Código Pi
Nivel Secundario

Un estadio de fútbol a medida

APRENDER CONECTADOS

Ministerio de Educación,
Cultura, Ciencia y Tecnología
Presidencia de la Nación

Autoridades

Presidente de la Nación
Mauricio Macri

Jefe de Gabinete de Ministros
Marcos Peña

Ministro de Educación, Cultura, Ciencia y Tecnología
Alejandro Finocchiaro

Secretario de Gobierno de Cultura
Pablo Avelluto

Secretario de Gobierno de Ciencia, Tecnología e Innovación Productiva
Lino Barañao

Titular de la Unidad de Coordinación General del Ministerio de Educación, Cultura, Ciencia y Tecnología
Manuel Vidal

Secretaria de Innovación y Calidad Educativa
Mercedes Miguel

Directora Nacional de Innovación Educativa
María Florencia Ripani

ISBN en trámite

Este material fue producido por el Ministerio de Educación, Cultura, Ciencia y Tecnología de la Nación en el marco del Plan Aprender Conectados.

Índice

Ficha técnica

Nivel educativo	Educación Secundaria.
Año	2do/3ero.
Área del conocimiento	Matemática.
Tema de la clase	Generar secuencias de programación con ciclos de repetición que incluyan escalas y patrones para diseñar espacios sobre ejes x, y, z.
NAP de matemática relacionados	El reconocimiento, uso y análisis de funciones en situaciones problemáticas que requieran: • interpretar gráficos y fórmulas que modelicen variaciones lineales y no lineales (incluyendo la función cuadrática) en función de la situación; • modelizar y analizar variaciones lineales expresadas mediante gráficos y/o fórmulas, interpretando sus parámetros (la pendiente como cociente de incrementos y las intersecciones con los ejes).

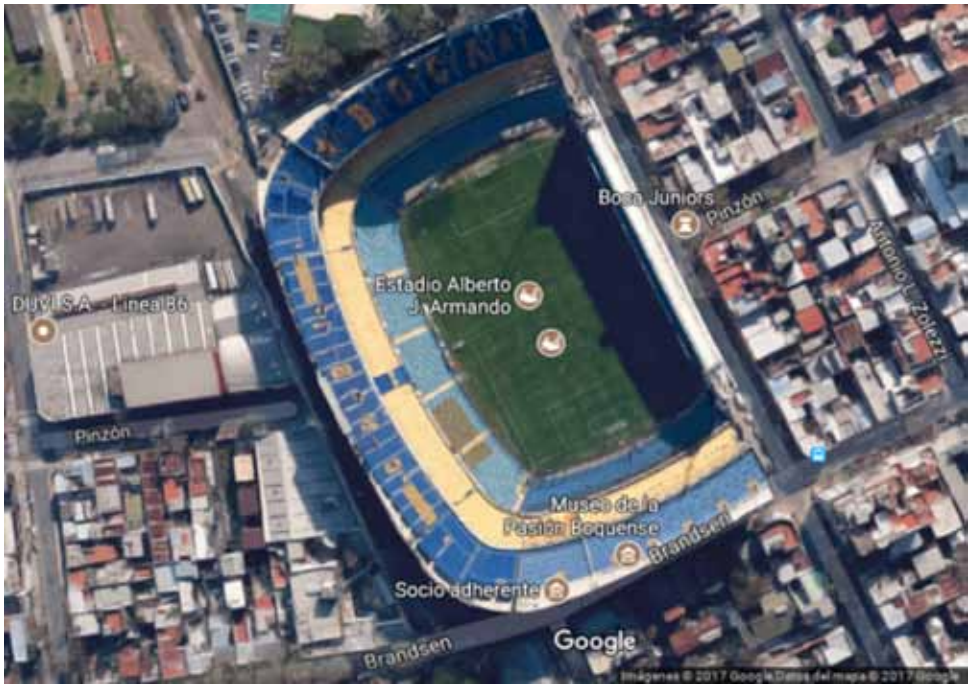
Habilidad de Programación y robótica relacionada:	• Desarrollar proyectos creativos que involucren la selección y el uso de programas para solucionar problemas del mundo real, incluyendo el uso de uno o más dispositivos y la aplicación, redacción y análisis de información; • resolver problemas a partir de su descomposición en partes pequeñas y aplicando diferentes estrategias, utilizando entornos de programación tanto textuales como icónicos, con distintos propósitos, incluyendo el control, la automatización y la simulación de sistemas físicos.
Duración	2 clases.
Materiales	Una computadora del eje de implementación de Código Pi, por grupo.
Desafíos pedagógicos	• Modelizar objetos en tres dimensiones a partir del posicionamiento espacial con ejes de coordenadas. • Conocer la dinámica de la programación de Minecraft Pi. • Conceptualizar qué es un bucle y una variable a partir de su inclusión en un código de programación.
Resumen de la actividad	A partir de la observación de distintos estadios de fútbol, se propone a los estudiantes la realización de uno propio en tres dimensiones utilizando el programa Minecraft Pi. Aplicarán variables y bucles de repetición, reconocerán cómo se marcan posiciones en los ejes de coordenadas X, Y, Z. Establecerán patrones y relaciones de escala..

