nclab
Learning for Industry 4.0

KAREL EL ROBOT:

PLAN DE LECCIÓN PARA LA HORA DE CÓDIGO

DESCRIPCIÓN DE LA ACTIVIDAD

Un robot determinado llamado Karel corre a través de las selvas, nada en los mares y escala montañas heladas, coleccionando cosas útiles como golosinas y arañas mientras trata de no chocar contra bardas. Karel es un robot después de todo y necesita a un humano para que lo guíe, un humano dispuesto a escribir programas simples para que Karel sepa qué camino tomar, qué recoger y dónde ponerlo.

Karel el Robot le enseña a los estudiantes a cómo escribir programas usando el lenguaje de programación Karel, que se basa en Python. Comenzando con pulsaciones simples en el Juego 1, los estudiantes luego pasan a escribir líneas de código para crear y ejecutar programas. Aprenden a reconocer patrones que pueden ser escritos como bucles, probar condiciones, y escribir comandos definidos para procedimientos de rutina.

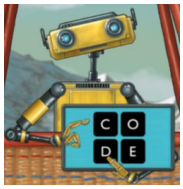
La actividad de Karel El Robot consiste en 15 minijuegos. La mayor parte de la codificación ya está hecha. Los estudiantes rellenan unas pocas líneas para completar cada programa, y luego prueban el código ejecutando a Karel el Robot a través de un laberinto. Al completar cada juego exitosamente se desbloquea el siguiente. La actividad está diseñada para estudiantes de 10 a 15 años, pero puede ser disfrutada por cualquier persona que quiera tratar la programación de computadoras.

OBJETIVOS

Los estudiantes aprenderán a:

Controlar el movimiento con los comandos de `go` (`avanza`), `left` (`izquierda`) y `right` (`derecha`). Recoger y colocar objetos con los comandos `get` (`recoger`) y `put` (`pon`). Codificar una serie de instrucciones con estos comandos básicos.	Juegos 1 a 3
Crear bucles `repeat` (`repite`) basados en patrones repetidos. Repetir comandos individuales y secuencias de comandos.	Juegos 4 a 6
Poner a prueba condiciones usando sentencias `if` (`si`).	Juegos 7 a 9
Crear `while` loops, (bucles `mientras`) para repetir comandos hasta que una condición continua sea falsa.	Juegos 10 a 12
Definir un comando (`def nombre_descriptivo`) que llame a una serie de comandos. Utilizar el comando en el programa principal.	Juegos 13 a 15

MATERIALES Y PREPARACIÓN

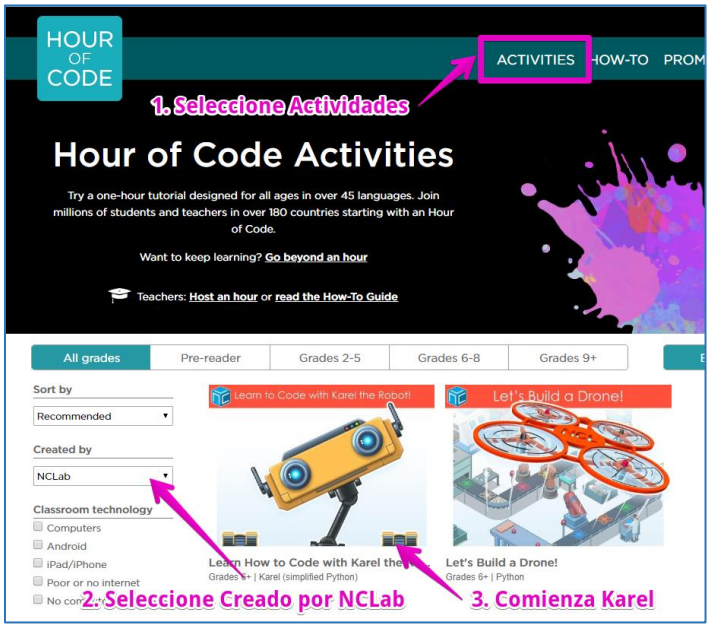


Computadoras personales o tabletas, con acceso a Internet: una para cada estudiante o par de estudiantes.

