



Módulo de Computación

Algoritmos y Programas

Ingreso 2024
Segundo Encuentro

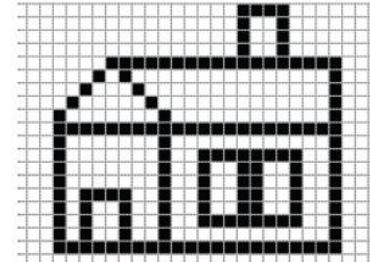
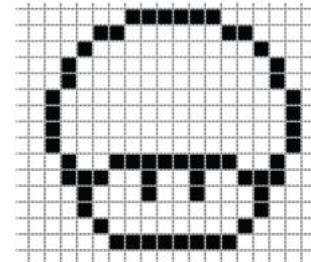
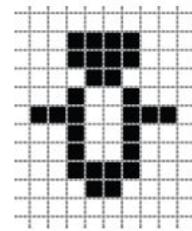
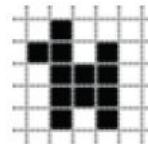


Departamento de Computación
Facultad de Ciencias Exactas Físico-Químicas y Naturales
Universidad Nacional de Río Cuarto



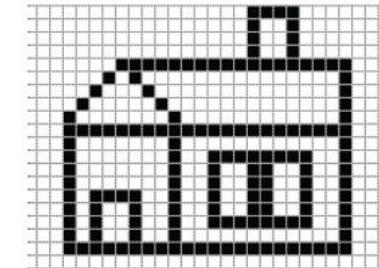
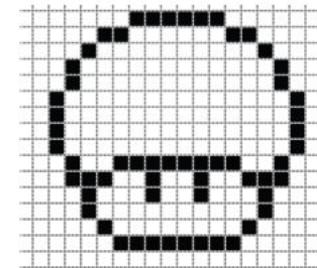
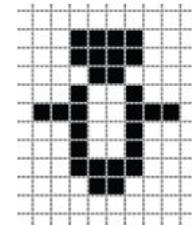
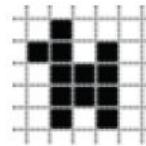
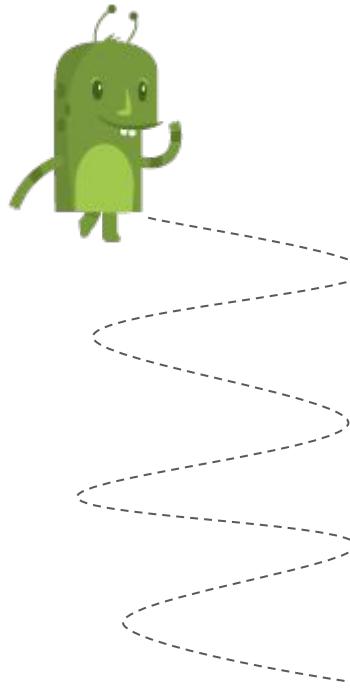
Programar en Papel

Cuadriculado



Actividad

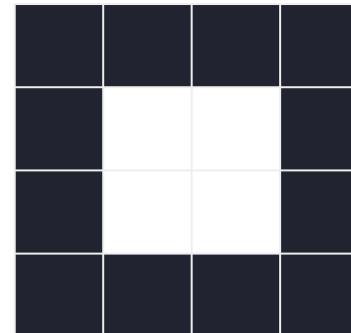
Todo al pie de la letra



Actividad

Programamos en papel

Escribir las instrucciones para
pintar un **cuadrado**



Actividad Todo al pie de la letra

Para reflexionar

¿Qué sucede si una instrucción no es válida?

¿Qué sucede si la instrucción no puede realizarse?

¿Cualquier orden de las instrucciones da lo mismo?



Actividad Todo al pie de la letra

¿Qué elementos encontramos hasta ahora?



Actividad Todo al pie de la letra

¿Qué elementos encontramos hasta ahora?

- Un **problema** que requiere solución
 - No es cualquier problema
 - Hay un **estado inicial** específico
- Un **autómata** (máquina)
 - realiza acciones a partir de instrucciones **primitivas**
 - no entiende nada más que esas instrucciones



Actividad Todo al pie de la letra

¿Cómo solucionamos el problema?

- Desarrollamos un **programa** para que la máquina lo **ejecute**
- Así, es la máquina la que soluciona en forma **autónoma** el problema con nuestras instrucciones
 - No hay (en principio) intervención humana durante la ejecución



Actividad Todo al pie de la letra

¿Qué elementos precisamos para obtener la solución?

- Un **programa** que:
 - ... al ser *ejecutado* por la máquina cumpla el propósito
 - ... *describa* la solución para una persona (expresando de alguna forma la estrategia)

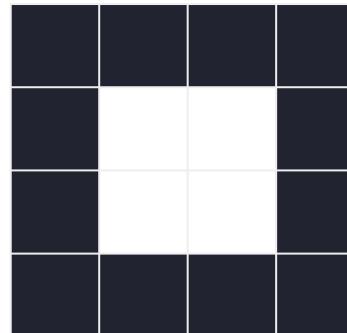


Actividad

Retomemos

Programamos en papel 1

Escribir un programa que
pinte un **cuadrado**



Actividad

Programamos en papel

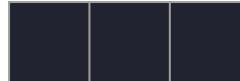
Instrucciones	
←	Mover el lápiz al casillero de la izquierda
→	Mover el lápiz al casillero de la derecha
↑	Mover el lápiz al casillero de arriba
↓	Mover el lápiz al casillero de abajo
[■]	Pintar el casillero

Para **ejecutar** un programa, usaremos hojas de papel cuadriculado

Para **crear** un programa, escribiremos cada secuencia de instrucciones de izquierda a derecha

Ejemplo: [■] → [■] → [■]

Resultado:



Actividad Programamos en papel 1

Una posible solución:



Actividad Programamos en papel 1

Una posible solución:



Otra posible solución:



Actividad Todo al pie de la letra

¿Qué elementos precisamos para obtener la solución?

- Un **programa**
 - Que *describa* la solución para una persona (expresando de alguna forma la **estrategia**)
 - Que sirva para que la ejecución de la máquina cumpla el propósito





Programa

Descripción de una solución a un problema, **ejecutable** por algún autómata (máquina)



Cierre Todo al pie de la letra

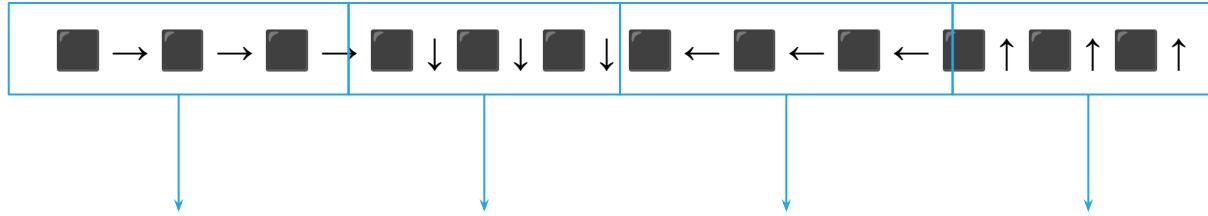
¿Qué conceptos fundamentales trabajamos en esta actividad?

- **InSTRUCCIONES** a las que responde el autómata o máquina que ejecuta nuestros programas.
- **Estrategia** Manera en la que vamos a resolver el problema, como una serie de ***pASOS O PROBLEMAS MÁS PEQUEÑOS***.
- **Programa Descripción** de una solución a un problema, ejecutable por algún autómata (máquina)



Actividad Programamos en papel 1

Una posible solución:

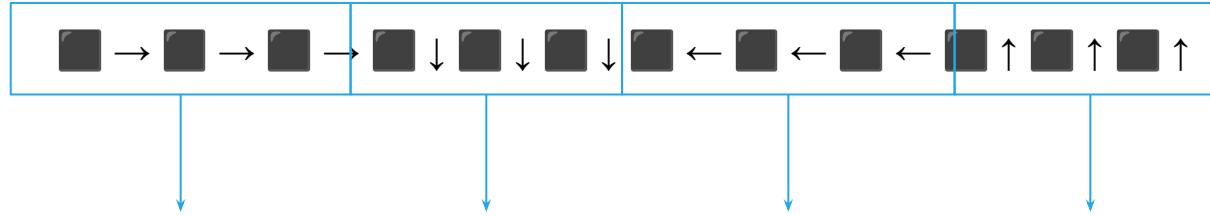


Podemos ver que cada una de estas partes es el resultado de copiar **3 veces** las primitivas para pintar y mover.



