



PROPUESTA DE MODELO DE INTEGRACIÓN CURRICULAR DEL PENSAMIENTO COMPUTACIONAL

Mayo 2022

Fernando Posada Prieto

Propuesta de Modelo de Integración Curricular del Pensamiento Computacional

A partir del análisis de los distintos modelos internacionales que han sido desarrollados y puestos en práctica por instituciones, asociaciones, investigadores ... en diversos países del mundo, se propone abordar un borrador de propuesta curricular de cómo incorporar el Pensamiento Computacional (CT) a lo largo de la escolarización obligatoria en España.

Los criterios de selección y diseño en que se basa esta propuesta son:

- **Concepto.** Se apoya en un concepto de CT que sitúa su foco en la resolución creativa de problemas mediante su conversión en computable, es decir, resolubles mediante la aplicación de un agente de computación: ordenador o humano.
- **Inspiración.** Surge de una síntesis de las propuestas curriculares del *CAS del Reino Unido*¹ y del *CSTA de USA*² por ser las más elaboradas incluyendo algunas aportaciones de otras iniciativas.
- **Fundamentación epistemológica.** Recoge contenidos y prácticas que resultan significativas en el contexto de las ciencias de la computación.
- **Enfoque interdisciplinar.** Se seleccionan aquellos conceptos y habilidades cuya aplicación a otras disciplinas o áreas resulta interesante, fácil y significativa.
- **Adaptado al sistema educativo español.** Se adaptan los criterios para los ciclos del sistema educativo español en las etapas de enseñanza obligatoria de 3 a 16 años: Infantil, Primer-Segundo-Tercer Ciclo de Primaria, y Primer-Segundo Ciclo de ESO.

¹ <https://www.barefootcomputing.org/>

² <https://k12cs.org/>

- **Ciclicidad y progresión.** Se estructura una secuencia de complejidad creciente en función de la edad del alumnado que se aborda bajo el principio de “secuencia elaborativa” de contenidos como de hecho acontece en el resto de materias.
- **Anclaje curricular.** La propuesta trata de ser acorde con el currículo del resto de áreas pudiendo ser desarrollada como un área específica o bien como un área transversal mediante una metodología ABP o similar.
- **Relevancia.** La distribución de contenidos pretende familiarizar al alumnado con el razonamiento algorítmico moderando la carga de contenidos informáticos. De esa forma se potencia su incorporación en el aula junto con otros vectores de innovación actuales, y se facilita la formación del profesorado que la imparte.
- **Experiencia.** Se aprovecha la experiencia desarrollada durante estos años en el trabajo de aula en distintos niveles educativos y con diversos tipos de recursos y materiales de pensamiento computacional.
- **Revisable.** Es un modelo sujeto a permanente discusión, revisión y evolución.

Nota:

Este documento ha sido extraído del libro “*Pensamiento Computacional y Currículum. Algo más que aprender a programar*”, escrito por Fernando Posada Prieto (Mayo 2022), publicado con licencia **Creative Commons BY-NC** y coste gratuito en la URL: <https://canaltic.com/blog/?p=4992>



Nivel 0. Infantil. De 3 a 6 años

AP. Algoritmos y programación	DA. Datos: análisis y representación	SC. Sistemas de computación	RI. Redes e Internet	TI. Tecnologías de la Información
<ol style="list-style-type: none"> 1. Construir y aplicar una secuencia ordenada de pasos para resolver un problema o situación. 2. Mostrar orden y precisión en la realización de una tarea por pasos para evitar errores. 3. Reconocer que los ordenadores y robots necesitan instrucciones precisas. 4. Crear y ejecutar un sencillo programa de animación de uno o dos personajes en pantalla usando un entorno visual de programación horizontal por bloques (scratchjr) 5. Diseñar y ejecutar un sencillo programa de control de un robot de suelo o simuladores mediante instrucciones paso a paso usando un entorno visual de programación (bluebot, beebot ...) 6. Interpretar, depurar y modificar programas sencillos elaborados por sí mismo y por otros compañeros/as. 7. Resolver con éxito aplicaciones que proponen juegos de puzzles basados en programación de bloques para resolver situaciones-problema adaptados a su edad. 	<ol style="list-style-type: none"> 1. Reconocer los distintos formatos digitales que se pueden utilizar para representar la información: textos, enlaces, fotografías, dibujos, audios, videos, animaciones ... 2. Distinguir entre algunas de estas formas y explicar los detalles principales de un objeto digital observado. 	<ol style="list-style-type: none"> 1. Identificar y nombrar las principales partes visibles de un ordenador, tablet o robot describiendo su función. 2. Comprender que los dispositivos computacionales no tienen inteligencia y que funcionan gracias al programa de que disponen. 	<ol style="list-style-type: none"> 1. Buscar información visual (imágenes, dibujos, videos ...) sobre temas de interés usando comandos de voz o de texto en el navegador web. 2. Comprender la importancia de comunicarse de manera segura y respetuosa en línea. 3. Saber qué hacer y a quién dirigirse cuando resulta molesto algún contenido o contacto al que se accede. 	<ol style="list-style-type: none"> 1. Utilizar programas en ordenador y/o tablets bajo supervisión docente o parental para crear, almacenar y editar contenido digital. 2. Manejar básicamente los dispositivos tecnológicos del aula: proyección, audios, videos, pizarra interactiva ... 3. Identificar y comprender el funcionamiento básico de las tecnologías de la información dentro y fuera del aula.

