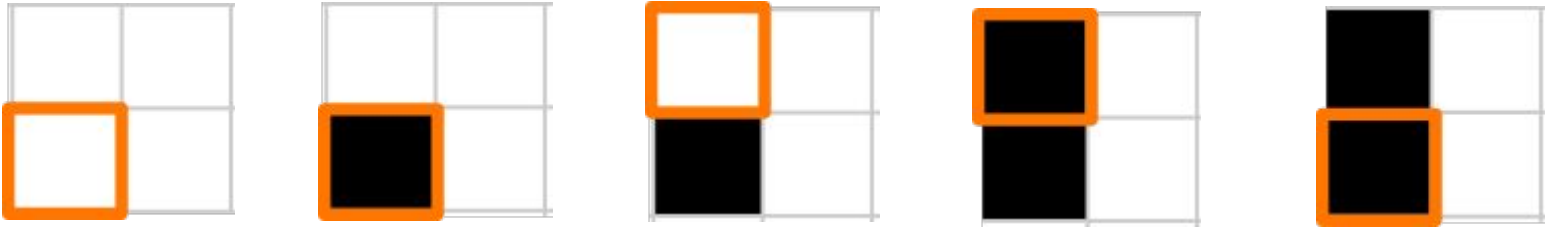
```
programa {  
    DibujarTroncoDeArbol()  
    MoverIzquierda  
    DibujarTroncoDeArbol()  
}
```



# Ejecución del programa anterior

DibujarTroncoDeArbol()

Inicio



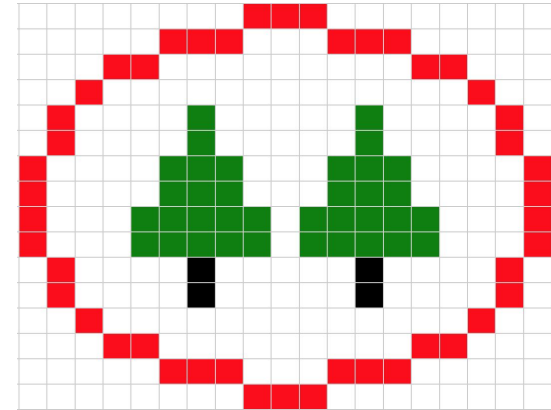
DibujarTroncoDeArbol()

MoverDerecha



# Llamar a Procedimientos en otro Procedimiento

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



**Cada procedimiento debería tener su propósito y comentarios**

# Dibujamos el árbol

```
programa{
```

```
  /* ... */
```

```
  DibujarArbol ()
```

```
}
```

**Mi programa se resume a llamar al procedimiento principal, y solo eso.**



# Ejercitamos un poco con Lightbot



# Nivel 1: Ejemplo

The image shows a game level editor interface. On the left is a 3D scene with a pink robot on a grey and blue block structure. On the right is a code editor with two sections: 'MAIN' and 'PROC1'. The 'MAIN' section contains a sequence of blocks: 'P1', a right-turn arrow, 'P1', and another right-turn arrow. The 'PROC1' section contains four blocks: three up arrows and a lightbulb icon. An orange arrow points from the 'P1' block in the 'MAIN' section to the text 'Llamada al procedimiento'. Another orange arrow points from the lightbulb icon in the 'PROC1' section to the text 'Descripción del procedimiento'. At the bottom of the code editor, there is a label 'Cod P1' with a blue arrow pointing to the right. A toolbar at the bottom left of the editor contains icons for up arrow, lightbulb, left-turn arrow, right-turn arrow, stack, and 'P1'.