- Ambas plataformas PC y MAC son compatibles.
- Usa los navegadores Chrome, Firefox, Safari o Edge. Internet Explorer NO es compatible y puede causar problemas con la visualización gráfica.
- Se prefieren las computadoras de escritorio, computadoras portátiles y Chromebooks. Los estudiantes encuentran que es más fácil escribir con un teclado.
- Las tabletas iPad son compatibles. Haz que los estudiantes usen sus iPads en modo horizontal para obtener mejores resultados. Las tabletas de Android no son compatibles en este momento.
- Los teléfonos no son compatibles.

Proyector o Smartboard (opcional) conectado a una computadora para demostrar o modelar.

Requisitos de la Hora de Código: Si estás haciendo este ejercicio como parte de la Hora de Código, visita https://hourofcode.com para obtener instrucciones sobre cómo inscribir a tu clase para un evento de codificación. Los estudiantes pueden encontrar fácilmente a Karel en la página de Actividades al escribir "NCLab" en el menú de “Created by” (“Creado Por”).



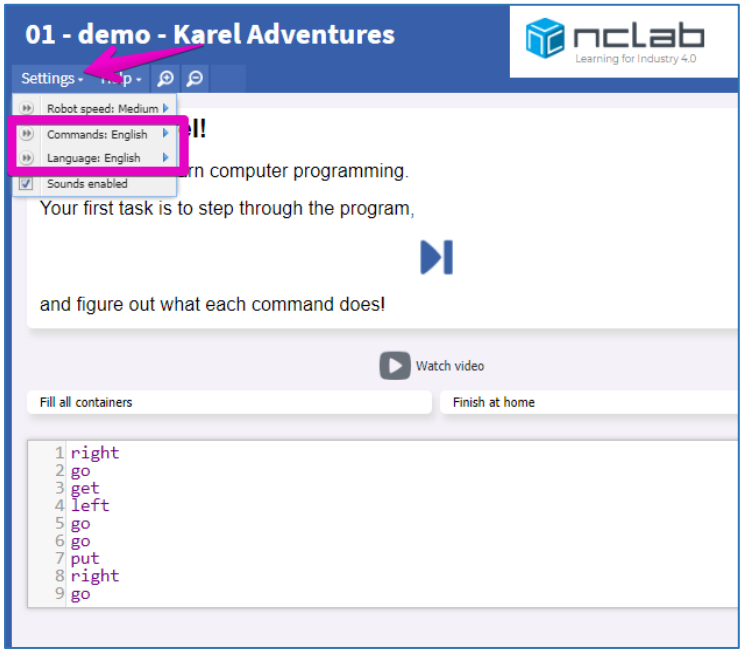
Videos de YouTube: Algunas escuelas bloquean YouTube. Es posible que los videos de demostración embebidos en los juegos necesiten ser desbloqueados por un administrador para que estén disponibles para los estudiantes.

Termina con un video o demostración de robótica: (opcional) Hay muchos videos geniales que demuestran la robótica. Usa términos de búsqueda como robots en cirugía, agricultura, militar, o manufactura avanzada para encontrar ejemplos actuales y compartir durante el resumen. Demuestra o haz que los estudiantes trabajen con robots o juguetes programables. Hay muchas opciones económicas en el mercado. Al ver a los robots en acción le ayudará a tus estudiantes a conectar el funcionamiento de Karel con los desafíos robóticos del mundo real.

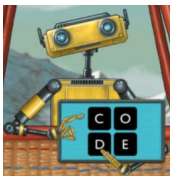
Imprimibles: (opcional) Los estudiantes pueden tomar notas a medida que avanzan en los juegos y luego usar esas notas para completar un boleto de salida. Los materiales imprimibles están adjunto al final de este plan de lección.

Certificados: Imprime certificados para los estudiantes que completen exitosamente los 15 juegos.

Configuración de Idioma Español: Después de abrir el primer juego, usa "Configuración" para cambiar el idioma. Puedes cambiar las instrucciones, los comandos, o ambos.