Inicio

Si hay algo que enorgullece a la hinchada de fútbol, es que su club aparezca en los diferentes rankings mundiales.

La revista FourFourTwo, especializada en este popular deporte, realizó un ranking de los 100 mejores estadios de fútbol del mundo.

El ranking comienza con... ¡La bombonera!



https://www.google.com.ar/maps/@-34.6357206,-58.3644041,285m/data=!3m1!1e3?hl=es-419

El estadio tiene forma de “D” porque “originalmente el diseño abarcaba el doble del espacio que finalmente pudo ser utilizado. La razón de su diseño compacto fue que se debía construir el nuevo estadio en el mismo solar donde se encontraba el anterior, de madera y mucho más pequeño”[1]

[1] https://es.wikipedia.org/wiki/Estadio_Alberto_J._Armando

El segundo en el ranking es el Camp Nou, de Barcelona, España:

https://www.google.com.ar/maps/place/Collblanc-Cardenal+Reig/@41.3809694,2.1214582,370m/data=!3m1!1e3!4m5!3m4!1s0x12a498fa1149c1bf:0xa4ed9a7cb7f1e6fe!8m2!3d41.375919!4d2.114639?hl=es-419

Y el tercero, el estadio de Wembley, Londres, Inglaterra:

https://www.google.com.ar/maps/place/Estadio+de+Wembley/@51.5560739,-0.2819214,431m/data=!3m1!1e3!4m5!3m4!1s0x48761181d57a876d:0xa64f9f185de8e097!8m2!3d51.5560208!4d-0.2795188

2. Desarrollo

Más allá del ranking, es interesante analizar cómo están diseñados los estadios de fútbol.

¿Qué diferencias encontraron entre los ejemplos anteriores?

Utilizando Minecraft Pi en la computadora del eje de implementación de Código Pi, es posible crear un estadio.
No tenemos limitaciones en cuanto a los costos: podemos gastar infinita cantidad de materiales. Sin embargo, tenemos limitaciones en las medidas:

- ¿Qué factor limita el tamaño y la capacidad de un estadio?
- ¿Por qué piensan que los estadios de fútbol suelen construirse en las afueras de las ciudades?

¿Cuánto mide una cancha de fútbol profesional?

Existe una reglamentación que indica exactamente la medida para que una cancha de fútbol sea considerada profesional.
Estas son las medidas:

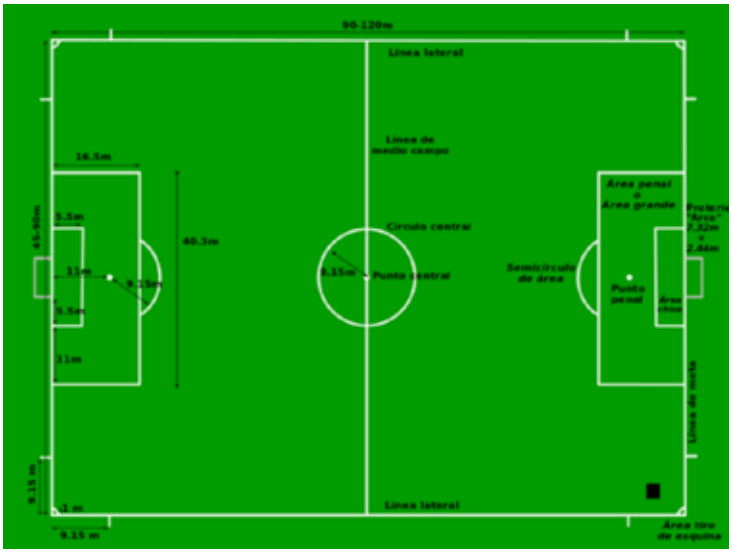


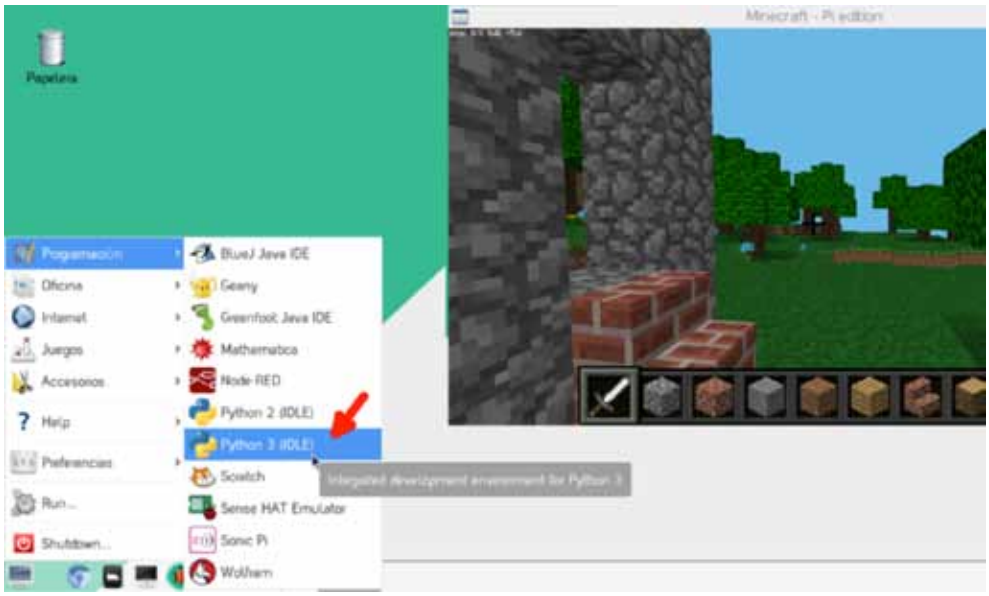
Imagen: By https://commons.wikimedia.org/wiki/User:Ffahm [CC BY-SA 3.0 (http://creativecommons.org/licenses/by-sa/3.0)], via Wikimedia Commons

Como primer paso para la creación del estadio, deberán decidir cuál será la escala que utilizarán en Minecraft Pi para luego utilizar este patrón en cálculos de medidas. Por ejemplo, se pueden utilizar 2 cubos (uno al lado del otro) para determinar la ocupación de 1 metro. O 10 cubos para referenciar a 1 metro de ocupación real y así siguiendo con demás ejemplos.
En la siguiente tabla encontrarán 3 escalas modelo. Tendrán que realizar todos los cálculos correspondientes para determinar cuántos cubos ocupará cada sector de la cancha.

Vuelquen en esta tabla los cálculos realizados:

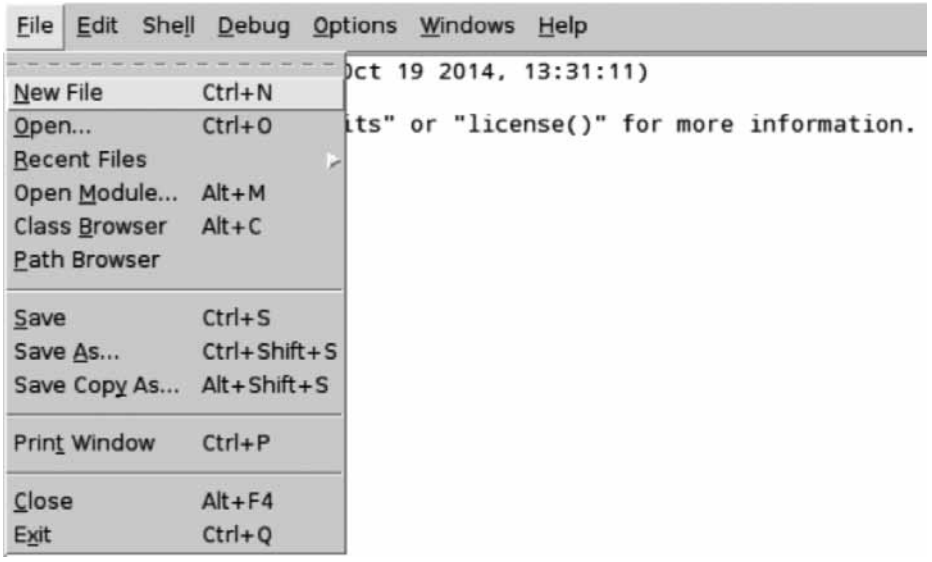
Medida original	**Escala A: 2 cubos = 1 metro**	**Escala B: 1 cubo = 3 metros**	**Escala C: 1 cubo = 2 metros**
Largo: 90 metros	180 cubos de largo	...	...
Ancho:			
Arco:			
Área chica:			
Área grande:			
...			

