Pensamiento computacional

Para consensuar los contenidos mínimos fundamentales que se espera que los estudiantes obtengan durante su escolaridad, en septiembre de 2018 se aprobaron los Núcleos de Aprendizaje Prioritarios (NAP) de Educación Digital, Programación y Robótica. Es a partir de esta resolución que la educación digital, la programación y la robótica comenzarán a ser obligatorias en todos los establecimientos del país. Según lo determinado allí, las jurisdicciones llevarán adelante la implementación de los NAP y su inclusión en sus documentos curriculares, adoptando diferentes estrategias y considerando las particularidades de sus contextos, necesidades, realidades y políticas educativas en el lapso de dos años.

Algunas metas de aprendizaje que se proponen son:

- Iniciarse en la resolución de situaciones problemáticas transitando las diferentes etapas del proceso: identificar el problema, formular hipótesis, investigar y elaborar conclusiones.
- Iniciarse en el desarrollo del pensamiento computacional como estrategia para el planteo y la resolución de situaciones problemáticas.
- Intercambiar ideas, realizar diversos registros y analizarlos haciendo uso de diversas herramientas digitales.

Por ello, en *Malabares matemáticos 5*, incluimos la propuesta + Di gi tal, con actividades que refuerzan los temas abordados en cada módulo. En ellas se utilizan recursos digitales que potencian el desarrollo del pensamiento computacional, principalmente a través de la programación.

Pero... ¿qué entendemos por pensamiento computacional?

El *pensamiento computacional* es un proceso que permite formular problemas de manera que sus soluciones puedan representarse como secuencias de instrucciones, llamadas algoritmos.

Este proceso de resolución de problemas comprende las siguientes características:

- Organizar y analizar lógicamente la información.
- Representar la información a través de abstracciones (por ejemplo, simulaciones).
- Automatizar estableciendo una serie de pasos ordenados para llegar a la solución, es decir, utilizando algoritmos.
- Identificar, analizar e implementar posibles soluciones con el objetivo de lograr la combinación más efectiva y eficiente de pasos y recursos.

Apunta a generar en los niños una forma de pensar que les permita aprender a plantearse problemas y sus soluciones, cumpliendo una secuencia determinada de pasos en el proceso. El pensamiento computacional ayuda a tomar decisiones de una manera ordenada, secuenciada, lógica y sin ambigüedades. Algo que a veces resulta difícil en el ámbito de las ciencias de corte más social.

Hay muchas formas de desarrollar el pensamiento computacional en la escuela. Aquí aportamos algunas maneras de incluirlo. Lo importante es que una vez que los alumnos logran fluidez en el uso de las herramientas, empiezan a aplicarlo por su cuenta y en un espacio más amplio del propuesto.

Si bien el pensamiento computacional está ligado al razonamiento que se logra programando frente a una computadora, no debe trabajarse necesariamente de esta forma; podemos abordarlo de manera *unplugged* (desconectada/sin PC), es decir, mediante ejercicios y experiencias de resolución de problemas, realizando trabajos de conceptualización sobre los pasos llevados a cabo en la experiencia.

¿Qué relación hay o en qué medida se diferencian las varias formas de pensamiento computacional de aquellas correspondientes al pensamiento matemático?

Pensemos en un caso. Un alumno desea graficar datos de un experimento y encuentra un patrón común entre estos datos.
La matemática le permite expresar ese patrón mediante una ecuación o una fórmula. De esta manera va a poder predecir resultados posibles.

Cuando incluimos las nuevas tecnologías, los alumnos pueden usar una PC para dar un paso más allá de lo que a primera vista se puede indagar y así lograr hacer análisis con resultados basados en la evidencia.

Es ahí donde aparece el pensamiento computacional, cuando se usan métodos de simulación, redes, recolección automática de datos, razonamiento algorítmico y programación, entre otros.

¿Cómo trabajar con cada una de las propuestas + Digital?