CONSEJOS PARA ENSEÑAR



- Pasa una hora o dos antes de tiempo probando los juegos tú mismo.
- Entendiendo los movimientos básicos:
  - `left` (`izquierda`) y `right` (`derecha`) son siempre desde el punto de vista de Karel. Estos son giros de 90 grados. Karel puede dar la vuelta usando dos comandos a la `izquierda` o dos a la `derecha` (180 grados).
- Karel está equipado con sensores, muy parecido a un robot real. Uno está debajo del robot. Los comandos `get` (`recoge`) y `put` (`pon`) usan este sensor.
  - `get` (`recoge`) recoge objetos coleccionables que están en el mismo cuadro que Karel.
  - `put` (`pon`) pone objetos en un contenedor que esté en el mismo cuadro que Karel.
- Karel también está equipado con un sensor que mira al frente, al igual que usamos nuestros ojos. Este sensor detecta obstáculos como bardas. Por ejemplo, en la condición:

  ```
  if wall     si barda
    left        izquierda
  ```

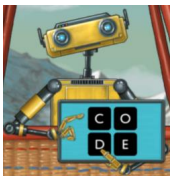
  Karel revisa el cuadro frente a él para ver si hay una barda, y si hay alguna presente, el girará a la izquierda.
- Las computadoras son la "policía de gramática" máxima. El lenguaje Karel simplifica la gramática (sintaxis) pero todavía tiene algunas reglas:
  - Sangría: las líneas que siguen a un `repeat` (`repite`), `if` (`si`) o un `while` (`mientras`) deben estar con sangría dos veces (cuatro espacios también funcionan). Las líneas verticales codificadas por colores hacen que la sangría sea más fácil de ver.
  - Ortografía: debe ser perfecta.
  - Mayúsculas: todas las palabras son minúsculas.
  - Los comandos en minúsculas que están correctamente escritos se convierten al color azul.
  - Puntuación: Karel no usa la puntuación para los comandos en estos ejercicios.

```
while not home
  go
  if rose or tulip
    get
  if water
    right
    if water
      repeat 2
        left
```

```
mientras no casa
  avanza
  si rosa o tulipán
    recoge
  si agua
    derecha
    si agua
      repite 2
        izquierda
```

- Cuanto más se pueda, dale a los estudiantes la libertad de averiguar por ellos mismos qué hacer. Anima a los alumnos a observar lo que hace Karel cuando ellos siguen los pasos del programa o cuando lo ejecutan.
- Decide con anticipación si tus estudiantes trabajarán solos, en pares o en equipos, o si ciertos estudiantes se beneficiarían al estar de compañeros con un estudiante mentor. Un método de pares efectivo es "Navegador/Piloto". El Piloto está sentado en la computadora y controla el mouse y el teclado. El Navegador se encuentra al lado del Piloto, lee todas las instrucciones y da orientación. Los estudiantes pueden cambiar sus papeles en el juego siguiente o en el mismo juego usando la computadora del otro estudiante, de modo que ambos estudiantes reciban certificados.

LISTA DE COMANDOS BÁSICOS Y PALABRAS CLAVES



Palabras de Comando:

`go` (`avanza`), `left` (`izquierda`), `right` (`derecha`), `get` (`recoge`), `put` (`pon`)

Comandos Direccionales [`go` (`avanza`), `left` (`izquierda`), `right` (`derecha`)] son siempre desde el punto de vista del robot.

`go` (`avanza`) avanza el robot un paso.

`left` (`izquierda`) gira el robot a su izquierda.

`right` (`derecha`) gira el robot a su derecha.

Recogiendo y colocando objetos [`get` (`recoge`), `put` (`pon`)]

`get` (`recoje`) recoge un objeto (Karel detecta objetos debajo de él, en el mismo cuadro)

`put` (`pon`) coloca un objeto (Karel detecta contenedores debajo de él)

Bucles (Loops)

`repeat x` (`repite x`), donde x = el número de veces que se repite el cuerpo del bucle (comandos con sangría).

`while x` (`mientras x`), donde x = una condición continua definida. El cuerpo del bucle se repetirá hasta que la condición continua sea falsa.

Condiciones

`if x` (`si x`), donde x = es una condición definida. Si la condición es verdadera, se ejecutarán los comandos con sangría. Si es falso, los comandos serán omitidos.

