



Girls Who Code en casa

Generador de contraseñas

Descripción de la actividad

Usamos contraseñas para casi todo lo que hacemos digitalmente. Quizá usted tenga una contraseña para su correo electrónico, sus sitios web de compras, Instagram, las aplicaciones de entrega de alimentos, etc. ¿Sabía que para asegurar su información personal, como su nombre, la fecha de nacimiento o su ubicación, usted debería crear una contraseña distinta para cada una de sus cuentas? En 2019, aproximadamente 1 de cada 15 personas fue víctima de [fraude de identidad](#), y uno de los motivos más frecuentes de los ataques fue una contraseña poco segura.

Los especialistas en seguridad cibernética se encargan de implementar medidas de seguridad en cualquier computadora y de reconocer las posibles amenazas. [La seguridad cibernética](#) es un sector de rápido crecimiento que representa entre el 32 % y el 45 % de todos los puestos de trabajo en tecnología de Estados Unidos, con un salario básico promedio de **\$83,000**. Hay muchas cosas que puede hacer en casa para proteger los sistemas informáticos y evitar un ataque cibernético. En esta actividad, usted generará contraseñas de 10 caracteres de longitud de manera aleatoria que sean lo suficientemente seguras como para evitar ataques cibernéticos a su información personal.

Si quiere obtener más información sobre seguridad cibernética, consulte esta actividad: [Detective cibernética](#).

Objetivos del aprendizaje

Al finalizar esta actividad, será capaz de:

- comprender la importancia de crear contraseñas seguras para proteger la información personal;
- crear y usar [listas](#) en Python para almacenar información;
- usar la biblioteca [aleatoria](#) para aleatorizar números dentro de un intervalo especificado.

Materiales

- [Editor Repl.it](#)
- [Proyecto de muestra del generador de contraseñas](#)
- [Guía de referencia del generador de contraseñas](#)

Conocimientos previos

Antes de comenzar este proyecto, le recomendamos que:

- pueda explicar con sus propias palabras qué es una [variable](#) y describir cómo pueden usarse en un programa.
- pueda explicar con sus propias palabras qué es un [enunciado condicional](#) y describir cómo pueden usarse en un programa.

Si desea hacer un repaso rápido de Python, le recomendamos que consulte esta actividad: [¿Puedo ayudarla?](#)

“Mujeres en tecnología” artículo destacado: Jaya Baloo



Fuente de la imagen:
[Columbia Journalism Review](#)

Violet Blue es periodista, autora, asesora y educadora sobre piratería y delitos cibernéticos. Violet llegó a la conclusión de que las mujeres y la comunidad [LGBTQIA](#) tienen un riesgo mayor de ser víctimas de delitos cibernéticos. Los hackers usan o venden información personal y dejan a estas personas vulnerables para que extraños las [troleen](#), suplanten su identidad o las acosen físicamente. Violet sigue usando su plataforma en línea para generar conciencia sobre los delitos cibernéticos, educar a otras personas sobre cómo proteger la información y asesorar a empresas e individuos sobre medidas de seguridad cibernética.

En 2015, Violet escribió un libro, [The Smart Girl's Guide to Privacy](#) (Guía sobre la privacidad para muchachas inteligentes). Este libro brinda información a sus lectoras respecto de cómo hacer lo siguiente:

- Eliminar el contenido personal de los sitios web.
- Usar eficazmente los controles de privacidad de los sitios web y de los navegadores.
- Recuperar y prevenir el robo de identidad.
- Averiguar en qué la ley protege a las mujeres y en qué no.
- Crear perfiles en línea seguros.
- Irse de sitios web donde se buscan a las personas.

Mire este [video sobre la opinión de Violet en lo que respecta a la cultura de la piratería](#) (hasta el minuto 3:20). ¿Desea obtener más información sobre Violet? Lea este [artículo con sugerencias para otras mujeres periodistas](#) sobre cómo protegerse de los delitos cibernéticos. [Obtenga más información sobre cómo usar los administradores de contraseñas](#) o [cómo usar la VPN](#), o la red privada virtual, para mantener la privacidad de sus datos mientras navegar por Internet.