Nivel 1. 1º ciclo de Primaria. 1º-2º (6-8 años)

AP. Algoritmos y programación	DA. Datos: análisis y representación	SC. Sistemas de computación	RI. Redes e Internet	TI. Tecnologías de la Información
<ol style="list-style-type: none"> 1. Comprender que los algoritmos se implementan en dispositivos digitales en forma de programas para lograr objetivos determinados. 2. Diseñar algoritmos y programas que utilicen eventos, paralelismo, bucles simples y selección simple (if) 3. Crear narraciones multimedia de varios personajes y escenas con bocadillos de texto y/o locuciones de audio usando un entorno visual de programación horizontal por bloques (scratchjr). 4. Crear y ejecutar programas de control de un robot de suelo mediante una secuencia continua de instrucciones para trazar rutas entre dos puntos teniendo en cuenta las instrucciones de la narrativa que describe el problema (bluebot o truebot). 5. Utilizar el razonamiento lógico para predecir los resultados de un algoritmo o programa. 6. Detectar y corregir errores, mediante iteración y depuración tanto en programas propios como ajenos. 7. Resolver con éxito aplicaciones que proponen juegos de puzzles basados en programación de bloques para resolver situaciones-problema adaptados a su edad. 	<ol style="list-style-type: none"> 1. Reconocer diferentes tipos de datos: textos, números, imágenes, audios, videos 2. Recopilar y presentar datos recopilados en distintos formatos visuales. 3. Aprender que los programas pueden trabajar con diferentes tipos de datos. 4. Identificar y describir patrones en visualizaciones de datos como cuadros o gráficos para realizar predicciones. 	<ol style="list-style-type: none"> 1. Utilizar la terminología adecuada para identificar y describir la función de los componentes físicos comunes de un ordenador, tablet o robot. 2. Reconocer la estructura y funcionalidad comunes de los distintos dispositivos digitales computacionales integrados en nuestra vida cotidiana. 3. Describir los problemas básicos de hardware y software utilizando un vocabulario adecuado. 	<ol style="list-style-type: none"> 1. Realizar búsquedas web sencillas para recopilar contenido digital. 2. Citar la autoría cuando se utilizan las ideas y creaciones de otras personas en el desarrollo de programas y materiales. 3. Mantener en privado las credenciales de tu cuenta personal. 4. Mostrar un uso de los medios tecnológicos de forma segura y responsable. 5. Conocer distintas formas de denunciar contenidos y contactos en línea inadecuados. 	<ol style="list-style-type: none"> 1. Participar de forma colaborativa en un proyecto de programación describiendo su proceso de elaboración y el resultado final. 2. Exponer su trabajo y realizar mejoras a las soluciones en base a los comentarios recibidos. 3. Mostrar conciencia de los criterios de calidad del contenido digital producido. 4. Compartir sus experiencias de tecnología dentro y fuera del aula. 5. Manejar básicamente los dispositivos tecnológicos del aula: proyección, audios, videos, pizarra interactiva ...