Actividad Programamos en papel 1

Una posible solución:



Podemos ver que cada una de estas partes es el resultado de copiar **3 veces** las primitivas para pintar y mover.

¿Podremos expresar **repeticiones** ?

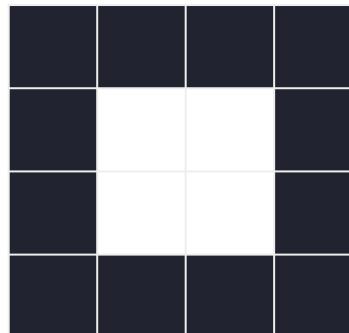


Actividad

Programamos en papel 1

Reescribir el programa anterior
con **repeticiones**

Para expresar repeticiones utilizamos
paréntesis y números: ( →) 3



Actividad Programamos en papel 1

Solución con
repeticiones

vs.

Solución sin
repeticiones

(→)3 (↓)3 (←)3 (↑)3

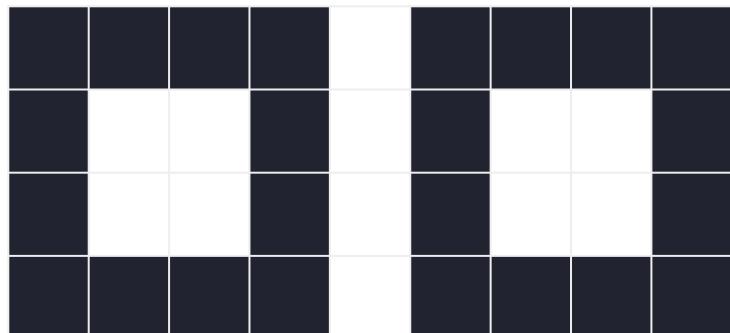
→ → → ↓ ↓ ↓ ← ← ← ↑ ↑ ↑



Actividad

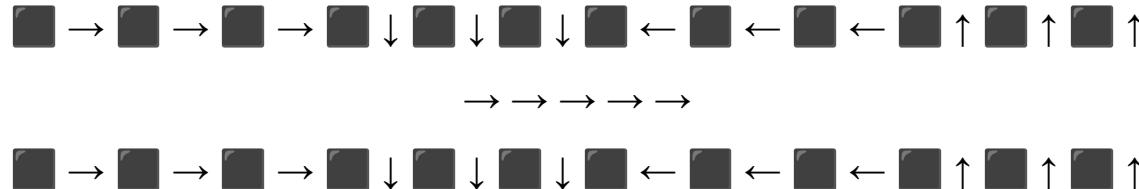
Programamos en papel 1

Escribir un programa que
pinte dos cuadrados



Actividad Programamos en papel 1

Una posible solución:



Otra posible solución:

$(\square \rightarrow)3 (\square \downarrow)3 (\square \leftarrow)3 (\square \uparrow)3 (\rightarrow)5 (\square \rightarrow)3 (\square \downarrow)3 (\square \leftarrow)3 (\square \uparrow)3$



Actividad Programamos en papel 1

Definición de
funciones/procedimientos

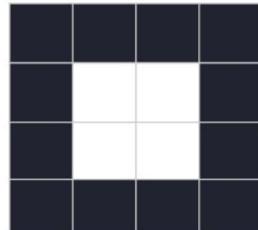
“Dibujar Un Cuadrado”

(→) 3

(↓) 3

(←) 3

(↑) 3

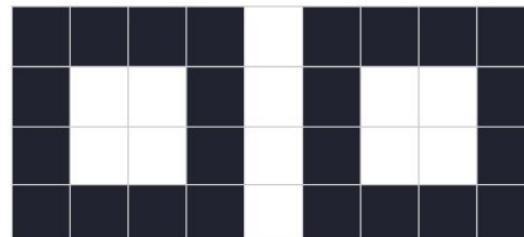


Llamado de
funciones/procedimientos

[DibujarUnCuadrado]

(→) 5

[DibujarUnCuadrado]



Actividad Programamos en papel 1

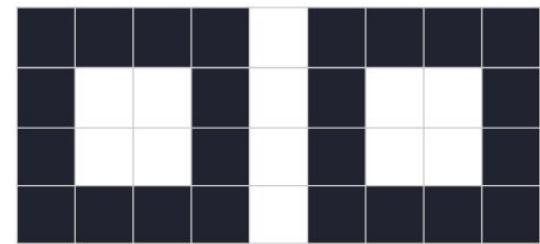
Otra otra posible solución usando definición de [funciones/procedimientos](#)

“Dibujar Un Cuadrado”

- (→) 3
- (↓) 3
- (←) 3
- (↑) 3

Programa principal:

```
[DibujarUnCuadrado]  
    (→) 5  
    [DibujarUnCuadrado]
```



Actividad Programamos en papel 1

“Dibujar Un Cuadrado”

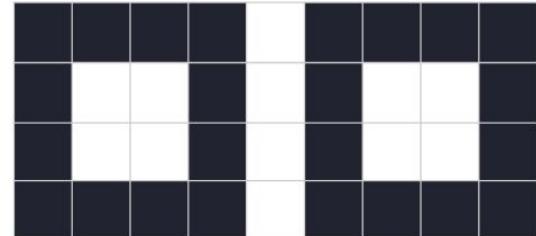
(→) 3
(↓) 3
(←) 3
(↑) 3

Programa principal:

[DibujarUnCuadrado]
→ 5
[DibujarUnCuadrado]

VS.

(→)3 (↓)3 (←)3 (↑)3 → 5 (→)3 (↓)3 (←)3 (↑)3



Cierre Programamos en papel 1

¿Por qué usamos **actividades desenchufadas** (sin computadora)?

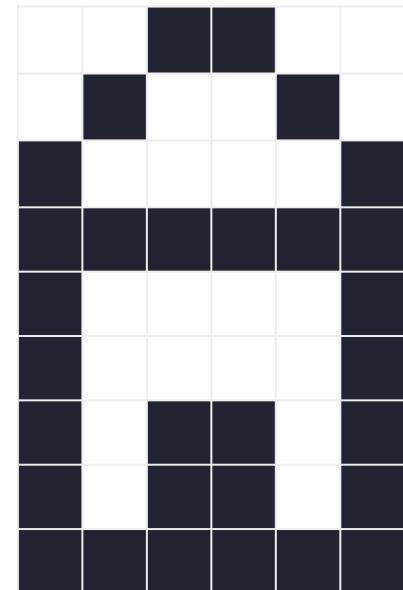
- Los conceptos son **independientes** de las herramientas
- Copiar+pegar y prueba y error tienen una **dificultad agregada** cuando se trabaja en papel
- La **necesidad** de expresar repeticiones y procedimientos aparece más claramente



Actividad

Programamos en papel 2

Escribir un programa que
dibuje una **casa**



Actividad Programamos en papel 2

¿Es fácil saber cuál es el dibujo **sin ejecutar el programa?**

¿Es fácil ejecutar el programa **sin cometer errores?**

Si la respuesta a alguna de esas preguntas es No:

¿qué podemos hacer para que sea fácil?



Actividad

Programamos en papel 2

¿Qué dibujo realiza el siguiente programa?

```
(█↑)7 ←←(↓)3 █↓█→█↓█→→↑  
█↑█→█↑█↑↑← (█←)3 (█↑)3 (█→)4 (↓█)2  
(↑)3 (█←←)2 █
```



Actividad

Programamos en papel 2

¿Y qué dibujo realiza este otro programa?