Reflexión

Ser un experto informático es más que sencillamente ser bueno programando. Tómese algunos minutos para reflexionar sobre cómo Violet y su trabajo reflejan las características que todos los verdaderos expertos informáticos deben desarrollar en sí mismos: valentía, resiliencia, creatividad y propósito.

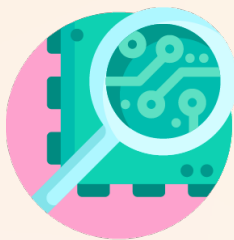


CREATIVIDAD

Violet sigue escribiendo artículos y educando a otras personas sobre la seguridad cibernética a través de sus plataformas en línea. Si usted tuviera su propia plataforma, ¿sobre qué temas brindaría información a otras personas?

Comparta sus respuestas con un familiar o amigo. Anime a los demás para que lean más sobre Violet y se unan a la charla.

Paso 1: ¿Qué es la seguridad cibernética? (de 2 a 5 minutos)



La **seguridad cibernética** hace referencia a la práctica de defender las computadoras (cualquier dispositivo electrónico que pueda almacenar y procesar datos), los servidores, las redes y los datos en general de los ataques cibernéticos. Ahora que casi todos los tipos de dispositivos electrónicos se conectan a Internet a través de **wifi** o **5G**, proteger la información de ataques cibernéticos es más importante que nunca. Quizá ya esté familiarizada con algunos ataques famosos, que van desde filtraciones de información sobre la privacidad de las celebridades hasta la publicación de información confidencial de los gobiernos o los programas de delincuencia en la televisión; o tal vez conozca personalmente a alguien que haya sufrido un ataque cibernético.

En 2019, aproximadamente 1 de cada 15 personas fue víctima de un **fraude de identidad**. Uno de los ataques más frecuentes se debe a la falta de contraseñas fuertes y seguras de cualquier cuenta en línea. En esta actividad, analizaremos las cualidades y la importancia de una contraseña segura, y luego aprenderá a codificar un generador de contraseñas en Python.

Contraseñas fuertes o seguras (2 minutos)

Si un hacker ha podido acceder a su cuenta, lo más probable es que se deba a que usted tiene una **contraseña poco segura o vulnerable**. Cuando se crea una contraseña, suele haber algunos requisitos sencillos para que la cuenta sea válida. Pueden aplicarse algunas restricciones:

- un mínimo de 6 a 8 caracteres;
- una variedad de letras mayúsculas y minúsculas;
- al menos un número;
- al menos un carácter especial.

Observe el gráfico de la derecha y compare el tiempo que tardan los hackers en descifrar su contraseña.

TIME IT TAKES FOR A HACKER TO CRACK YOUR PASSWORD					
Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	1 sec	5 secs
7	Instantly	Instantly	25 secs	1 min	6 mins
8	Instantly	5 secs	22 mins	1 hour	8 hours
9	Instantly	2 mins	19 hours	3 days	3 weeks
10	Instantly	58 mins	1 month	7 months	5 years
11	2 secs	1 day	5 years	41 years	400 years
12	25 secs	3 weeks	300 years	2k years	34k years
13	4 mins	1 year	16k years	100k years	2m years
14	41 mins	51 years	800k years	9m years	200m years
15	6 hours	1k years	43m years	600m years	15 bn years
16	2 days	34k years	2bn years	37bn years	1tn years
17	4 weeks	800k years	100bn years	2tn years	93tn years
18	9 months	23m years	6tn years	100 tn years	7qd years

 **HIVE SYSTEMS**

Cybersecurity that's approachable.
Find out more at hivesystems.io

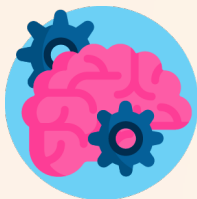
Fuente de la imagen: [Reddit](#)

A menudo, las personas usan la misma contraseña para todas sus cuentas en línea, desde Facebook y Apple hasta sus cuentas bancarias. Si bien esto facilita que las personas recuerden sus contraseñas, también las hace más vulnerables a los ataques cibernéticos.