El primer paso será acceder a **Minecraft Pi**.
Después, es necesario hacer clic en "**Start Game**" para comenzar y elegir la opción "continuar el mismo mundo" o "**crear uno nuevo**".
Con la tecla TAB se devuelve el control al mouse para poder entrar en el lenguaje de programación desde el Menú:



Arrastren la ventana desde la barra de título para poder ver las dos a la vez.

Comiencen a trabajar en un archivo nuevo: en Python, elegimos "**New file**" en el menú "**File**", o pulsando ***CTRL+N***:



Para que este lenguaje de programación entienda las órdenes específicas de Minecraft Pi, comenzamos importando la librería (conjunto de órdenes que se agregan a un lenguaje de programación) correspondiente:

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
```

Ahora, pueden utilizar las distintas órdenes que fueron aprendiendo:

- Para cambiar de lugar a Steve, utilicen **mc.player.setPos()**, con números fijos. Por ejemplo:

```
mc.player.setPos(5, 10, 6)
```

- Para sumar un bloque al lado de donde están posicionados:

```
x, y, z = mc.player.getPos()
mc.setBlock(x+1, y, z, 1)
```

- Para agregar un bloque de bloques:

```
x, y, z = mc.player.getPos()
mc.setBlocks(x+1,y, z+1, x+4, y+3, z+4,1)
```

❑ ¿Cómo se podría hacer un muro utilizando este último comando? ¿Y un piso plano?

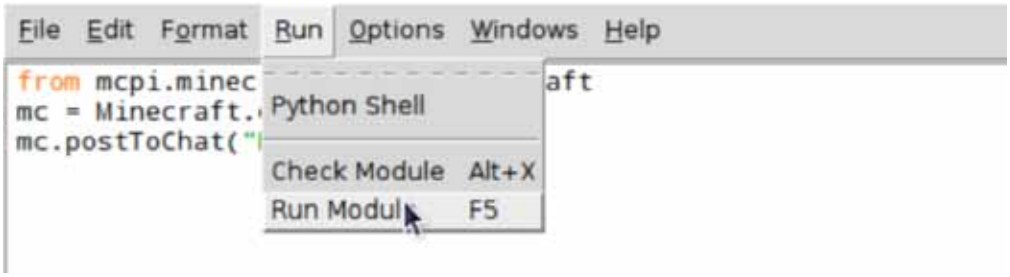
❑ Entre los materiales que se pueden utilizar hay uno que es para poner “aire”. ¿Para qué sería útil?

Contamos con cuatro materiales para ir probando:

0 - Aire 1 - Piedra 2 - Pasto 3 - Tierra

❑ Prueben otros materiales que no están en la lista ¿Todos se comportan de la misma manera con respecto a la gravedad?

Para ejecutar el programa es posible utilizar la combinación de teclas ***fn + F5***.



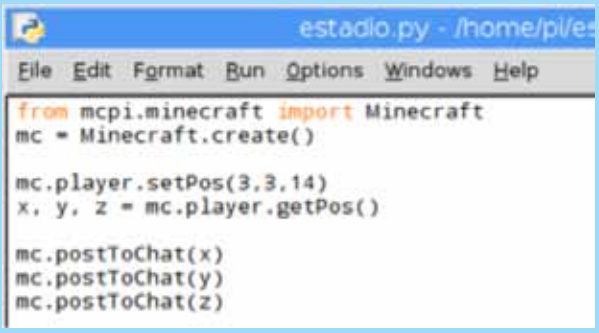
Comienza la construcción

¿Qué sería lo primero a tener en cuenta para construir un estadio?
Claramente, la base de todo es la cancha de fútbol. Ese aspecto definirá el inicio: ¿Dónde queremos que esté la cancha? ¿Subterránea? ¿Sobre la superficie? ¿Volando? ¿En el agua?

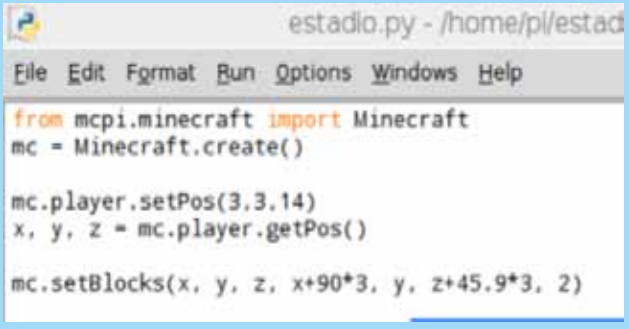
Comiencen, entonces, ubicando al personaje en lo que será un extremo de la cancha de fútbol.