[Dibujar tallo]

$\leftarrow\leftarrow(\downarrow)3$

[Dibujar hoja izquierda]

$\rightarrow\rightarrow\uparrow$

[Dibujar hoja derecha]

$\uparrow\uparrow\leftarrow$

[Dibujar pétalos]

“Dibujar tallo”

$(\blacksquare\uparrow)7$

“Dibujar pétalos”

$(\blacksquare\leftarrow)3(\blacksquare\uparrow)3(\blacksquare\rightarrow)4$

$(\downarrow\blacksquare)2$

$(\uparrow)3(\blacksquare\leftarrow\leftarrow)2\blacksquare$

“Dibujar hoja izquierda”

$\blacksquare\downarrow\blacksquare\rightarrow\blacksquare\downarrow\blacksquare$

“Dibujar hoja derecha”

$\blacksquare\uparrow\blacksquare\rightarrow\blacksquare\uparrow\blacksquare$



Conclusión

- Ambos tienen las **mismas instrucciones...**
- Se observa la importancia de la **legibilidad**

[Dibujar tallo]

$\leftarrow\leftarrow(\downarrow)3$

[Dibujar hoja izquierda]
 $\rightarrow\rightarrow\uparrow$

[Dibujar hoja derecha]
 $\uparrow\uparrow\leftarrow$



[Dibujar pétalos]

“Dibujar tallo”

$(\blacksquare\uparrow)7$

“Dibujar pétalos”

$(\blacksquare\leftarrow)3(\blacksquare\uparrow)3(\blacksquare\rightarrow)4$

$(\downarrow\blacksquare)2$

$(\uparrow\blacksquare)3(\blacksquare\leftarrow\leftarrow)2\blacksquare$

“Dibujar hoja izquierda”
 $\blacksquare\downarrow\blacksquare\rightarrow\blacksquare\downarrow\blacksquare$

“Dibujar hoja derecha”
 $\blacksquare\uparrow\blacksquare\rightarrow\blacksquare\uparrow\blacksquare$

VS

$(\blacksquare\uparrow)7\leftarrow\leftarrow(\downarrow)3\blacksquare\downarrow\blacksquare\rightarrow\blacksquare\downarrow\blacksquare\rightarrow\rightarrow\uparrow$
 $\blacksquare\uparrow\blacksquare\rightarrow\blacksquare\uparrow\blacksquare\uparrow\leftarrow(\blacksquare\leftarrow)3(\blacksquare\uparrow)3$
 $(\blacksquare\rightarrow)4(\downarrow\blacksquare)2$
 $(\uparrow)3(\blacksquare\leftarrow\leftarrow)2\blacksquare$

Actividad Programamos en papel 2

- ¿Se dieron cuenta que ambos programas tienen las **mismas instrucciones**?
- ¿En cuál de los dos programas fue más fácil saber qué se dibujaba? ¿Por qué?
- ¿En cuál de los dos programas sería más fácil corregir un error? ¿Por qué?
- ¿En cuál de los dos programas sería más fácil introducir una modificación? ¿Por qué?



Cierre Programamos en papel 2

Legibilidad: Facilidad con la que una **persona** puede interpretar un programa para comprenderlo y entender cómo está estructurado

- Expresada en la creación de **procedimientos** y la **elección de sus nombres**, y **repeticiones**
- Hace que los programas sean más fáciles de ...
 - **compartir** (facilidad de interpretación)
 - **corregir** (facilidad para encontrar los errores)
 - **modificar/adaptar** (facilidad para identificar las partes dónde se deben realizar los cambios)

[Dibujar tallo]

←←(↓)3

[Dibujar hoja izquierda]

→→↑

[Dibujar hoja derecha]

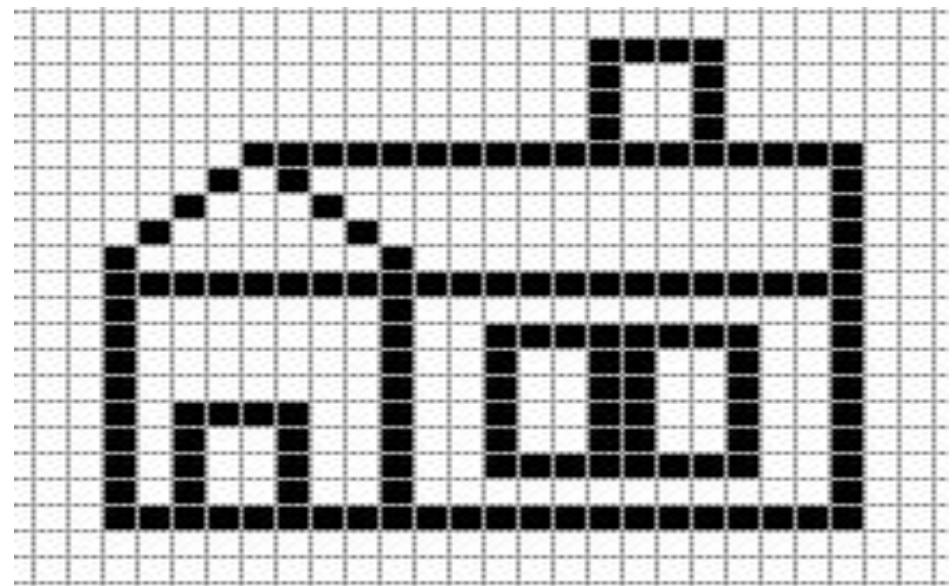
↑↑←

[Dibujar pétalos]

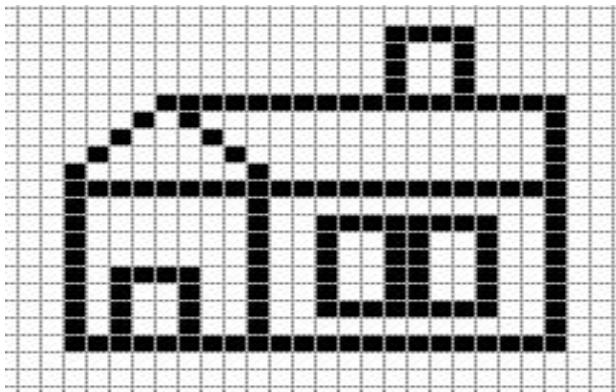


Actividad Programar en papel cuadriculado (opcional)

Escribir un programa que
dibuje la casa



Actividad Dibujar una casa



- Tratá de **descomponer el problema** en subproblemas más sencillos
- De esta forma, el cuerpo del programa principal se parece a una **descripción** de cómo dibujar una casa



Actividad Programar en papel cuadriculado

“Dibujar casa”

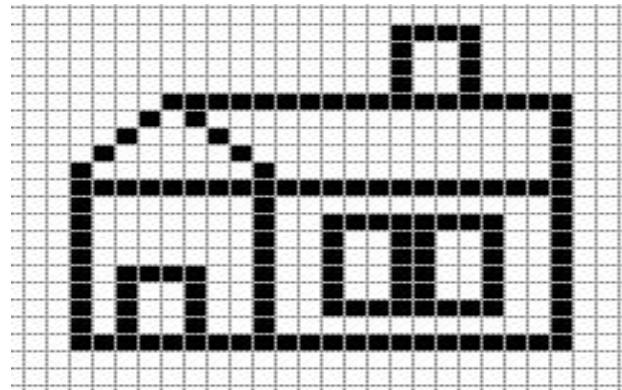
```
[Dibujar frente de la casa]  
[Ir a la posición del fondo]  
[Dibujar fondo de la casa]  
[Ir a la posición del techo del frente]  
[Dibujar techo del frente]  
[Ir a posición del techo del fondo]  
[Dibujar techo del fondo]
```