Comandos que son Operadores Lógicos

`not` (`no`) La condición es que el sensor `no` está presente.

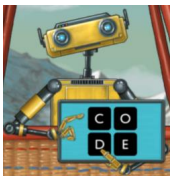
Palabras Importantes del Sensor

`north` (`norte`)
Karel está orientado al `norte`, o la parte superior del laberinto.

`wall` (`barda`)
Una barda es una clase de obstáculo. Karel detecta obstáculos que están un cuadro adelante de él. Karel evita los obstáculos dando vuelta. De lo contrario, Karel se estrellará.

`home` (`casa`)
El cuadro `casa`. Karel detecta cuando él está en ese cuadro (la casa está debajo de él).

VOCABULARIO



Código es una serie de instrucciones enviadas a la computadora. Por ejemplo, cuando escribes la palabra `go` (`avanza`), el programa de computadora sabe que debe mover a Karel un cuadro hacia adelante. El código es la forma en que se traducen las instrucciones de "humano" a "máquina". En estos ejercicios de laberinto, Karel recoge objetos, los coloca en contenedores y debe llegar a el cuadro casa sin encontrarse con ningún obstáculo. Todas estas acciones están controladas por código. Observa el comportamiento de Karel a medida que se ejecuta cada línea de código, utilizando el botón de Seguir.

Un programa de computadora es un algoritmo o serie de instrucciones escritas usando un lenguaje de programación. El lenguaje Karel de NCLab se basa en Python, un lenguaje utilizado en matemáticas, ingeniería, ciencias, programación "back-end", análisis de datos e inteligencia artificial.

Casa es el cuadro de destino, marcado por rayas diagonales rojas que cambian a verde después de que Karel haya cumplido con las metas del programa, como recogiendo todos los objetos. Es común usar la `casa` para finalizar un programa, y se usa en condiciones tales como "`while not home`" (`mientras no casa`).

Máx es el número máximo de pasos, operaciones o líneas de código.

Pasos son el número de cuadrados que Karel atraviesa. El icono del zapato cuenta el número de pasos.

Operaciones son todo lo que hace Karel: mover, girar, recoger o colocar objetos. El ícono de la computadora cuenta el número de operaciones.

Objetos son artículos colocados o recogidos en el laberinto. (La palabra "objeto" tiene otros significados en la programación que no se utilizan aquí).

Bucle (Loop) es una serie de comandos que se repiten un determinado número de veces (`repite`), o hasta que una condición pare (`mientras`).

Cuerpo: El cuerpo contiene los comandos que serán repetidos. Los comandos se escriben en las líneas siguiendo una sentencia de `repeat` (`repite`), `while` (`mientras`) or `def`. El cuerpo tiene sangría dos veces.

Algoritmo: Una serie de pasos lógicos que conducen a la solución de una tarea.

Error Lógico: Es un error en un algoritmo. La planificación ayuda a reducir el número de errores.

Sintaxis: La forma en que se escribe una línea de comando.

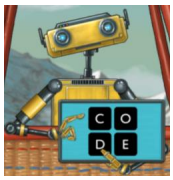
Error de Sintaxis: Un error en la ortografía, operadores, sangría, espacios

Palabras del Sensor: Palabras claves/comandos de la biblioteca Karel, puede incluir artículos coleccionables u objetos (como `orchid` [`orquídea`]), contenedores (como `basket` [`canasta`]), obstáculos (como `wall` [`barda`], `plant` [`planta`]) u otras condiciones (`north` [`norte`], `home` [`casa`]). Una palabra que esté tanto en la biblioteca como correctamente escrita estará coloreada de azul.

Condición: Las condiciones toman decisiones mientras el programa se está ejecutando, cuando no sabemos exactamente qué ocurrirá de antemano. Por ejemplo, Karel tal vez necesite recoger todas las monedas que encuentre, pero puede ser que no sepa dónde se ubicarán las monedas. La condición `if` (`si`) dice: "Si hay una moneda, recógela".