MÓDULO 1. Tramo 1

A tener en cuenta

Para guardar imágenes, las computadoras convierten la información en una cuadrícula de píxeles. Cada línea de píxeles se representa a su vez con números, según su color y posición. En las imágenes a color, se trabaja con tres colores básicos a partir de los cuales construyen todos los demás: el rojo, el azul y el verde. Este patrón es conocido como RGB (Red, Green, Blue).
Cada píxel tiene reservada una posición en la memoria de la computadora para almacenar la información sobre el color que debe presentar. Los bits de profundidad de color marcan cuántos bits de información disponemos para almacenar el número del color asociado según la paleta usada. Con esta información, la tarjeta gráfica del ordenador genera unas señales de voltaje adecuadas para representar el correspondiente color en el monitor.
A más bits por píxel, mayor número de variaciones de un color primario podemos tener. Para 256 colores se precisan 8 bits (sistema básico), para obtener miles de colores necesitamos 16 bits (color de alta densidad) y para obtener millones de colores hacen falta 24 bits (color verdadero).

MÓDULO 1. Tramo 3

A tener en cuenta

El programa *Scratch* es un lenguaje de programación creado por el MIT, diseñado para que cualquier persona pueda realizar diferentes programas de manera muy intuitiva utilizando bloques agrupados en categorías.
Las categorías que se utilizan en este ejemplo son: EVENTO (al presionar espacio), SENSORES (Preguntas), VARIABLES, CONTROL (si/si no), APARIENCIA (decir...), OPERADORES (unir) (multiplicación).
En esta placa se propone realizar un programa para que los alumnos puedan practicar cálculos mentales.
Se sugiere realizarlo entre todos en el aula y utilizarlo durante las clases para hacer un "campeonato de cálculos mentales" e invitarlos a realizarlo en una computadora de su casa para seguir practicando.
Es importante tener en cuenta que hay varias maneras de realizar lo mismo. Se puede invitar a pensar qué cambios podrían realizarse para mejorar la calculadora.
En caso de querer introducir otras operaciones, simplemente, se deberían incluir otros operadores.

MÓDULO 2. Tramo 1

A tener en cuenta

Usamos los metros para medir las longitudes; litros, para medir capacidades, y el tiempo, lo medimos en horas, minutos y segundos. Para medir la capacidad de almacenamiento de información, utilizamos los *Bytes*. La unidad más utilizada actualmente es GB (*Gigabyte* o "*Giga*") que equivale a 1.000 millones de *Bytes*.
Para calcular la información que se utiliza en las diferentes unidades, dividimos o multiplicamos por 1.024:

- 1 *Terabyte* = 1.024 GB
- 1 *Gigabyte* = 1.024 MB
- 1 *Megabyte* = 1.024 KB
- 1 *Kilobyte* = 1.024 Bytes
- 1 *Byte* = 8 bits
- 1 Bit = 1 (uno) o 0 (cero)

Si un Byte, son 8 bits, la base es 8, a diferencia del sistema decimal, cuya base es 10.

- 1.024 es un múltiplo de 8.

  1.024 : 8 = 128

  128 : 8 = 16

  16 : 8 = 2

Este último "2", representa las dos únicas posibilidades: el uno (1) o el cero (0).

Para convertir algo que tenemos en *Gigas* a *Megas*, tendremos que multiplicar por 1.024.
Si de Megabytes lo queremos pasar a una unidad inferior como por ejemplo los Bytes, primero haremos una multiplicación por 1.024, y después volveremos a multiplicar por 1.024 el resultado.

MÓDULO 2. Tramo 2

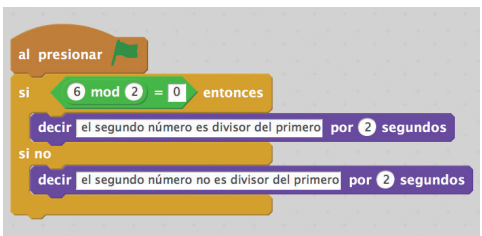
A tener en cuenta

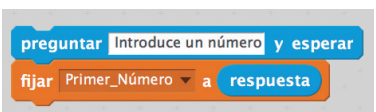
Se propone realizar un programa que nos muestre si un determinado número es divisible o no por otro.

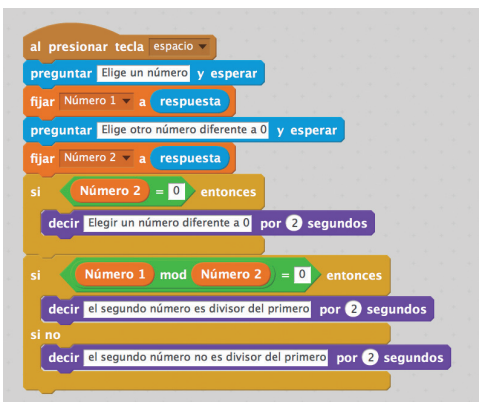
En este caso no se utilizan variables, pero sugerimos que una vez realizado el programa, como respuesta a la pregunta "¿Se te ocurre otra forma de hacerlo?", se trabaje con dos variables para reemplazar los números "6" y "2". ¿Qué logramos con el uso de variables? Que el programa sea interactivo. Para ello deberemos combinarla con el sensor "Pregunta - Respuesta".