Nivel 2. 2º ciclo de Primaria. 3º-4º (8-10 años)

AP. Algoritmos y programación	DA. Datos: análisis y representación	SC. Sistemas de computación	RI. Redes e Internet	TI. Tecnologías de la Información
<ol style="list-style-type: none"> 1. Comparar y afinar distintos algoritmos para realizar una misma tarea determinando cuál es el más apropiado. 2. Utilizar el razonamiento lógico para predecir la salida en un programa a partir del conocimiento de las entradas. 3. Reconocer que existen diferentes soluciones para un mismo problema. 4. Codificar situaciones problema que impliquen el uso de variables para almacenar y operar con datos. 5. Diseñar algoritmos y programas que incluyan secuencias, eventos, variables, operadores aritméticos y de comparación, condicionales bidireccionales (if ... else) y bucles simples. 6. Descomponer los problemas en subproblemas más pequeños y manejables para facilitar el proceso de desarrollo de un programa. 7. Crear narraciones multimedia interactivas de varios personajes y escenas con bocadillos y/o locuciones de audio usando un entorno visual de programación vertical por bloques (scratch) 8. Diseñar trazados geométricos usando comandos de dibujo en una herramienta visual de programación por bloques (scratch). 9. Crear y ejecutar programas de control de un robot de suelo o simuladores para trazar rutas teniendo en cuenta las instrucciones recibidas en la narrativa que describe el problema (lego wedo). 10. Construir distintos modelos de máquinas programables con sensores básicos usando bloques de construcción y diseñando un programa de control que se comporte de acuerdo con un repertorio previo de requisitos funcionales. 11. Asumir diferentes roles, con la guía docente, para colaborar con sus compañeros/as en las etapas de diseño, implementación y revisión del desarrollo del programa. 	<ol style="list-style-type: none"> 1. Reconocer que la ordenación alfabética de los datos en un archivo plano puede mejorar la eficiencia en la búsqueda de información. 2. Comprender cómo los patrones de bits representan números e imágenes. 3. Conocer que los ordenadores transfieren datos en binario. 4. Comprender la relación entre el sistema binario y tamaño de archivo (sin comprimir). 	<ol style="list-style-type: none"> 1. Identificar y conocer el funcionamiento básico de algunos tipos de sensores: distancia, inclinación, temperatura ... incorporando los datos recogidos al control del programa. 2. Diferenciar entre hardware y software aportando ejemplos de la vida cotidiana. 3. Conocer las características diferenciales de las redes fijas, inalámbricas y móviles. 	<ol style="list-style-type: none"> 1. Diferenciar el funcionamiento básico de los distintos servicios soportados en Internet: www, email, videoconferencia ... 2. Realizar búsquedas de información complejas en un navegador web usando operadores y filtros en la página de resultados. 3. Reconocer comportamientos aceptables e inaceptables cuando se utilizan tecnologías y servicios en línea. 4. Mostrar un uso responsable de las tecnologías y los servicios en línea, y conocer distintas formas de denunciar situaciones inapropiadas 	<ol style="list-style-type: none"> 1. Recopilar, organizar y presentar datos e información en formato digital. 2. Crear contenido digital para lograr un objetivo determinado mediante el uso combinado de distintas soluciones de software local y online. 3. Realizar las mejoras adecuadas en las soluciones basándose en los comentarios recibidos.

Nivel 3. 3º ciclo de Primaria. 5º-6º (10-12 años)