Para ello, observen la pantalla de Minecraft Pi, donde nos indica la posición actual, por ejemplo:

Pos: 2.3, 0.0, 13.3

Desplacen al personaje hasta el lugar donde debería comenzar la zona de césped, o utilicen la orden setPos.



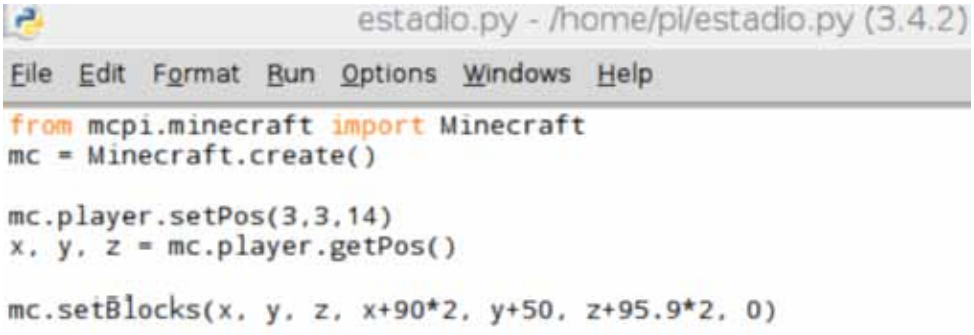
Ahora vamos a crear el césped de la cancha de fútbol. Para ello, utilicen “setBlocks” y dejen intacta la variable Y para construir solo sobre el plano horizontal:



❑En el primer intento, ¿lograron obtener exactamente lo que imaginaban?

❑Si observan el entorno, seguramente aparecerán árboles y elevaciones. ¿De qué manera crearían un “hueco” sobre la superficie del tamaño de una cancha de fútbol?

❑Analicen el siguiente código. ¿Qué crees que hace?:



Agregando comentarios al código

Probablemente quieras hacer pruebas antes de obtener el diseño definitivo.
Para generar renglones “de prueba”, y saber rápidamente a lo que refieren podemos utilizar comentarios. Los comentarios sirven también para explicar el código. Son utilizandos para obtener una lectura más clara del mismo.
Para crear un comentario, tendrán que comenzar el renglón con un signo #
En el siguiente código los renglones de color rojo no se están ejecutando. Cuando la computadora encuentra un renglón que comienza con # lo ignora. Veamos el siguiente ejemplo de utilización de comentarios:

```
File Edit Format Run Options Windows Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

mc.player.setPos(3,3,14)
x, y, z = mc.player.getPos()

#El siguiente renglón sirve para borrar lo que haya en la zona
mc.setBlocks(x, y, z, x+90*2, y+50, z+95.9*2, 0)

#El siguiente renglón sirve para crear el césped
mc.setBlocks(x, y, z, x+90, y, z+45.9, 2)
```

Creando las líneas blancas

Para crear una línea necesitamos colocar una fila de bloques.
Podemos crear, por ejemplo, un bloque en x, otro en x+1, otro en x+2, otro en x+3... y así sucesivamente hasta el límite de la cancha.

¿Cómo podemos realizar esto en unas pocas líneas? Utilizando un **bucle**[2].
Mediante un bucle podemos decirle a MinecraftPi -por ejemplo- que coloque un bloque desde x hasta x+90

Para aprender a utilizar un bucle, vamos a ensayar lo siguiente. Le vamos a pedir a la computadora que escriba los números del 1 al 10 en la pantalla, con este código:

```
numerodeprueba = 1
for numerodeprueba in range(1, 10):
    mc.postToChat(numerodeprueba)
numerodeprueba = numerodeprueba+1
```

Ahora vamos a agregarle que escriba algo más:

```
numerodeprueba = 1
for numerodeprueba in range(1, 10):
    mc.postToChat("El valor de la variable ahora es: ")
    mc.postToChat(numerodeprueba)
numerodeprueba = numerodeprueba+1
```

También podemos mostrar el doble de la variable:

```
numerodeprueba = 1
for numerodeprueba in range(1, 10):
    mc.postToChat("El doble del valor actual de la variable es: ")
    mc.postToChat(numerodeprueba*2)
numerodeprueba = numerodeprueba+1
```

Para hacer la línea blanca empezamos por crear una variable[3] donde almacenar el primer valor de x, pero sin decimales; para eso sirve la función **int**, que viene de "**integer**" ("**entero**")

```
posicionx=int(x)
```

Después le decimos que repita una acción de acuerdo a lo que valga x, comenzando en "posicionx" y terminando en "posicionx+90" (por ejemplo).
¿Y cuál es la acción que debe repetir?

- Colocar un bloque en x, y, z de tipo 80 (nieve)
- Aumentar el valor de x

---

[2] Un **bucle** o ciclo, en **programación**, es una sentencia que ejecuta repetidas veces un trozo de código, hasta que la condición asignada a dicho bucle deja de cumplirse.
[3] Una **variable** es un espacio de la memoria en la computadora a la que asignamos un contenido que puede ser un valor numérico (sólo números, con su valor de cálculo) o alfanumérico (sólo texto o texto con números).

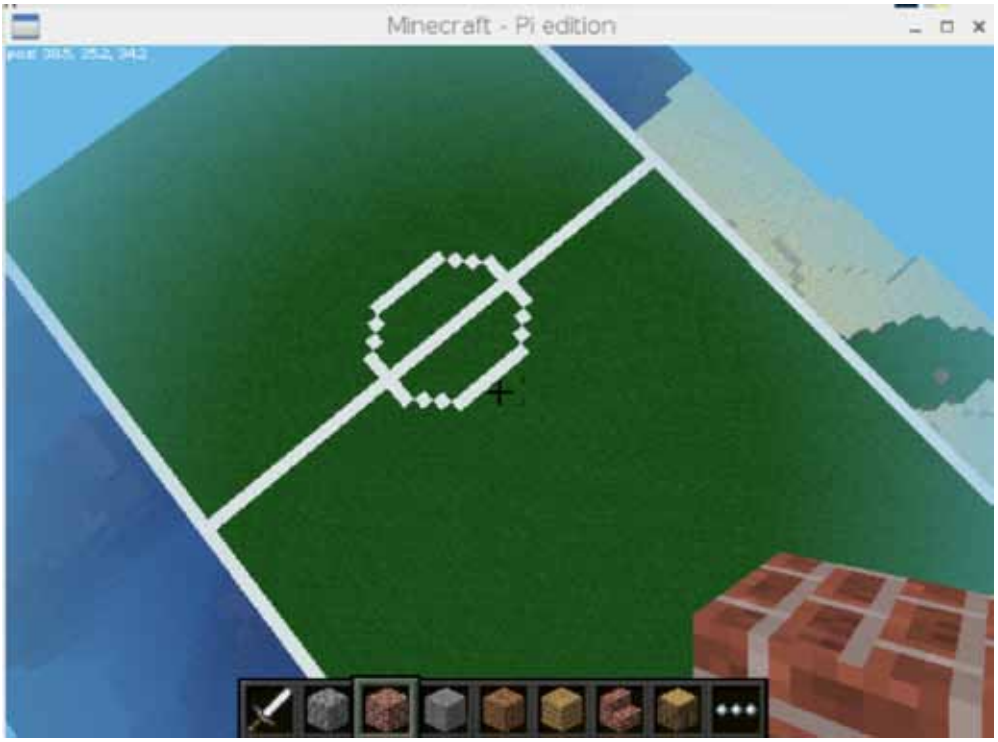
Entonces, el bucle que utilizaremos para trazar una línea blanca en uno de los lados de la cancha será:

```
for x in range(posicionx, posicionx+90):
    mc.setBlock(x, y, z, 80)
x=x+1
```

Con el siguiente código, pueden visualizar en pantalla los valores de x:

```
for x in range(posicionx, posicionx+90):
        mc.postToChat(x)
        mc.setBlock(x, y, z, 80)
    x=x+1
```

- ❑¿Cómo haríamos la otra línea blanca, paralela a la que ya trazamos?
- ❑Y las líneas blancas perpendiculares?
- ❑¿Cómo podría resolverse el círculo central?



Aquí les mostramos un ejemplo de código completo para inspirarse y luego modificarlo:

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

mc.player.setPos(3,3,14)
x, y, z = mc.player.getPos()

#El siguiente renglón sirve para borrar
mc.setBlocks(x, y, z, x+90*2, y+50, z+95.9*2, 0)

#Ahora creamos el césped
mc.setBlocks(x, y, z, x+90, y, z+45.9, 2)

#Trazamos las líneas blancas
posicionx=int(x)
for x in range(posicionx, posicionx+90):
    mc.postToChat(x)
    mc.setBlock(x, y, z, 80)
    mc.setBlock(x, y, z+45.9, 80)
x=x+1
```

```
#reiniciamos los valores de las variables
x, y, z = mc.player.getPos()

#Ahora creamos las líneas blancas perpendiculares
posicionz=int(z)
for z in range(posicionz, posicionz+46):
    mc.setBlock(x, y, z, 80)
    mc.setBlock(x+90, y, z, 80)
    mc.setBlock(x+45, y, z, 80)
z=z+1

#reiniciamos los valores de las variables
x, y, z = mc.player.getPos()

#Por último, agregamos el círculo del centro
mc.setBlock(x+45+5, y, z+23, 80)

mc.setBlock(x+45+5, y, z+23-1, 80)
mc.setBlock(x+45+5, y, z+23+1, 80)

mc.setBlock(x+45+5, y, z+23-2, 80)
mc.setBlock(x+45+5, y, z+23+2, 80)

mc.setBlock(x+45+5, y, z+23-3, 80)
mc.setBlock(x+45+5, y, z+23+3, 80)

mc.setBlock(x+45+4, y, z+23-4, 80)
mc.setBlock(x+45+4, y, z+23+4, 80)

mc.setBlock(x+45+3, y, z+23-5, 80)
mc.setBlock(x+45+3, y, z+23+5, 80)

mc.setBlock(x+45+2, y, z+23-6, 80)
mc.setBlock(x+45+2, y, z+23+6, 80)

mc.setBlock(x+45+1, y, z+23-6, 80)
mc.setBlock(x+45+1, y, z+23+6, 80)

mc.setBlock(x+45-1, y, z+23-6, 80)
mc.setBlock(x+45-1, y, z+23+6, 80)


mc.setBlock(x+45-2, y, z+23-6, 80)
mc.setBlock(x+45-2, y, z+23+6, 80)


mc.setBlock(x+45-3, y, z+23-5, 80)
mc.setBlock(x+45-3, y, z+23+5, 80)
```

```
mc.setBlock(x+45-4, y, z+23-4, 80)
mc.setBlock(x+45-4, y, z+23+4, 80)

mc.setBlock(x+45-5, y, z+23-3, 80)
mc.setBlock(x+45-5, y, z+23+3, 80)

mc.setBlock(x+45-5, y, z+23-2, 80)
mc.setBlock(x+45-5, y, z+23+2, 80)


mc.setBlock(x+45-5, y, z+23-1, 80)
mc.setBlock(x+45-5, y, z+23+1, 80)

mc.setBlock(x+45-5, y, z+23, 80)
mc.setBlock(x+45-5, y, z+23, 80)
```

3. Cierre

Hoy avanzamos muchos pasos en el aprendizaje de la programación de Minecraft Pi. Aprendimos a utilizar diferentes materiales, y fundamentalmente, comenzamos a utilizar **bucles de repetición**.
Los bucles de repetición son estructuras que optimizan la escritura de código, podemos simplificar escritura con la utilización de estas estructuras.
Con lo aprendido:

- ❑ ¿Se animan a hacer las gradas?
- ❑ ¿Y a decorar el estadio?