“Dibujar frente de la casa”

```
[Dibujar pared del frente]  
[Ir a posición de la puerta]  
[Dibujar puerta]
```

“Dibujar fondo de la casa”

```
[Dibujar pared del fondo]  
[Ir a posición de las ventanas]  
[Dibujar ventanal]
```



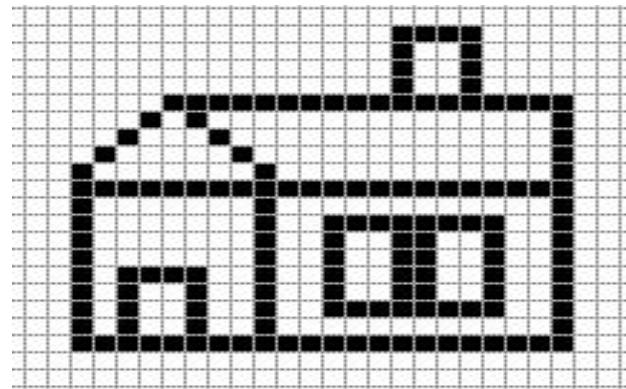
Importante

- ¡No usamos aún comandos primitivos!
- ¿Qué **estrategia** utilizamos aquí?
- ¿Cuál usarían ustedes?



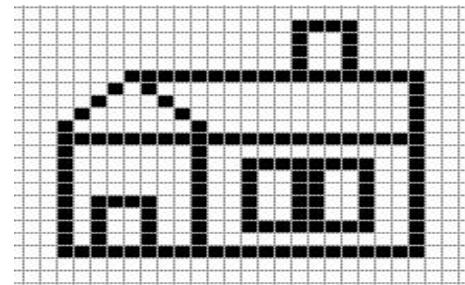
Conclusión Programar en papel cuadriculado

- ¿Cómo escribimos el programa para que pueda ser reproducido con el mínimo de errores posibles?
- ¿Definimos la estrategia?
- ¿Utilizamos procedimientos?



Conclusión Programar en papel cuadriculado

- Visión de programa como **ejecución** vs.
Visión de programa como **descripción**
- En el marco de este curso, todo programa tiene un
propósito
- **Estrategia** expresada por el programa
- **Procedimientos** como herramienta fundamental
para expresar la estrategia.



Parametrización

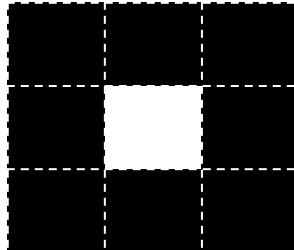


Actividad Programar en papel cuadriculado

Definición y uso de procedimientos

“Dibujar cuadrado”

(↗) 2
(↓) 2
(↖) 2
(↑) 2

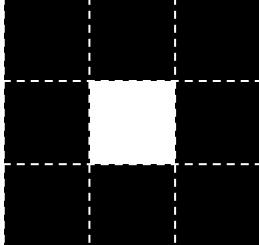


Actividad Programar en papel cuadriculado

Definición y uso de procedimientos

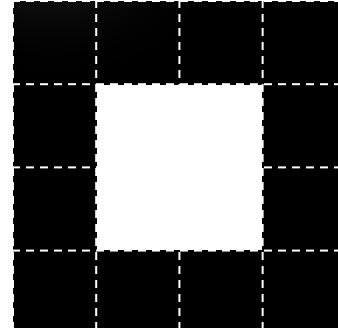
“Dibujar cuadrado tres”

(↗) 2
(↙) 2
(↖) 2
(↘) 2



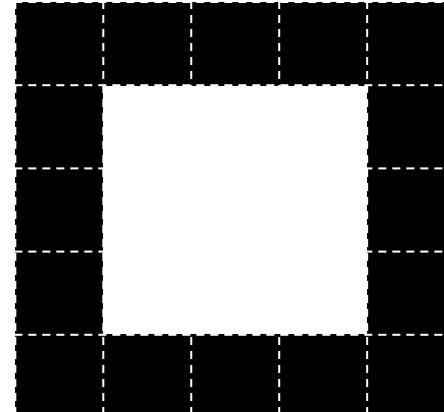
“Dibujar cuadrado cuatro”

(↗) 3
(↙) 3
(↖) 3
(↘) 3



“Dibujar cuadrado cinco”

(↗) 4
(↙) 4
(↖) 4
(↘) 4

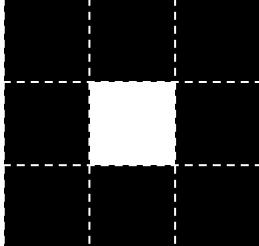


Actividad Programar en papel cuadriculado

Definición y uso de procedimientos

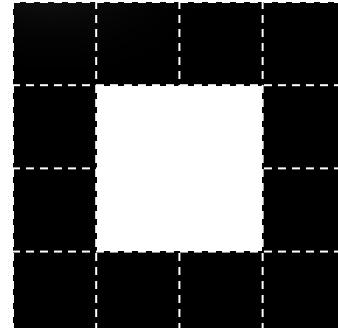
“Dibujar cuadrado tres”

(↗→) 2
(↘↓) 2
(↙←) 2
(↗↑) 2



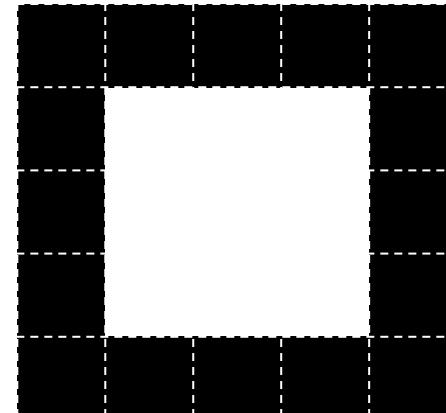
“Dibujar cuadrado cuatro”

(↗→) 3
(↘↓) 3
(↙←) 3
(↗↑) 3



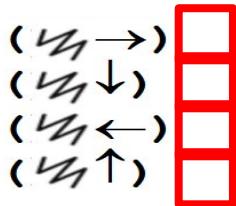
“Dibujar cuadrado cinco”

(↗→) 4
(↘↓) 4
(↙←) 4
(↗↑) 4

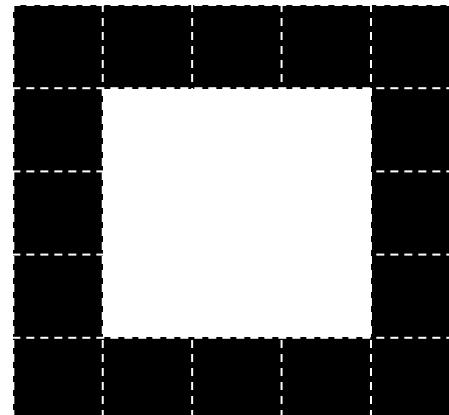
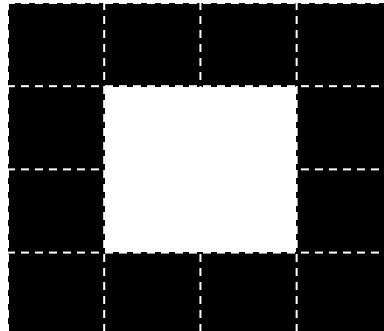
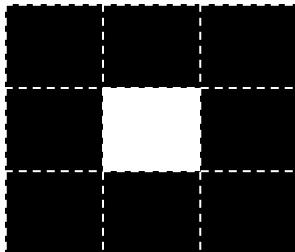


Actividad Programar en papel cuadriculado

“Dibujar cuadrado”