Para proteger sus cuentas, lo mejor es crear contraseñas únicas con diferentes letras, números y caracteres especiales. Pero, como puede imaginar, esto es sumamente difícil de recordar. ¡Para eso están los administradores de contraseñas! Este software de seguridad ayuda a las personas a crear y recordar contraseñas seguras de todas las cuentas; todo lo que usted debe hacer es recordar una contraseña.

Paso 2: Planificar su contraseña (de 15 a 20 minutos)

¡A explorar! (de 10 a 15 minutos)



Para este proyecto, usaremos el editor web [Repl.it](https://repl.it). Repl.it es un editor gratuito, colaborativo y basado en el navegador que admite múltiples lenguajes de programación. Esta potente herramienta le permite incluso codificar y hablar con un grupo de amigos al mismo tiempo.

- ❑ **Clasifique las siguientes contraseñas del 1 al 5, donde 1 representa la contraseña más segura y 5 representa la contraseña menos segura.** Para ayudar con la clasificación de las contraseñas, incluimos algunas columnas: cantidad de caracteres y variación de caracteres.
 - ❑ **Complete la columna con la cantidad de caracteres y la variación de caracteres para cada contraseña.**
 - ❑ **Clasifique las siguientes contraseñas del 1 al 5.**
 - ❑ **Verifique sus respuestas con [este sitio web](#) y escriba cada opción.** Complete la última columna; las horas que demorará un hacker en descifrar la contraseña, para comparar sus clasificaciones.

Clasificar 1: más segura; 5: menos segura	Contraseña	Cantidad de caracteres ¿Cuántos caracteres tiene la contraseña?	Variación de caracteres ¿La contraseña incluye letras mayúsculas o letras minúsculas? ¿Números? ¿Símbolos?	Horas para que el hacker descifre la contraseña Según _____, ¿cuánto tiempo tardaría un hacker en descifrar la contraseña?
Contraseña de ejemplo	ku8@}:'\$	8	Letras minúsculas, símbolos y números	4 horas
	hcVESx			
	vWESp3Tt			
	Sg3Jpezyhv			
	password1			
	jG/8ab{s			



Recuerde verificar sus ideas con la Guía de referencia (página 2).

Paso 2: Planificar su contraseña (continuación)

- ❑ Tómese **2 minutos** para idear sus propias contraseñas seguras y escríbalas en el espacio a continuación. Los hackers deberían tardar *al menos un año* en descifrar las contraseñas.
- ❑ Navegue hasta [este sitio web](#) que usamos en el paso anterior para verificar cuán seguras son las contraseñas.

Crear el patrón de su generador de contraseñas (de 2 a 3 minutos)

En esta actividad, creará un proyecto que genere aleatoriamente una contraseña de 10 caracteres de longitud. Usted deberá decidir el *orden de los caracteres* de la contraseña.

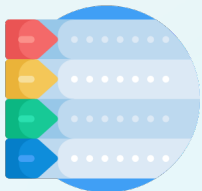
Ejemplo

En este proyecto de ejemplo, todas las contraseñas generadas tendrán el siguiente patrón:

- **Carácter 0 al carácter 5:** una letra aleatoria, ya sea mayúscula o minúscula
- **Carácter 6:** un número aleatorio
- **Carácter 7:** un carácter especial aleatorio
- **Carácter 8 al carácter 9:** una letra aleatoria, ya sea mayúscula o minúscula

```
q1Qe1B7-En  
ihhniK2&kR  
iGVRNz3^Hb  
oYJJC2+s0  
bTedeQ7&Rm
```

Estos son cinco ejemplos de una contraseña generada aleatoriamente que emplea el mismo patrón mencionado arriba.