AP. Algoritmos y programación	DA. Datos: análisis y representación	SC. Sistemas de computación	RI. Redes e Internet	TI. Tecnologías de la Información
<ol style="list-style-type: none"> 1. Diseñar soluciones descomponiendo un problema y creando una solución para cada una de sus partes. 2. Asumir que existen diferentes soluciones para un mismo problema. 3. Utilizar variables, con datos numéricos (enteros y decimales), cadenas de texto y booleanos (verdadero y falso), operadores y expresiones, bucles condicionales y con contador(for), y procedimientos aplicándolos en el control del programa. 4. Codificar problemas aritméticos cotidianos mediante el uso de variables y abstracción para solucionar problemas tipo. 5. Identificar similitudes y diferencias en situaciones y usarlas para resolver problemas mediante reconocimiento de patrones. 6. Incorporar los diagramas de flujo al diseño algorítmico como paso previo a la creación de un programa. 7. Diseñar, escribir y depurar programas de forma modular usando procedimientos e interiorizando sus ventajas. 8. Crear animaciones interactivas y videojuegos básicos poniendo especial énfasis en el componente lúdico (control de tiempo, puntuación, tentativas ...), la interacción del usuario (teclado, botones en pantalla, ratón ...) y el uso de procedimientos modulares en un entorno visual de programación vertical por bloques (scratch) 9. Diseñar trazados geométricos usando comandos de dibujo en una herramienta visual de programación por bloques (scratch). 10. Modelar un sistema domótico básico construido con placas programables (crumble, microbit, makey-makey ...) y componentes electrónicos (leds, motores dc, servomotores, sensores ...) programando con una herramienta visual. 11. Manejar un modelo de robots programables con sensores básicos y diseñando un programa de control con herramientas visuales que se comporte de acuerdo con un repertorio previo de requisitos funcionales (mbot, codey rocky, lego ev3) 12. Modelar una aplicación de inteligencia artificial como chatbots o juegos interactivos basada en el reconocimiento de textos, imágenes o sonidos usando una herramienta de programación visual (ml4k o pictoblox) para atender las necesidades de un colectivo de usuarios/as (aprendizaje-servicio). 13. Asumir diferentes roles, con la guía docente, para colaborar con sus compañeros/as en las etapas de diseño, implementación y revisión del desarrollo del programa. 	<ol style="list-style-type: none"> 1. Comprender cómo los números, imágenes, sonidos y conjuntos de caracteres utilizan los patrones de bits para su almacenamiento y transmisión por las redes. 2. Comprender la relación entre la resolución y la profundidad del color, incluido su efecto sobre el tamaño de un archivo de imagen. 3. Reconocer el código hexadecimal que identifica el color de cada píxel en una imagen digital. 4. Comprender la relación entre binario y tamaño del archivo (sin comprimir). 5. Adecuar el formato de los elementos multimedia que se integran en un programa. 6. Valorar la importancia del entrenamiento de los programas de Inteligencia Artificial creados para recopilar datos y responder al usuario de una forma más precisa. 	<ol style="list-style-type: none"> 1. Analizar qué tareas realizan mejor los humanos y cuáles los ordenadores. 2. Comprender las funciones principales del sistema operativo en un dispositivo. 3. Conocer la existencia de distintos sistemas operativos para diferentes o un mismo tipo de dispositivos: ordenadores, tablets, smartphones ... 4. Reconocer el funcionamiento de distintos tipos de sensores: distancia, luz, sonido, temperatura ... incorporando los datos recogidos al control del programa. 5. Comprender la función de las principales partes internas de la arquitectura informática básica en un ordenador o tablet. 6. Determinar posibles soluciones para resolver problemas simples de software y software utilizando estrategias comunes de resolución. 	<ol style="list-style-type: none"> 1. Utilizar los motores de búsqueda de forma eficaz usando operadores y filtrando los resultados por parámetros. 2. Seleccionar, combinar y utilizar servicios de Internet. 3. Mostrar un uso responsable de las tecnologías y los servicios en línea, y conocer distintas formas de denunciar situaciones inapropiadas. 4. Trabajar en línea de forma individual y colaborativa en una plataforma de aula virtual con el adecuado respeto y responsabilidad. 5. Analizar los problemas de ciberseguridad del mundo real y cómo se puede proteger la información personal. 	<ol style="list-style-type: none"> 1. Emitir valoración sobre el contenido digital al evaluarlo y reutilizarlo para una audiencia concreta. 2. Evaluar la idoneidad de los dispositivos digitales, servicios de Internet y software de aplicación para lograr los objetivos dados. 3. Reconocer los problemas éticos que rodean la aplicación de la tecnología de la información. 4. Diseñar criterios para evaluar críticamente la calidad de las soluciones, utilizando los criterios para identificar mejoras y realizar los ajustes adecuados a la solución. 5. Utilizar distintos servicios de Internet para publicar contenidos digitales en su portfolio personal (blog) con los productos elaborados. 6. Documentar la implementación de un programa con comentarios en el código y recursos gráficos con intención de hacerlo más legible a los demás.