Pero aún necesitamos expresar el dato que completa el cuadrado

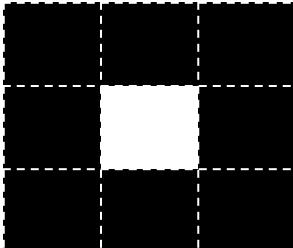


Actividad Programar en papel cuadriculado

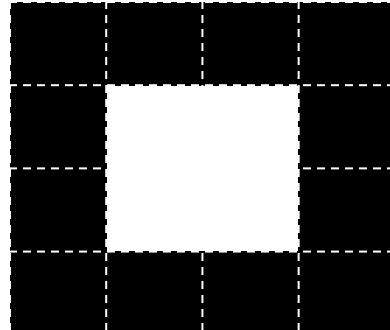
“Dibujar cuadrado (nro_cuadraditos)”

(→) nro_cuadraditos
(↓) nro_cuadraditos
(←) nro_cuadraditos
(↑) nro_cuadraditos

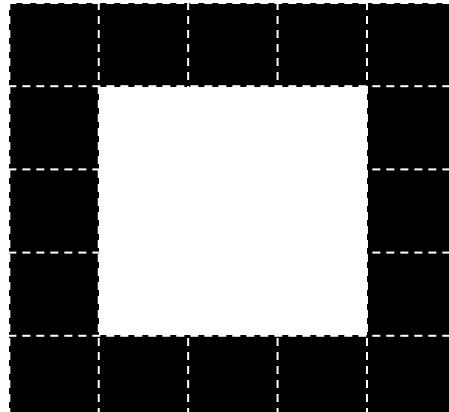
“Dibujar cuadrado (2)”



“Dibujar cuadrado (3)”



“Dibujar cuadrado (4)”



Actividad Programar en papel cuadriculado

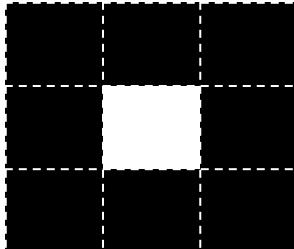
“Dibujar cuadrado (nro_cuadraditos)”

(→) nro_cuadraditos
(↓) nro_cuadraditos
(←) nro_cuadraditos
(↑) nro_cuadraditos

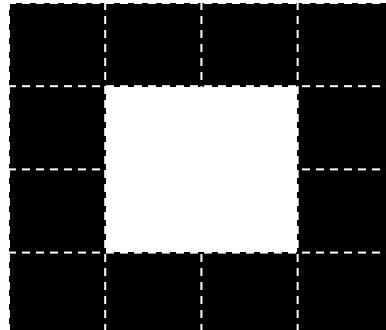
Definición de la función

Usos/invocaciones de la función

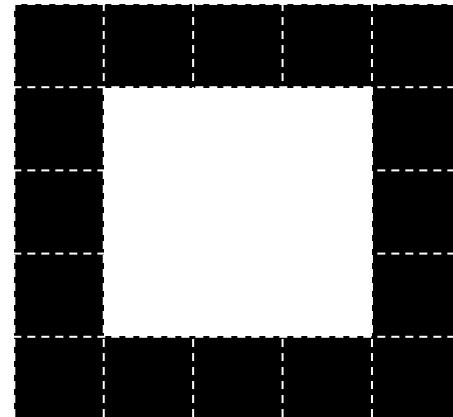
“Dibujar cuadrado (2)”



“Dibujar cuadrado (3)”

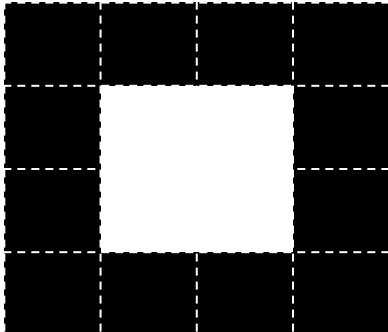


“Dibujar cuadrado (4)”



Actividad Programar en papel cuadriculado

“Dibujar cuadrado (3)”



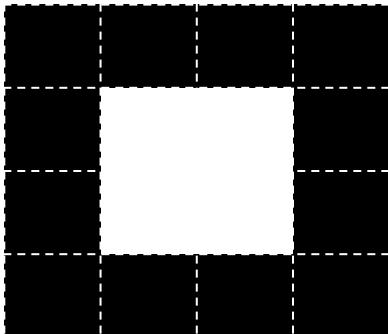
“Dibujar cuadrado (nro_cuadraditos)”

(↗) nro_cuadraditos
(↓) nro_cuadraditos
(←) nro_cuadraditos
(↑) nro_cuadraditos



Actividad Programar en papel cuadriculado

“Dibujar cuadrado (3)”



“Dibujar cuadrado (nro_cuadraditos)”

(↗) nro_cuadraditos
(↘) nro_cuadraditos
(↖) nro_cuadraditos
(↙) nro_cuadraditos

Entonces nro_cuadraditos = 3

(↗) nro_cuadraditos <- 3
(↘) nro_cuadraditos <- 3
(↖) nro_cuadraditos <- 3
(↙) nro_cuadraditos <- 3



Conclusión Parámetros

- Los **parámetros** permiten que los procedimientos sean más generales (es decir, resuelvan más problemas).
- Sirven para **no tener que escribir muchos (¡infinitos!) procedimientos similares**.
- Sirven para **resolver problemas conocidos con datos desconocidos** (sabemos que tenemos que hacer un cuadrado, pero no sabemos de qué tamaño).

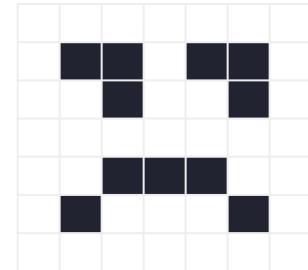
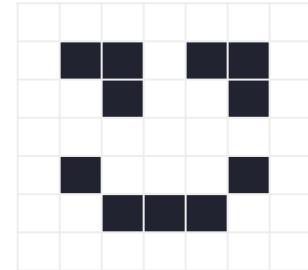
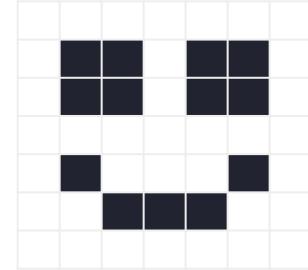


Actividad

Programamos en papel 3

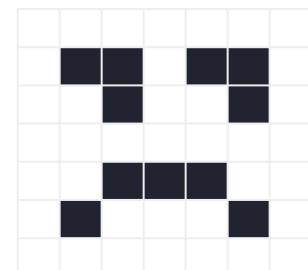
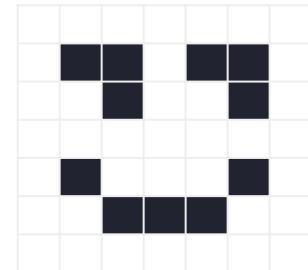
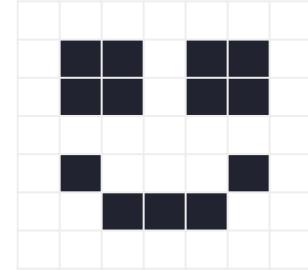
Escribir programas que dibujen cada una de las tres **caritas**

Importante: Pueden utilizar el mismo procedimiento en diferentes programas



Actividad Programamos en papel 3

- ¿Hay algún subproblema que se repita en más de un programa?
- ¿Hay algún subproblema que se repita dentro del mismo programa?
- Si quisiéramos dibujar alguna de las caritas pero con sombrero, ¿qué podríamos aprovechar de lo ya programado para que sea fácil realizar el nuevo dibujo?



