Sesión 1

Marco inicial del pensamiento computacional

Pensamiento computacional y robótica educativa · sesión 1 de 6

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Objetivo

- Entender los cuatro pilares del pensamiento computacional con ejemplos cotidianos, sin ordenador de por medio.
- Diagnosticar qué dispositivos, licencias y experiencia previa tiene ya el centro, antes de proponer nada nuevo.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Para empezar

Pregunta

¿Qué os viene a la cabeza cuando digo "pensamiento computacional"?

Casi siempre sale "programar", "robots" o "informática". Al final de la sesión volvemos sobre esta misma lista.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Vídeo: iniciación al pensamiento computacional

Vídeo

▶ Iniciación al pensamiento computacional (2026-27)

Para verlo, inicia sesión con tu cuenta `@edu.gva.es`.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

¿Qué es el pensamiento computacional?

Un **método de resolución de problemas** que utiliza la computación para crear soluciones.

Y, además, un proceso que ayuda a **comprender y resolver dificultades sin usar ordenadores**.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

"Es un pensamiento aplicado a la resolución de problemas y a la representación de sus soluciones, donde los resultados pueden ser procesados por un humano, por los ordenadores, o por una combinación de ambos." — Jeannette Wing

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

De dónde viene: 2006, Jeannette Wing

Aunque hay menciones anteriores, en **2006** Jeannette Wing publicó el artículo que puso nombre y marco al concepto: el "pistoletazo de salida" de este movimiento.

Wing lo planteó como una **habilidad y una actitud de aplicación universal** para todas las personas:

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

"Al igual que se enseña lengua y matemáticas, es necesario desarrollar las habilidades que rodean a este pensamiento."

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Programar por bloques: de Logo a Scratch

- **Años 80:** el lenguaje **Logo** llevó la programación a la escuela; después cayó en desuso.
- **2005-2008: Scratch, Alice, Kodu, AppInventor** y otros devuelven la programación al aula.
- Muchos usan **programación por bloques**: entornos visuales y amigables que permiten trabajar el pensamiento computacional **desde edades muy tempranas**.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Cuatro operaciones mentales, no un aparato

No hace falta un ordenador para practicarlas: papel, el propio cuerpo del alumnado, o una tarea cotidiana bastan.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

1. Descomposición

Dividir un problema grande en partes manejables.

Ejemplo: organizar una excursión de centro no es "un bloque" — es transporte, seguro, autorizaciones, itinerario, comida. Cada parte se resuelve por separado.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

2. Reconocimiento de patrones

Encontrar semejanzas entre problemas para no resolver cada uno desde cero.

Ejemplo: un docente que corrige exámenes reconoce los errores que se repiten curso tras curso, y anticipa dónde va a fallar el alumnado.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

3. Abstracción

Quedarse con lo esencial de un problema y dejar fuera el detalle que no aporta nada.

Ejemplo: un plano de metro no representa calles ni distancias reales, solo líneas y paradas — lo único que importa para orientarse.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

4. Algoritmos

La secuencia ordenada de pasos que resuelve un problema, escrita de forma que cualquiera que la siga llegue al mismo resultado.

Ejemplo: una receta de cocina. Si se saltan pasos o se cambia el orden, el resultado no es el mismo plato.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Las fases, en orden

El pensamiento computacional se aplica como una secuencia de pasos, todos ellos **previos al producto final** (la solución):

1. **Descomponer**
2. **Buscar patrones**
3. **Abstraer**
4. **Crear algoritmos → producto final**

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Qué ganamos al trabajarlo

- **Confianza** para afrontar problemas complejos.
- Capacidad para identificar qué partes se pueden resolver **con medios informáticos**.
- **Creatividad** para encontrar soluciones innovadoras.
- Preparación para aplicar la **estrategia adecuada** en cada momento.
- **Tolerancia** para aceptar soluciones distintas ante un mismo problema.
- Habilidad para **trabajar en equipo** y llegar al producto final.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Y una idea de fondo: la vertiente práctica es consustancial al método — **pensar, crear, probar, rectificar y volver a probar** hasta conseguir el resultado.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Aprender creando: constructivismo (Seymour Papert)

Cuando el alumnado crea un producto —un programa o cualquier otra cosa, digital o no— está **aprendiendo, construyendo, comprobando, rehaciendo y modificando**. Y al analizar los fallos, es capaz de plantear soluciones nuevas y creativas.

Es la idea de **Seymour Papert**: aprender construyendo. El *constructivismo* como aprendizaje en estado puro.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Qué NO es pensamiento computacional

- No es sinónimo de **programar** — se puede pensar de forma computacional sin escribir una línea de código.
- No es sinónimo de **usar tecnología** — una tablet para ver un vídeo no implica pensamiento computacional.
- No es una **asignatura aparte** — se integra en matemáticas, lengua, ciencias o cualquier materia.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Por qué empezar sin dispositivos

Idea clave

Si la primera experiencia es un robot que no hace lo esperado, la atención se va entera al aparato — y el razonamiento de fondo se pierde en la frustración técnica.

Por eso se empieza con actividades desconectadas, donde el fallo siempre es fácil de analizar.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Relación con el currículo

- **DigCompEdu**: dentro de resolución de problemas y creación de contenido digital.
- **LOMLOE Comunitat Valenciana**: vinculado a competencia matemática y competencia STEM, no solo a Tecnología.
- Cabida natural en comprensión lectora de textos secuenciales (instrucciones, procedimientos) y en el método científico.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Puerta de entrada a proyectos STEM/STEAM

Muchos centros quieren lanzar un proyecto STEAM y no saben por dónde empezar si el punto de partida es "hacer un proyecto grande".

Los cuatro pilares, trabajados primero con actividades pequeñas y desconectadas, dan confianza antes de enfrentarse a un proyecto complejo con robótica real.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Evaluar también el proceso

No solo el producto final (un dibujo, un recorrido programado, un proyecto de Scratch) — también el proceso:

- Corregir un error sin ayuda → validación de procesos.
- Reconocer que dos problemas se resuelven igual → reconocimiento de patrones.

Se retoma con una rúbrica completa en la **sesión 6**.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Dos rutas, un producto

Esencial: rutina real del aula → 5–8 pasos → prueba literal con otra pareja. **Desafío:** corregir ambigüedades. Resultado: ficha breve lista para probar.

Profundización: añadir una condición y una variable. **Desafío:** probar dos casos con resultados distintos. Resultado: versión adaptable a dos niveles.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Cierre

Recapitulando

Volved a la lista de la pizarra: ¿qué añadiríais o quitaríais ahora que conocéis los cuatro pilares?

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

Fuentes

- **«El concepto de Pensamiento Computacional»** — INTEF, MOOC *Pensamiento Computacional educativo* (#MoocBot). De aquí salen la definición, la cita de Jeannette Wing (2006) y el recorrido Logo → Scratch → programación por bloques.
- **«Fases del Pensamiento Computacional»** — INTEF, mismo MOOC. De aquí salen las fases en orden, las capacidades que se desarrollan y el constructivismo de Seymour Papert.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

¡Muchas gracias!

Para cualquier duda o consulta:

info@sasogu.es

Samuel Soriano · samuelsoriano.es

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.

© Samuel Soriano · Licencia CC BY-NC-SA 4.0.