Comentarios: Los comentarios se utilizan para explicar lo que hace el código, escrito con #, seguido por la explicación. El símbolo de hashtag le dice al programa que ignore esa línea. Tú también puedes "comentar" una línea de código al probar tus programas, para que el programa salte esa línea.

INTRODUCIENDO LA LECCIÓN (5 MINUTOS)



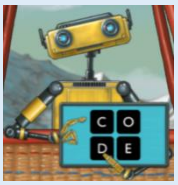
Hoy aprenderás a escribir código al jugar juegos con Karel el Robot.

Código es una serie de instrucciones enviadas a la computadora. Por ejemplo, cuando escribes la palabra `avanza` (`go`), el programa de computadora sabe que debe mover a Karel un cuadro hacia adelante. El código es la forma en que se traducen las instrucciones de "humano" a "máquina".
Una vez que aprendas algunas técnicas de programación, te sorprenderá al ver cuán pocas líneas de código se necesitan. En estos juegos de laberinto, Karel recoge objetos, los pone en contenedores y debe llegar a casa sin encontrarse con ningún obstáculo. Todas estas acciones están controladas por código. Usa el botón de Seguir para ver el comportamiento de Karel a medida que se ejecuta cada línea de código.

Un programa de computadora es un algoritmo, o serie de instrucciones, escritas usando un lenguaje de programación. El lenguaje Karel de NCLab se basa en Python, un lenguaje utilizado en matemáticas, ingeniería, ciencias, análisis de datos e inteligencia artificial.

Hay quince juegos en total. Con cada juego, aprenderás una nueva idea en codificación. La mayoría del programa ya está escrito: solo tienes que completar las líneas en blanco con el código que falta.

Tomadores de Notas: Si has elegido utilizar los tomadores de notas, distribuirlos en este momento. Los estudiantes pueden completar el tomador de notas a medida que pasan por los juegos o tomar las notas después de toda la sesión como una revisión.

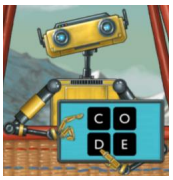


Si te atoras:

1. **Vuelve a leer las instrucciones.**
2. **Ejecuta a Karel a través del laberinto mentalmente. ¿Ves algún patrón?**
3. **Ejecuta el programa línea por línea usando el icono de paso.** 
4. **Lee el mensaje de error.**
5. **Revisa la lista de palabras que se requieren en el programa. ¿Las has usado todas?**
6. **Revisa el espacio, ortografía y sangría.**
7. **Usa el Menú de Ayuda en la parte superior de la pantalla para ver videos o leer el manual de referencia.**
8. **Consulta con tu compañero.**

Debes completar exitosamente cada juego para desbloquear el siguiente.
¡Buena suerte, y vamos a programar!

ACTIVIDAD KAREL EL ROBOT (40 MINUTOS)



Los estudiantes jugarán los juegos, progresando a su propio ritmo. Los estudiantes pueden trabajar individualmente, con un compañero, o en equipo según lo que determine el maestro/a.

Alienta y anima las habilidades de resolución de problemas y el progreso de los estudiantes; ayuda cuando sea necesario. Hazle saber a los estudiantes cuando les queden aproximadamente 5 minutos.

TERMINANDO (10 MINUTOS)

Hablen acerca de cómo les fue con los juegos. Si tienes videos de robótica o "robots" disponibles, muestra uno en este momento.

1. ¿Cómo se comparan las acciones de un robot del mundo real con Karel?
2. ¿Puedes ver Avanza, Izquierda, Derecha, Recoge y Pon?
3. ¿Ves patrones repetidos?
4. ¿Ves condiciones (si, mientras)?
5. ¿Ves rutinas que podrían ser escritas como funciones?

Hoy, participamos en la Hora de Código, junto con millones de otros programadores de todo el mundo. Las computadoras y los robots reales necesitan que los humanos escriban instrucciones para ellos, ya sea que se usen para diversión o para trabajar. Puedes escribir código para crear arte, construir un automóvil, probar medicamentos o diseñar un videojuego. La codificación se utiliza en todas partes.