Cierre Programamos en papel 3

- En vez de copiar y pegar un fragmento de programa que queremos reutilizar, es conveniente definir un procedimiento que lo contenga y llamar al procedimiento las veces que sea necesario
- Valoramos la reutilización de procedimientos porque hace que los programas sean más fáciles de ...
 - interpretar (nos damos cuenta que son las mismas instrucciones que ya leímos en otro lado del programa)
 - corregir/modificar (los cambios hay que hacerlos en un único lugar)

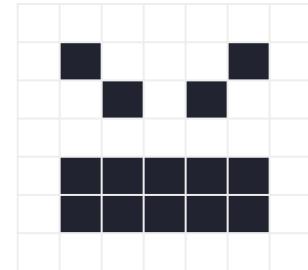
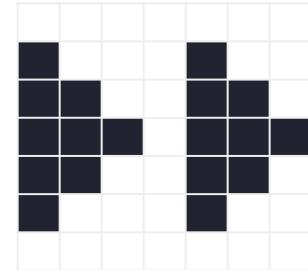
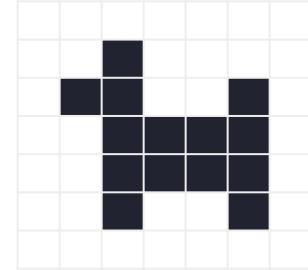


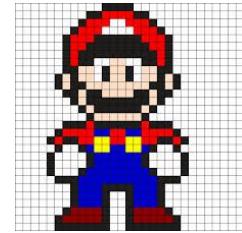
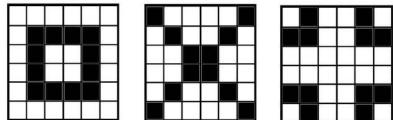
Actividad

Programamos en papel 4

1. En grupos pequeños, diseñen un dibujo cuadriculado y escriban un programa que lo dibuje
2. Intercambien programas con otro grupo y “ejecútenlos” a ver si logran reproducir el mismo dibujo

Importante: ¡El objetivo es que el dibujo **se pueda reproducir sin inconvenientes!**



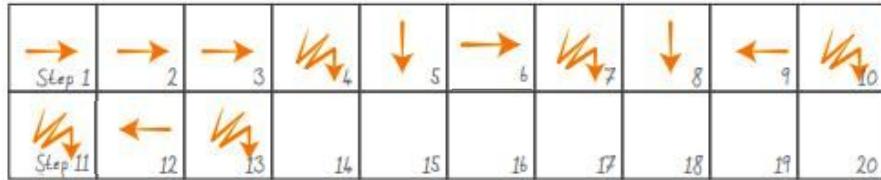
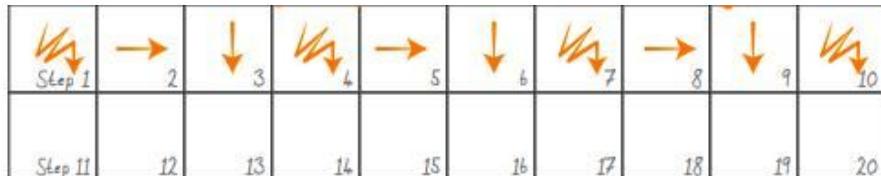
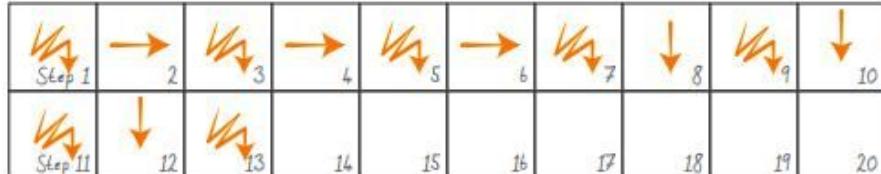
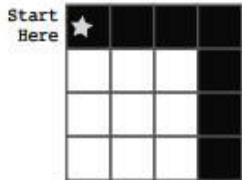
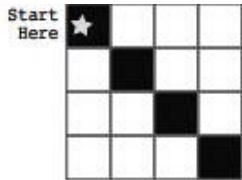


Actividad: Dibujando en una cuadricula

Ejercicio:

¿Qué imagen se corresponde con cada secuencia de instrucciones?

En caso de que quede algún par sin asociar, modifique la secuencia de instrucciones para que se correspondan



Reflexionamos...

¿Qué errores tuvieron a la hora de encontrar las soluciones?

¿Se repitieron esos errores?

¿Cómo detectaron esos errores? ¿Cómo encontraron en qué lugar se produce el error?

¿Consideran que utilizó algún método sistemático para corroborar que su solución era correcta?



Prueba y Depuración

Prueba (validación) de las soluciones: revisión de nuestras soluciones para corroborar que efectivamente realizan lo esperado.

Responde a la pregunta ¿es una solución al problema?

Depuración (debugging) de las soluciones: proceso de identificar y corregir errores.

Son actividades muy importantes a la hora de aprender a programar.



Algoritmos

En las Ciencias de la Computación, y en la programación, los algoritmos son más importantes que los lenguajes de programación o las computadoras.

Un lenguaje de programación es sólo un medio para expresar un algoritmo y una computadora es sólo un procesador para ejecutarlo



Algoritmos

Algoritmo: secuencia de instrucciones que realizadas en orden conducen a obtener la solución de un problema.

Programa: secuencia de instrucciones, escritas para realizar una tarea específica en una computadora.

Lenguaje de Programación: lenguaje formal que especifica una serie de instrucciones para que una computadora las ejecute

Los lenguajes de programación pueden usarse para crear programas que pongan en práctica algoritmos

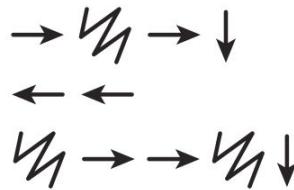


Lenguajes de Programación

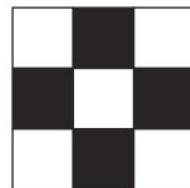
- ❖ Las instrucciones representan acciones y, si las ejecutamos, la computadora realiza dichas acciones.
- ❖ Las secuencias de instrucciones son series de acciones que se realizan una a continuación de la otra, en un orden determinado.
- ❖ Un **programa** es una descripción que le damos a la computadora para que realice lo que le indicamos.
- ❖ Las **instrucciones** incluidas originalmente en el entorno de programación se llaman **primitivas**.
- ❖ Se pueden definir **nuevos comandos**, que llamamos **procedimientos**, para explicarle a la computadora cómo realizar nuevas acciones.



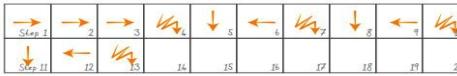
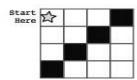
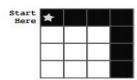
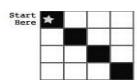
Introducción a los Algoritmos



Dar secuencias de instrucciones



“Ejecutar” secuencia de instrucciones



Validar/Verificar secuencia de instrucciones

Corregir secuencia de instrucciones incorrectas (bugs)



Metodología de resolución de ejercicios

1. Identificar la estrategia de resolución
2. Identificar las partes que componen la estrategia
3. Implementar la estrategia general y las partes que la componen



Programación y Ciencias de la Computación

Encarar un problema a resolver puede ser complicado, pero con estas simples herramientas, nada es imposible:

Paso 1) Descomponer — partir un gran problema en algo mucho más simple. Muchas veces, los grandes problemas consisten de muchos pequeños problemas, todos juntos en el problema mayor.

Paso 2) Patrones — A veces, cuando un problema tiene muchas pequeñas partes, notarás que esas partes tienen algo en común. Si no lo tienen, entonces tal vez se parecen en mayor o menor medida a algo que ya fue resuelto con anterioridad. Si evidencias estos patrones, se vuelve más simple entender las distintas piezas que forman el problema.

Paso 3) Abstracción — La abstracción permite reducir los detalles para centrarse en la información relevante para resolver un problema.

Paso 4) Algoritmos — Cuando una solución está completa, puedes realizar una descripción que permita procesarla paso a paso.

Y si queremos un programa que implemente nuestra solución

Paso 5) Implementar — traducir el algoritmo a un lenguaje de programación.



Próxima Actividad: Lightbot (Autoasistida)



2

A large blue downward-pointing arrow.

<https://www.minijuegos.com/juego/light-bot>

3

A large blue rightward-pointing arrow.