2-1

MAIN

P1 ↻ P1 ↻

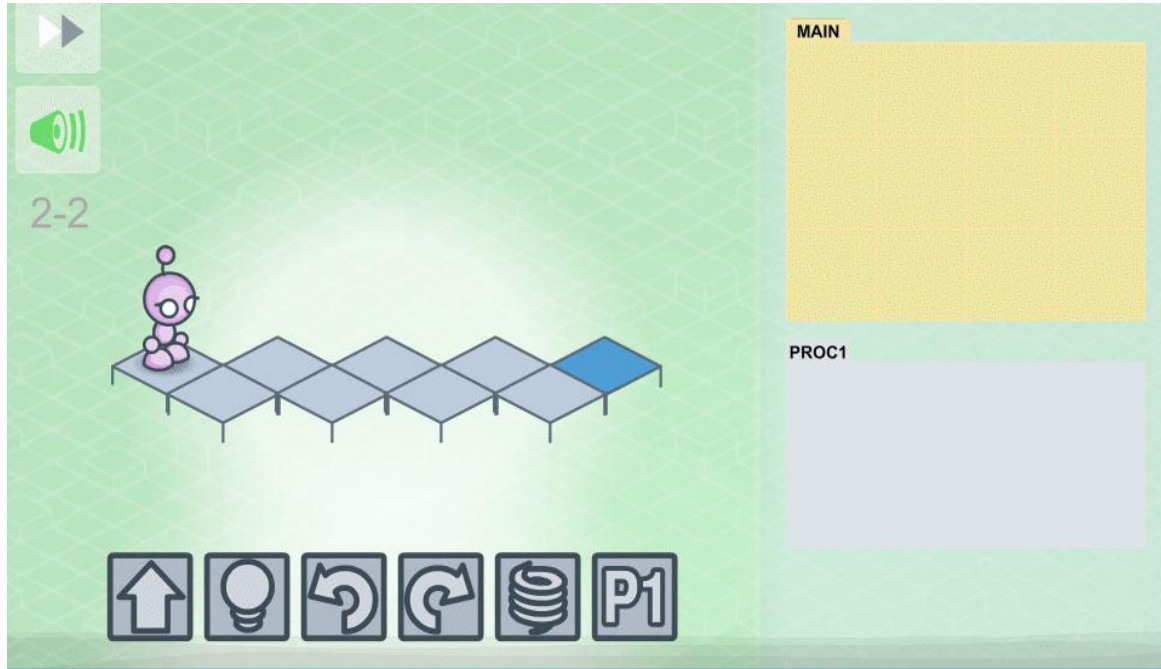
P1 → Llamada al procedimiento

PROC1

↑ ↑ ↑ 💡 → Descripción del procedimiento

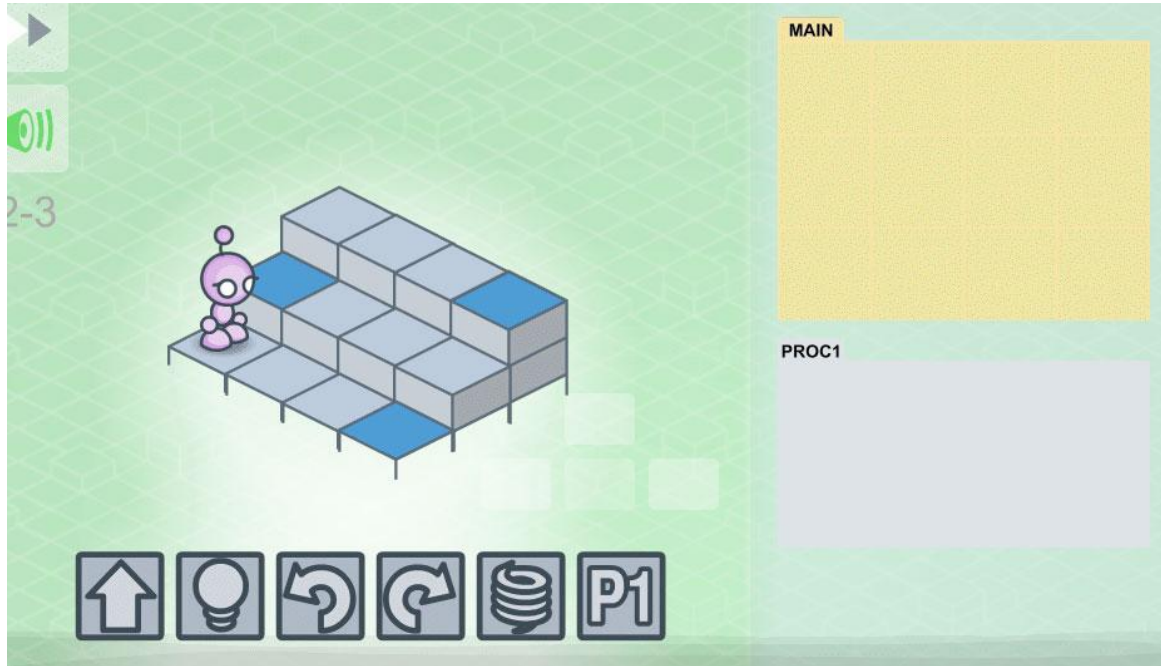
Cod P1 ↻

# Nivel 2





# Nivel 3



# Nivel 4

The image displays a 3D block-based programming environment. The main scene features a pink robot standing on a structure of blue and grey blocks. The interface includes several control elements:

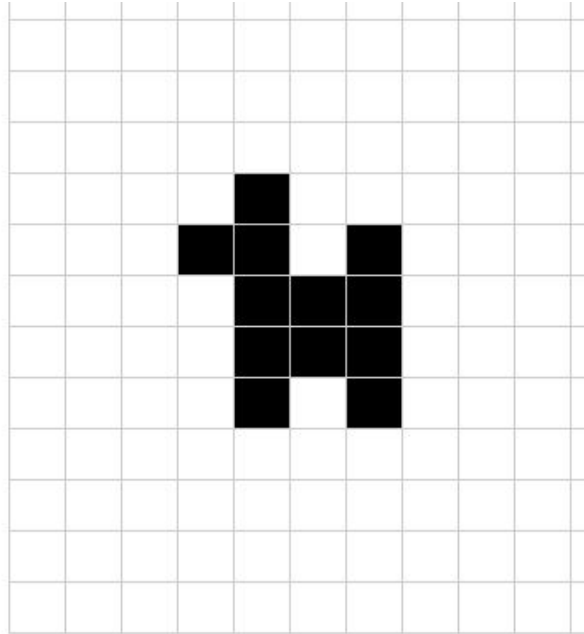
- Top left: Navigation buttons (back, forward, home, play).
- Top right: A question mark icon and a play button.