EVALUACIÓN (5 MINUTOS)

Dale a cada estudiante un Boleto de Salida para que escriban algunas oraciones acerca de Karel y la codificación: lo que les agradó, lo que se les hizo desafiante y lo que podrían imaginar haciendo con sus habilidades de codificación.

DIFERENCIACIÓN

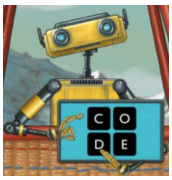
Apoyo:

- Modela el Juego 1 (demo) y el Juego 2 (requiere codificación), luego haz que los estudiantes comiencen los juegos.
- Si los estudiantes tienen dificultades para visualizar los movimientos y giros, inténtalo "en vivo" usando un suelo de baldosa, o una cuadrícula de cinta adhesiva en una alfombra.
- Haz que los estudiantes usen el botón de Seguir y hablen sobre cada paso. Por ejemplo:
  - "repite 3" significa que Karel repetirá los siguientes pasos tres veces
  - "si llave, recoge" significa que si hay una llave debajo de Karel, Karel recogerá la llave.
- Usa el apoyo de compañeros, como el método Navegador/Piloto.
- Recuérdales a los alumnos que ellos pueden reiniciar el juego e intentarlo de nuevo.
- Los estudiantes que hablen otro idioma pueden ejecutar a Karel en uno de varios idiomas.

Para los estudiantes que terminan los juegos temprano:

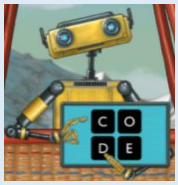
- Intenta una de las actividades de aprendizaje extendido en la página siguiente.
- Sé un "Experto" itinerante para ayudar a otros estudiantes.
- Explora otras actividades de la Hora de Código.

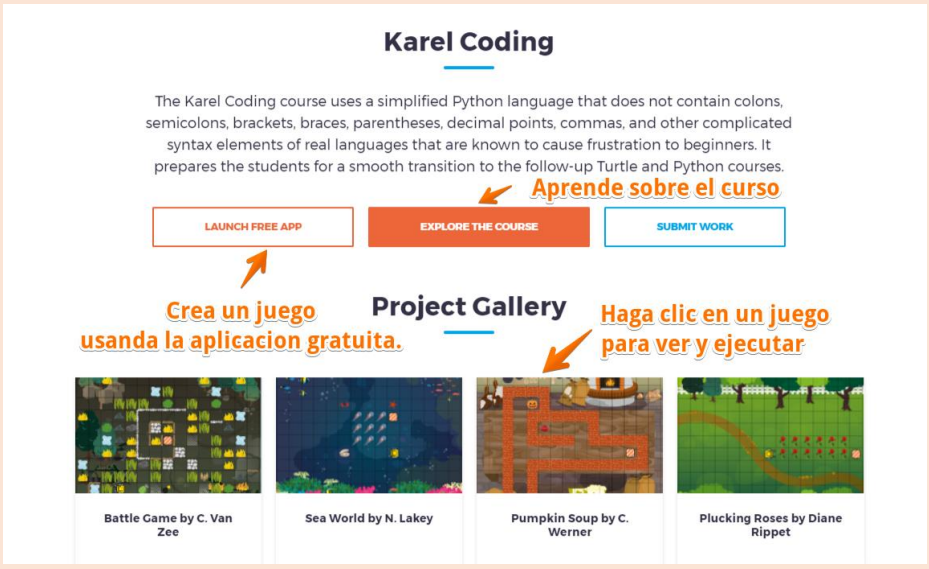
APRENDIZAJE EXTENDIDO



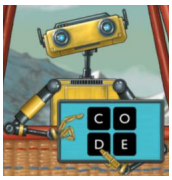
- Visita https://nclab.com/apps/ para ver y ejecutar juegos de Karel, modelos 3D y diseños de Python Turtle creados por estudiantes.
- Crea juegos usando la aplicación Karel.
- Aprende sobre el curso completo de Karel.
- Investiga carreras de programación en línea.
- Investiga la robótica en línea.
- Explora otras aplicaciones de la Hora de Código, como "¡Vamos a Construir Un Drone!" De NCLab

Para obtener más información sobre Karel El Robot, visita https://nclab.com/apps/





ESTÁNDARES (ESTADOS UNIDOS)



Estándares Comunes para las Prácticas Matemáticas:

- SMP 1: Dan sentido a los problemas y perseveran en su resolución.
- SMP 7: Reconocen y utilizan estructuras.
- SMP 8: Reconocen y expresan regularidad en el razonamiento repetitivo.