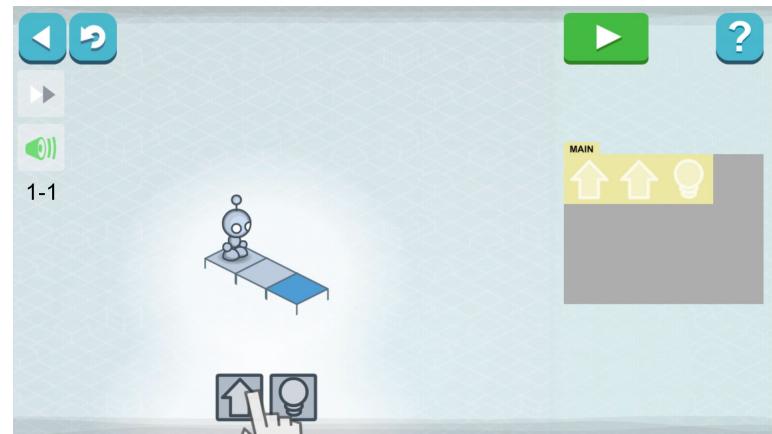
Actividad

Nivel 1: Básicos



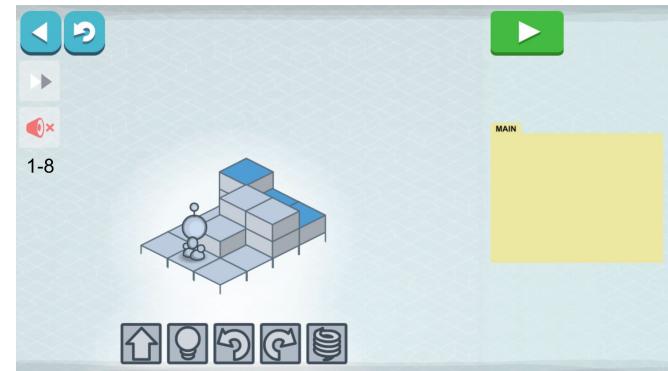
Actividad Desafío 1-1

- ¿Dónde están las **primitivas**?
¿Cuáles son?
 - ¿Dónde se hace el **programa**?
 - ¿Cuál es el **propósito** del programa?
 - ¿Cuáles son las **condiciones iniciales**?
 - ¿Cómo se **ejecuta** el programa?



Actividad Nivel 1: Básicos

- Recomiendan a sus alumnos **apoyarse** tanto en sus compañeros como en **ustedes, los docentes**, si ven que se traban **en algo**
- Sugieran **repensar** la estrategia global si detectan que encaran mal el problema
- Creemos que es importante **llover esta metodología al aula** (indagación y definición de estrategias)
- **No dejen** que los más avanzados **les resuelvan el problema** a los más rezagados

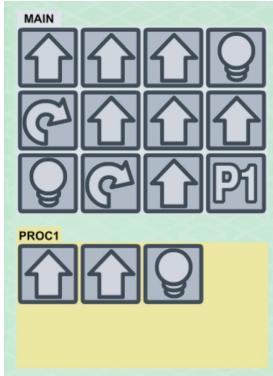


Actividad

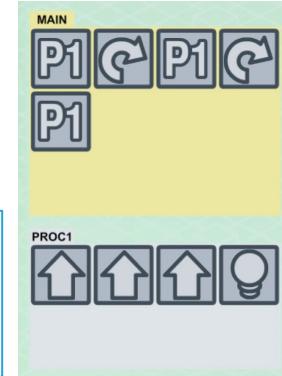
Nivel 2: Procedimientos



Actividad Desafío 2-1



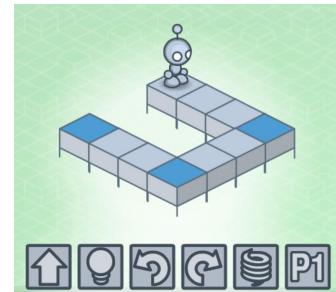
Esta es una posible solución que indica que aprendieron a utilizar el recurso *p1*.



Otra posible solución, con un mejor uso de la idea de procedimientos (aunque es probable que no aparezca).

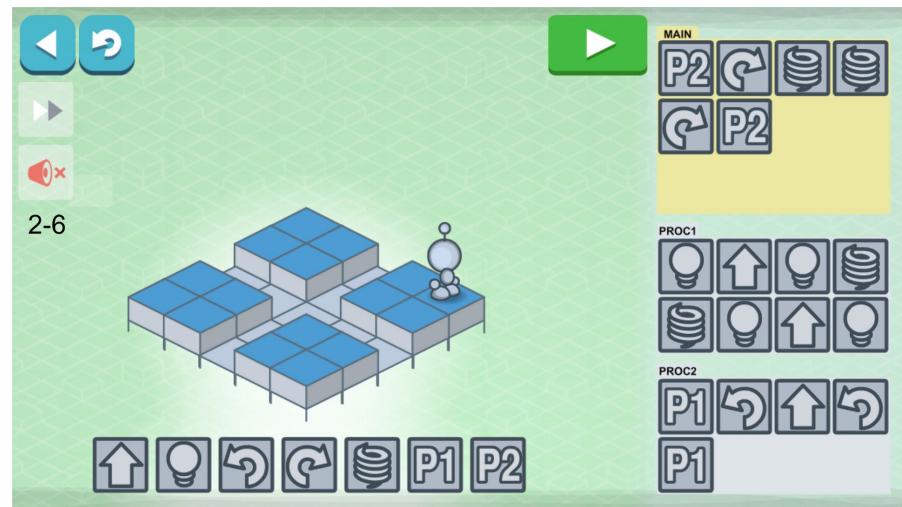


¡Ambas soluciones están bien!
Aunque para algunos desafíos va a ser necesario construir programas con un mejor uso de los procedimientos.



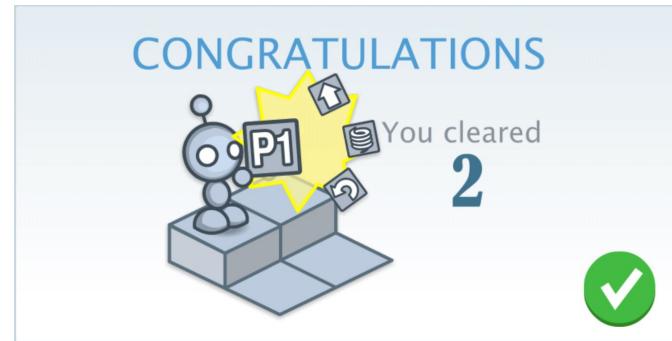
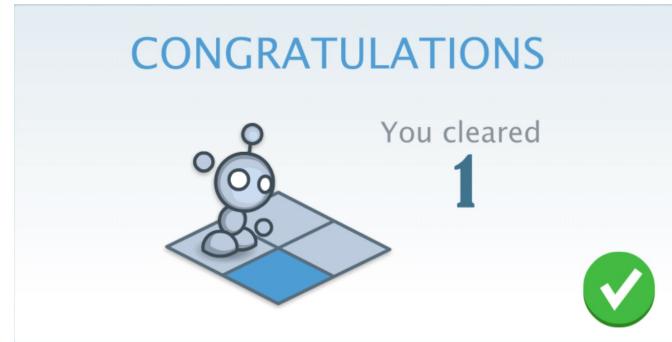
Conclusión Nivel 2: Procedimientos

- Los procedimientos **no son** para ganar casilleros
- En el caso de Lightbot, sirven para **agrupar secuencias de acciones que se repiten**



Conclusión Lightbot

- En cada nivel había un **objetivo** a cumplir
- El cumplimiento se lleva a cabo usando una **estrategia**
- En presencia de patrones que se repiten es conveniente usar **procedimientos**
- En los niveles de estas actividades hay una noción de **estado**



Dibujar en la cuadricula en p5.js

1



<https://p5js.org/>

p5.js

English Español简体中文简体中文

Inicio
Editor

Descargar
Donar
Empezar
Referencia
Bibliotecas
Aprender
Enseñar
Ejemplos

¡Hola!

p5.js es una biblioteca de JavaScript para la programación creativa, que busca hacer que programar sea accesible e inclusivo para artistas, diseñadores, educadores, principiantes y cualquier otra persona! p5.js es gratuito y de código abierto porque creemos que el software y las herramientas para aprenderlo deben ser accesibles para todos.