<p>14. Modelar mediante abstracción un sistema físico real para crear un programa de simulación (ejemplo: lanzamiento de un dado o una moneda al aire) que permite la experimentación y el análisis de datos con intención de alcanzar conclusiones.</p> <p>15. Aplicar la descomposición de un problema complejo en problemas más sencillos para repartir su codificación entre los miembros del equipo y resolver el programa final de forma colaborativa.</p>				<p>7. Participar en una comunidad online de programadores jóvenes compartiendo los programas diseñados.</p> <p>8. Respetar los derechos de propiedad intelectual y otorgar la atribución adecuada (cita) al crear programas remezclando código elaborado por otras personas.</p>
--	--	--	--	--

Nivel 4. 1º ciclo de Secundaria. 1º-2º ESO (12-14 años)

AP. Algoritmos y programación	DA. Datos: análisis y representación	SC. Sistemas de computación	RI. Redes e Internet	TI. Tecnologías de la Información
<ol style="list-style-type: none"> 1. Reconocer que algunos problemas comparten características comunes y utilizan el mismo algoritmo para resolverse. 2. Valorar el rendimiento de los algoritmos y seleccionar el más adecuado para la misma tarea. 3. Representar algoritmos utilizando diagramas de flujo. 4. Diseñar algoritmos y programas que incorporen sentencias IF anidadas, listas de datos, negación de operadores y funciones personalizadas que incluyan el uso de parámetros. 5. Diferenciar entre procedimientos y funciones, parámetros y valores utilizándolos de manera adecuada. 6. Depurar y mejorar programas elaborados por sí mismo o por otras personas. 7. Crear animaciones interactivas y videojuegos básicos poniendo especial énfasis en el componente lúdico (control de tiempo, puntuación, tentativas ...), la interacción del usuario (teclado, botones en pantalla ...) y el uso de funciones con parámetros en un entorno visual de programación (scratch) 8. Modelar un sistema domótico básico construido con placas programables (microbit, makey-makey, arduino ...) y componentes electrónicos (leds, motores dc, servomotores, sensores ...) programando con una herramienta de programación visual por bloques (mblock ...) 9. Manejar un modelo de robots programables con sensores básicos y diseñando un programa de control con herramientas visuales que se comporte de acuerdo con un repertorio previo de requisitos funcionales (mbot, edison, lego ev3, maker...) 10. Diseñar una app específica para android usando una herramienta de programación visual (mit app Inventor o kodular) a partir de las necesidades detectadas en una audiencia a través de un proceso creativo completo: diseño, programación y publicación. 11. Modelar una aplicación de inteligencia artificial como chatbots, juegos interactivos, soluciones de accesibilidad ... basada en el reconocimiento de textos, imágenes o sonidos usando una herramienta de programación visual (watson assistant, ml4k o pictoblox) para atender las necesidades de un colectivo de usuarios/as (aprendizaje-servicio). 12. Asumir diferentes roles, con la guía docente, para colaborar con sus compañeros/as en las etapas de diseño, implementación y revisión del desarrollo del programa. 	<ol style="list-style-type: none"> 1. Conocer la relación entre el tamaño del archivo y la calidad de una imagen. 2. Conocer qué es una hoja de cálculo y usarla para almacenar y gestionar sus datos. 3. Consultar datos en una tabla utilizando un lenguaje de consulta típico. 4. Adecuar el formato de los elementos multimedia que se integran en un programa. 5. Valorar la importancia del entrenamiento de los programas de Inteligencia Artificial creados para recopilar datos y responder al usuario de una forma más precisa. 	<ol style="list-style-type: none"> 1. Conocer la existencia de una variedad de sistemas operativos y del software de aplicación para el mismo hardware. 2. Comprender el ciclo de instrucción de la arquitectura Von Neumann de un dispositivo computacional incluyendo la forma en que almacenan los datos en la memoria. 3. Reconocer el funcionamiento de distintos tipos de sensores: distancia, luz, sonido, temperatura ... incorporando los datos recogidos al control del programa. 4. Identificar y solucionar problemas con dispositivos computacionales y sus componentes. 5. Diseñar proyectos que combinen componentes de hardware y software para recopilar e intercambiar datos. 	<ol style="list-style-type: none"> 1. Conocer cómo los motores de búsqueda clasifican los resultados de búsqueda. 2. Diseñar páginas web estáticas usando HTML y CSS. 3. Comprender la transmisión de datos entre dispositivos digitales a través de redes, incluida Internet, es decir, direcciones IP y conmutación de paquetes. 4. Conocer los nombres del hardware (concentrador, enrutador, conmutador ...) y los nombres de los protocolos (SMTP, POP, FTP, TCP/IP...) asociados a los sistemas informáticos en red. 5. Utilizar tecnologías y servicios en línea de forma segura y saber cómo identificar y denunciar conductas inapropiadas. 6. Trabajar en línea de forma individual y colaborativa en una plataforma de aula virtual con el adecuado respeto y responsabilidad. 	<ol style="list-style-type: none"> 1. Empezar proyectos tecnológicos creativos que recopilen y evalúen datos para satisfacer las necesidades de un grupo de usuarios conocido (aprendizaje - servicio). 2. Justificar la elección de distintas combinaciones de dispositivos digitales, servicios de Internet y software para lograr determinados objetivos. 3. Evaluar la confiabilidad del contenido digital y considerar la usabilidad al diseñar y crear artefactos digitales para una audiencia conocida. 4. Identificar y explicar cómo el uso de la tecnología puede impactar en la sociedad. 5. Diseñar criterios para que los usuarios evalúen la calidad de las soluciones, utilizar el feedback de los usuarios para identificar mejoras y hacer los ajustes necesarios a la solución. 6. Utilizar un sistema de gestión de contenidos (CMS)(tipo blog) para publicar los proyectos. 7. Documentar la implementación de un programa con comentarios en el código y recursos gráficos con intención de hacerlo más legible a los