Es importante que analicen qué pasa si se coloca como segundo número el valor 0. ¿Puede realizarse la división? ¿Cómo podría evitarse? Una vez analizado entre todos, podemos sugerir una manera de resolverlo.

Para ver las capturas de Scratch en color, escaneá este código.





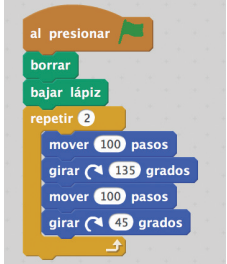


MÓDULO 3. Tramo 2

A tener en cuenta

Uno de los pilares del Pensamiento Computacional está dirigido a lograr "programaciones económicas". ¿Qué significa esto? Lograr programaciones cortas, que tengan la menor cantidad de bloques posibles. De esta manera, son más fáciles de decodificar por otros programadores.

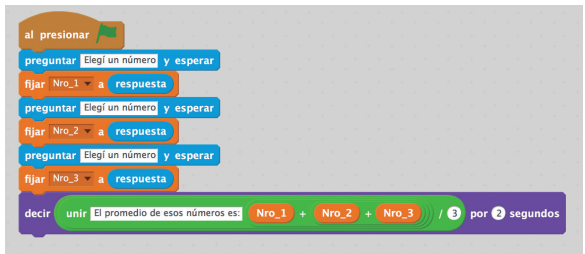
El bloque "repetir" se encuentra dentro de la categoría "CONTROL". Si en el mismo ejemplo de la placa quisiéramos realizar un "hexágono regular", deberíamos cambiar "repetir 4" por "repetir 6" y el giro, en lugar de ser de 90°, debería ser 60° (360°: 6).



En cambio, si quisiéramos realizar un rombo, podríamos repetir una misma estructura 2 veces. Siempre teniendo en cuenta el ángulo que deberá girar. Como tenemos dos ángulos con valores diferentes, podemos realizarlos en una misma estructura y luego repetir el bloque completo.

MÓDULO 4. Tramo 1

A tener en cuenta



En esta placa es importante tener en cuenta que *Scratch* reconoce como números decimales a aquellos que tienen un "punto", es decir, no utiliza la coma. Por otro lado, se apunta a mostrar cómo los bloques pueden combinarse entre sí para realizar cuentas más complejas.

Al final se invita a realizar una calculadora de promedios. Previamente se debería recordar: ¿qué datos nos permiten interactuar con el usuario? Los sensores de preguntas.
¿Qué bloques nos permiten guardar información durante un determinado tiempo? Las variables.
¿Qué bloques nos permiten realizar cálculos? Los operadores matemáticos.

Luego, se propone una etapa de exploración para que los alumnos intenten resolver utilizando los bloques nombrados. El desafío: "realizar una calculadora de promedios". Esta podría ser una de las posibles respuestas.

MÓDULO 4. Tramo 3

A tener en cuenta

BlocksCAD es una página web basada en bloques tipo *Scratch* que nos permite, entre otras tantas cosas, visualizar e imprimir, con una impresora 3D, diferentes cuerpos geométricos. Trabaja a partir de algunas formas que considera "básicas" como los cilindros, esferas y cubos. A partir de éstas, combinándolos con otros bloques, crea diferentes geometrías.
El entorno lo podemos dividir en tres partes:

1. Área de Programa: conjunto de bloques que representan las instrucciones a ejecutar para "renderizar" (hacer) el modelo 3D. Se arrastran desde la barra de bloques y se van encajando unos con otros para determinar la lógica de ejecución-construcción.
2. Barra de bloques: Paleta que contiene los bloques que se pueden utilizar en el área de programa. Los bloques se arrastran de una zona a otra.
3. Área de dibujo o renderizado: Al darle al botón "Hacer" (o "Render"), el programa ejecuta el modelo 3D a partir de los bloques que aparezcan en el área de programa.

Además, tenemos la típica Barra de Herramientas para manejar los archivos, determinar las preferencias del entorno o acceder a la ayuda y ejemplos del programa.