Estándares Comunes para las Artes del Lenguaje en Inglés:

- L1, L2: Demuestran dominio de las normativas de la ... gramática y uso, uso de mayúsculas, puntuación y ortografía al escribir. Las convenciones de codificación no son “negociables” y ayudan a moldear estos hábitos en los estudiantes.
- SL.x.1: Los estudiantes se involucran en las habilidades del hablar y escuchar al utilizar la técnica de Navegador/Piloto, la ayuda entre compañeros o al participar en discusión después de completar la actividad.
- W.10: Los estudiantes escriben respuestas como parte de una rutina de escritura.

Estándares de Ciencias para la Próxima Generación:

- SEP 2: Desarrollar y utilizar modelos. Los estudiantes están aprendiendo algoritmos que se utilizarán para construir sus propios programas.

Estándares de Ciencias de la Computación (CSTA 2017):

- 1B-AP-10: Crear programas que incluyen secuencias, eventos, bucles y condicionales. (P5.2)
- 1B-AP-11 Descomponer (separar en partes) los problemas en subproblemas más pequeños y manejables para facilitar el proceso de desarrollo del programa. (P3.2)

Nota: Hora de Código Karel no ha sido revisado con las organizaciones que supervisan el uso de estándares. Estas normas se presentan aquí sólo como referencia.

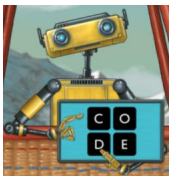
Citas:

Common Core: Autores: National Governors Association Center for Best Practices, Council of Chief State School Officers Título: Common Core State Standards (inserte el área de contenido específico si está utilizando solo uno) Editor: National Governors Association Center for Best Practices, Council of Chief State School Officers, Washington DC Fecha de Copyright: 2010

Next Generation Science Standards: Estados líderes de NGSS. 2013. Next Generation Science Standards: For States, By States. Washington, DC: The National Academies Press.

Computer Science Teachers Association (2017). CSTA K – 12 CSTA K–12 Computer Science Standards, Revised 2017. Obtenido de http://www.csteachers.org/standards.

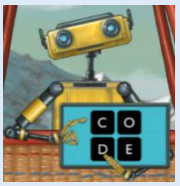
RECURSOS



Las siguientes páginas incluyen recursos imprimibles que puedes compartir con tus estudiantes.

1. Diapositivas compartibles con instrucciones
2. ¿Qué está Haciendo Karel? (explicación simplificada de los comandos)
3. Tomador de Notas de Karel
4. Boleto de Salida de Karel

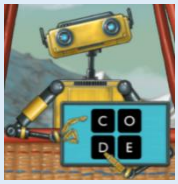
DIAPOSITIVA 1



Si te atoras:

1. **Vuelve a leer las instrucciones.**
2. **Ejecuta a Karel a través del laberinto mentalmente. ¿Ves algún patrón?**
3. **Ejecuta el programa línea por línea usando el icono de paso.** 
4. **Lee el mensaje de error.**
5. **Revisa la lista de palabras que se requieren en el programa. ¿Las has usado todas?**
6. **Revisa el espacio, ortografía y sangría.**
7. **Usa el Menú de Ayuda en la parte superior de la pantalla para ver videos o leer el manual de referencia.**
8. **Consulta con tu compañero.**

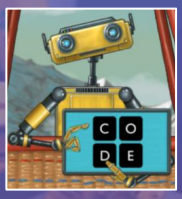
DIAPOSITIVA 2



Para obtener más información sobre Karel El Robot, visita https://nclab.com/apps/

¿QUÉ ESTÁ HACIENDO KAREL?

KAREL EL ROBOT: HORA DE CÓDIGO



¿QUÉ ESTÁ HACIENDO KAREL?