<p>13. Modelar mediante abstracción un sistema físico real para crear un programa de simulación (ejemplo: lanzamiento de un dado o una moneda al aire) que permite la experimentación y el análisis de datos con intención de alcanzar conclusiones.</p> <p>14. Aplicar la descomposición de un problema complejo en problemas más sencillos para repartir su codificación entre los miembros del equipo y resolver el programa final de forma colaborativa.</p>			<p>7. Analizar los problemas de ciberseguridad del mundo real y cómo se puede proteger la información personal.</p>	<p>demás.</p> <p>8. Participar en una comunidad online de programadores jóvenes compartiendo los programas diseñados.</p> <p>9. Respetar los derechos de propiedad intelectual y otorgar la atribución adecuada (cita) al crear programas remezclando código elaborado por otras personas.</p>
--	--	--	---	--

Nivel 5. 2º ciclo de Secundaria. 3º-4º ESO (14-16 años)

AP. Algoritmos y programación	DA. Datos: análisis y representación	SC. Sistemas de computación	RI. Redes e Internet	TI. Tecnologías de la Información
<ol style="list-style-type: none"> 1. Comprender que una solución recursiva a un problema se aplica repetidamente la misma solución a instancias cada vez más pequeñas del problema. 2. Evaluar la efectividad de algoritmos y modelos para la resolución de problemas similares. 3. Aprender la noción de rendimiento de los algoritmos y apreciar su diferente rendimiento para la misma tarea. 4. Representar algoritmos utilizando lenguaje estructurado. 5. Diseñar algoritmos y programas que incorporen bucles pre (Mientras-While) y post-condición (Hasta-Until), procedimientos y funciones, iteración recursiva, y listas de datos. 6. Crear animaciones interactivas y videojuegos avanzados poniendo especial énfasis en el componente lúdico (control de tiempo, puntuación, tentativas ...), la interacción del usuario (teclado, botones en pantalla ...) y el uso de funciones con parámetros en un entorno visual de programación por bloques (tipo scratch) 7. Crear programas básicos utilizando una herramienta de programación textual orientada a objetos como arduino, python, javascript ... que incorporen el uso de librerías estándar y funciones personalizadas. 8. Diseñar y escribir programas modulares anidados que refuerzan la reutilización utilizando subrutinas siempre que sea posible. 9. Modelar un sistema domótico básico construido con placas programables (arduino ...) y componentes electrónicos (leds, motores dc, servomotores, sensores ...) programando con una herramienta de programación visual por bloques (mblock ...) o textual (arduino...) 10. Manejar un modelo de robots programables con sensores básicos y avanzados y diseñando un programa de control con herramientas visuales que se comporte de acuerdo con un repertorio previo de requisitos funcionales (lego mbot, lego ev3, arduino maker ...) 11. Diseñar una app específica para android usando una herramienta de programación visual (mit app Inventor o kodular) a partir de las necesidades detectadas en una audiencia a través de un proceso creativo completo: diseño, programación y publicación. 	<ol style="list-style-type: none"> 1. Diferenciar entre los datos utilizados por un programa (variables) y la estructura de almacenamiento de esos datos. 2. Realizar operaciones de conversión de datos en formato decimal, binario y hexadecimal. 3. Adecuar el formato de los elementos multimedia que se integran en un programa. 4. Conocer qué es una hoja de cálculo y usarla para almacenar y gestionar sus datos. 5. Consultar datos en una tabla utilizando un lenguaje de consulta típico. 6. Valorar la importancia del entrenamiento de los programas de Inteligencia Artificial creados para recopilar datos y responder al usuario de una forma más precisa. 	<ol style="list-style-type: none"> 1. Conocer que los procesadores tienen conjuntos de instrucciones y que estos se relacionan con instrucciones de bajo nivel ejecutadas por un ordenador. 