- Left side: A play button, a volume icon, and the text "2-4".
- Bottom: A toolbar with icons for up, lightbulb, undo, redo, stack, and variables P1 and P2.
- Right side: Three empty code blocks labeled MAIN, PROC1, and PROC2.

# Ejercitamos un poco con QDraw



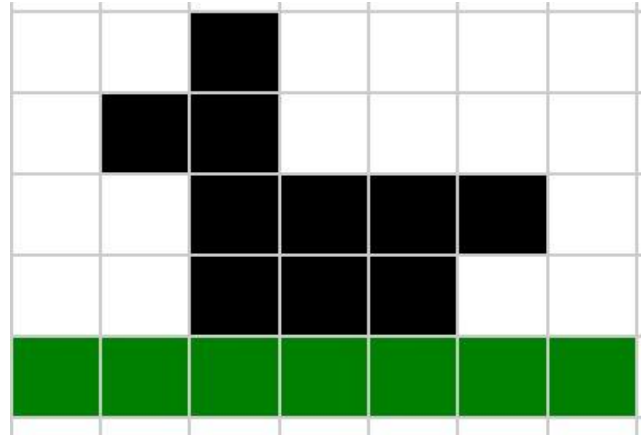
# Actividad 1

Implemente un programa que realice el siguiente dibujo utilizando procedimientos.



## Actividad 2

Implemente un programa que realice el siguiente dibujo utilizando procedimientos.



¿Encuentran algo parecido al dibujo anterior?



# Programación

Clase 3

División en subtareas

Universidad Nacional de Quilmes