Usando la metáfora de bosquejar, p5.js tiene un conjunto completo de funcionalidades para dibujar. Sin embargo, no estás limitado solo a dibujar en tu lienzo. Puedes tomar toda la página del navegador como tu bosquejo, incluyendo los objetos HTML5 para texto, entrada, video, cámara web y ...

https://editor.p5js.org/

2

```
1// function setup() {
2//   createCanvas(400, 400);
3// }
4// function draw() {
5//   background(220);
6// }
```

3



Sign Up

User Name

Email

Password

Confirm Password

Sign Up

Or

Login with GitHub

Login with Google



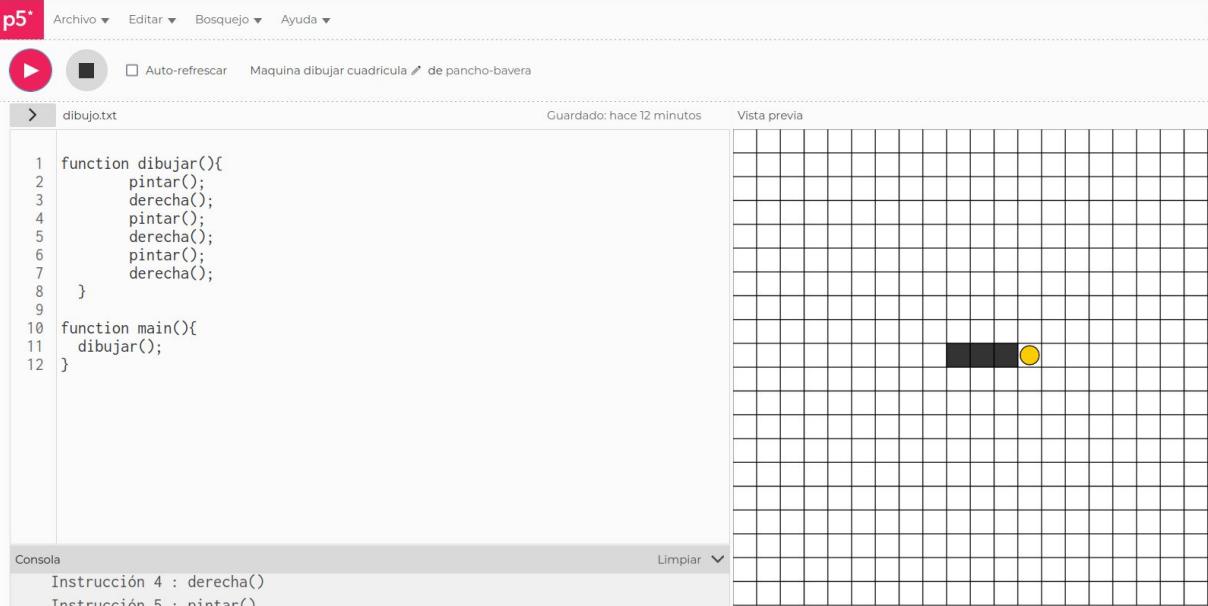
Dibujar en la cuadricula en p5.js

Recuerden duplicar el proyecto para poder guardar los cambios

4



<https://editor.p5js.org/pancho-bavera/sketches/8bDWrmYmg>
o
<https://acortar.link/BvSSFT>



```
p5* Archivo ▾ Editar ▾ Bosquejo ▾ Ayuda ▾ Es
Guardado: hace 12 minutos Vista previa
dibujo.txt Maquina dibujar cuadricula de pancho-bavera
function dibujar(){
  pintar();
  derecha();
  pintar();
  derecha();
  pintar();
  derecha();
}
function main(){
  dibujar();
}
```

Consola

Instrucción 4 : derecha()
Instrucción 5 : pintar()



Dibujar en la cuadricula en p5.js

4



<https://editor.p5js.org/pancho-bavera/sketches/8bDWrmYmg>

<https://acortar.link/BvSSFT>

En este lenguaje ya existen las siguientes funciones/instrucciones predefinidas:

- `izquierda()`: mueve el cursor 1 paso a la izquierda
- `derecha()`: mueve el cursor 1 paso a la derecha
- `arriba()`: mueve el cursor 1 paso hacia arriba
- `abajo()`: mueve el cursor 1 paso hacia abajo
- `pintar()`: pinta en color negro la posición actual del cursor

Puedes definir nuevas funciones y usarlas:

- `function NombreFuncion(){
 instrucciones de la funcion
}`
 - `main` es la función/programa principal
- `function NombreFuncion(parametros){
 instrucciones de la funcion
}`

Puedes usar repeticiones:

- `repetir(valor){instrucciones}`: repite “valor” de veces las instrucciones

```
p5* Archivo ▾ Editor ▾ Bosquejo ▾ Ayuda ▾
Guardado hace 12 minutos
dibujo.txt
Auto-refrescar Maquina dibujar cuadricula ✎ de pancho-bavera
Vista previa
1 function dibujar(){
2     pintar();
3     derecha();
4     pintar();
5     derecha();
6     pintar();
7     derecha();
8 }
9 function main(){
10     dibujar();
11 }
```

Consola

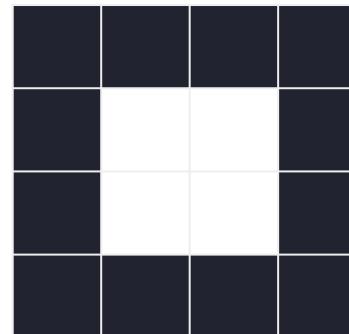
Instrucción 4 : derecha()
Instrucción 5 : pintar()



Actividad

Programamos en p5.js

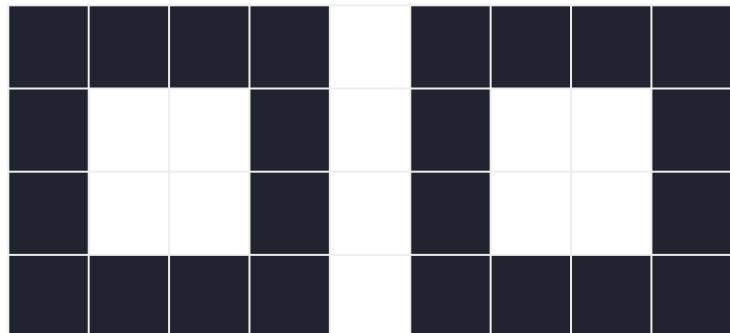
Escribir un programa que
pinte un **cuadrado**



Actividad

Programamos en p5.js

Escribir un programa que
pinte dos cuadros

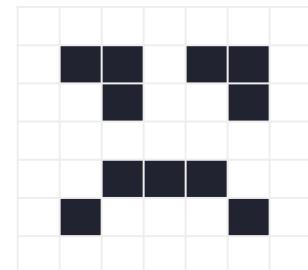
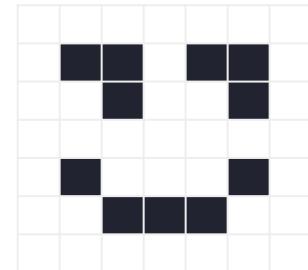
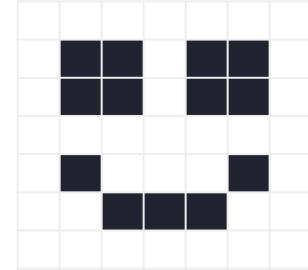


Actividad

Programamos en p5.js

Escribir programas que dibujen cada una de las tres **caritas**

Importante: Pueden utilizar el mismo procedimiento en diferentes programas



Ayuda: utilice funciones con parametros!!!

Actividad

Programamos en p5.js

Escribir un programa que
pinte estos tres **cuadrados**