2. Comprender y explicar los procesos multitarea por ordenador. 3. Reconocer el funcionamiento de distintos tipos de sensores: distancia, luz, sonido, temperatura ... incorporando los datos recogidos al control del programa. 4. Comparar los niveles de abstracción e interacciones entre el software de aplicación, software de sistema y las capas de hardware. 5. Crear pautas para conseguir estrategias de resolución de incidencias que otras personas pueden utilizar. 	<ol style="list-style-type: none"> 1. Conocer el propósito del hardware y los protocolos asociados con los sistemas informáticos en red. 2. Comprender el modelo cliente-servidor, incluida la forma en que las páginas web dinámicas utilizan secuencias de comandos del lado del servidor y que los servidores web procesan y almacenan los datos introducidos por los usuarios. 3. Reconocer que la persistencia de datos en Internet requiere una protección cuidadosa de la identidad y la privacidad en línea. 4. Conocer el hardware asociado con los sistemas informáticos de redes, incluidas las WAN y LAN. 5. Comprender el propósito y cómo funcionan los sistemas informáticos en red, incluidas las direcciones MAC. 6. Analizar los problemas de ciberseguridad del mundo real y cómo se puede proteger la información personal. 	<ol style="list-style-type: none"> 1. Empezar proyectos tecnológicos creativos que analicen y evalúen datos para satisfacer las necesidades de un grupo de usuarios conocido (aprendizaje - servicio). 2. Diseñar y crear con eficacia artefactos digitales para una audiencia más amplia o remota. 3. Considerar las propiedades de los medios al importarlos en artefactos digitales. 4. Explicar y justificar cómo el uso de la tecnología impacta en la sociedad, desde la perspectiva de los aspectos sociales, económicos, políticos, legales, éticos y morales. 5. Comprender las cuestiones éticas que rodean la aplicación de la tecnología de la información y la existencia de marcos legales que rigen su uso, por ejemplo, la ley de protección de datos, derechos de autor ... 6. Utilizar un sistema de gestión de contenidos (CMS)(tipo blog) para publicar los proyectos en Internet.

<p>12. Modelar una aplicación de inteligencia artificial como chatbots, juegos interactivos, soluciones de accesibilidad ... basada en el reconocimiento de textos, imágenes o sonidos usando una herramienta de programación visual (watson assistant) para atender las necesidades de un colectivo de usuarios/as (aprendizaje-servicio).</p> <p>13. Aplicar un enfoque modular para la detección y corrección de errores.</p> <p>14. Comprender que algunos problemas no se pueden resolver computacionalmente.</p> <p>15. Asumir diferentes roles, con la guía docente, para colaborar con sus compañeros/as en las etapas de diseño, implementación y revisión del desarrollo del programa.</p> <p>16. Modelar mediante abstracción un sistema físico real para crear un programa de simulación (ejemplo: lanzamiento de un dado o una moneda al aire) que permite la experimentación y el análisis de datos con intención de alcanzar conclusiones.</p> <p>17. Aplicar la descomposición de un problema complejo en problemas más sencillos para repartir su codificación entre los miembros del equipo y resolver el programa final de forma colaborativa.</p>			<p>7. Trabajar en línea de forma individual y colaborativa en una plataforma de aula virtual con el adecuado respeto y responsabilidad.</p>	<p>7. Documentar la implementación de un programa con comentarios en el código y recursos gráficos con intención de hacerlo más legible a los demás.</p> <p>8. Participar en una comunidad online de programadores jóvenes compartiendo los programas diseñados.</p> <p>9. Respetar los derechos de propiedad intelectual y otorgar la atribución adecuada (cita) al crear programas remezclando código elaborado por otras personas.</p>
---	--	--	---	---