JUEGOS 1 - 3

COMANDOS BÁSICOS

`go (avanza)`: Hacer un paso adelante.
`left (izquierda)`: Girar 90 grados a la izquierda.
`right (derecha)`: Girar 90 grados a la derecha.
`get (recoge)`: Recoger un objeto en el cuadro de Karel.
`put (pon)`: Poner un objeto en un contenedor en el cuadro de Karel.

JUEGOS 4 – 6

REPEAT LOOPS (BUCLES REPITE)

El bucle `repeat` (`repite`) se usa cuando el número de repeticiones se sabe de antemano. El cuerpo tiene dos espacios con sangría.

```
repite 5      repeat 5
  avanza        go
```

Este bucle le dice a Karel que avance (`avanza`) cinco veces.
Los bucles de repetición pueden contener secuencias de comandos, condiciones, comandos definidos y otros bucles.

JUEGOS 7 - 9

CONDICIONES IF (SI)

La condición `if` (`si`) le dice a Karel que compruebe una condición. Si la condición es verdadera, Karel ejecutará los comandos escritos en sangría. Si es falso, Karel saltará a la siguiente parte del programa. El cuerpo tiene sangría dos veces.

```
si barda      if wall
  derecha       right
```

Si Karel detecta una barda, Karel girará a la derecha.

JUEGOS 10 - 12

WHILE LOOPS (BUCLES MIENTRAS)

El bucle `while` (`mientras`) debe usarse cuando no se sabe de antemano cuántas repeticiones serán necesarias. Termina cuando la condición es falsa. El cuerpo tiene sangría dos veces.

```
mientras no barda while not wall
  avanza            go
```

Este bucle le dice a Karel que siga avanzando (`avanza`), siempre y cuando Karel no sienta una barda (`no barda` es verdad). Si Karel detecta una barda (`no barda` es falso), entonces el bucle termina.
Bucles mientras pueden contener secuencias de comandos, condiciones, comandos definidos y otros bucles.

JUEGOS 13 -15

COMANDOS PERSONALIZADOS

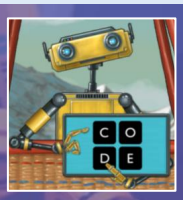
Los comandos personalizados se usan para definir una serie de instrucciones que se necesitan más de una vez en el programa. El cuerpo tiene sangría dos veces.

```
def giro
  repite 2
    izquierda
```

Ahora puedes utilizar `giro` en cualquier parte del programa escribiendo el comando `giro`.

TOMADOR DE NOTAS

KAREL EL ROBOT: HORA DE CÓDIGO



NOMBRE: _____

ESCRIBE NOTAS BREVES EN CADA CONJUNTO DE JUEGOS. ¿QUÉ APRENDISTE?

JUEGOS 1 (DEMO), 2, 3

JUEGOS 4 (DEMO), 5, 6

JUEGOS 7(DEMO), 8, 9

JUEGOS 10(DEMO), 11, 12

JUEGOS 13(DEMO), 14, 15

EXIT TICKET

KAREL EL ROBOT: HORA DE CÓDIGO

NOMBRE: _____

PUEDES ESCRIBIR CÓDIGO PARA CREAR ARTE, CONSTRUIR UN AUTOMÓVIL, PROBAR MEDICAMENTOS O DISEÑAR UN VIDEOJUEGO. ¡LA CODIFICACIÓN SE UTILIZA EN TODAS PARTES!

¿QUÉ PARTE DE KAREL EL ROBOT FUE LO MÁS AGRADABLE?

¿QUÉ PARTE DE KAREL EL ROBOT FUE LO MÁS DESAFIANTE?

SI FUERAS UN PROGRAMADOR, ¿QUÉ CODIFICARÍAS? AQUÍ HAY ALGUNAS IDEAS:

- Juegos, Simulaciones y Deportes
- Robots y Drones
- Arte y Música
- Diseño de Páginas Web
- Ingeniería y Construcción
- Big Data (Grandes Datos) y A.I. (Inteligencia Artificial)

¿POR QUÉ TE INTERESA ESA ÁREA? ¿QUÉ CREES QUE NECESITAS APRENDER A CONTINUACIÓN?