Quizá piense: ¿por qué empezamos con el 0? De hecho, para crear la contraseña necesitaremos un objeto denominado **lista** de Python. Una **lista** es un conjunto de datos ordenados que se usa para almacenar diferentes tipos de información. Las **listas** usan índices o un número para hacer referencia a cada elemento en su interior. Esto facilita la recuperación, la adición, la eliminación y la modificación de la información dentro de la **lista**. Los índices comienzan en 0 y se incrementan en 1; por eso, optamos por comenzar en 0 en nuestro patrón de caracteres para la contraseña en el ejemplo anterior. Veremos cómo se almacenan datos en las **listas** más adelante en esta actividad.

Paso 2: Planificar su contraseña (continuación)

- ❑ Tómese 2 minutos para planificar el patrón de su contraseña en la tabla a continuación. Para simplificar, sugerimos combinar letras mayúsculas y minúsculas. Si quiere que el patrón sea más complejo, puede especificar lugares que serán letras mayúsculas o minúsculas.

Cantidad de caracteres	Tipo de carácter <i>Letra (mayúscula o minúscula), número o carácter especial</i>
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

Nota: El patrón de la contraseña puede ser diferente en comparación con lo que se ilustra en el ejemplo de esta actividad. Recuerde esto cuando compare su código con los ejemplos de la Guía de referencia.

Consejo: Imprima esta página para que pueda consultarla más adelante.

Paso 3: Comenzar con Repl.it (de 15 a 20 minutos)



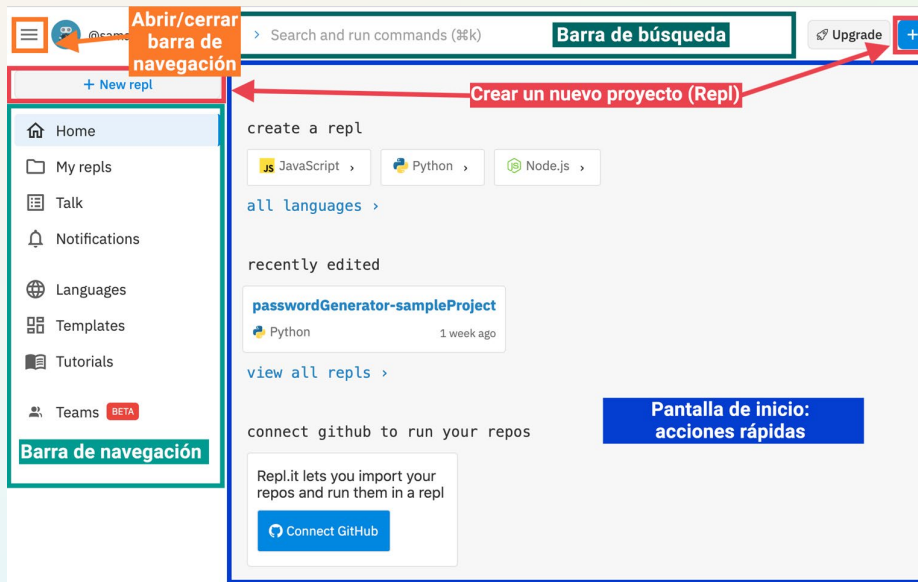
Para este proyecto, usaremos el editor web [Repl.it](https://repl.it). Repl.it es un editor gratuito, colaborativo y basado en el navegador que admite múltiples lenguajes de programación. Esta potente herramienta le permite incluso codificar y hablar con un grupo de amigos al mismo tiempo.

Crear una cuenta (de 3 a 5 minutos)

- ❑ **Suscríbase o inicie sesión en Repl.it.** Para guardar su trabajo, deberá crear una cuenta. Siga las instrucciones que aparecerán en el formulario de inscripción para crear una cuenta. Si usted es menor de 13 años de edad, necesitará la dirección de correo electrónico de uno de sus padres para suscribirse.
- ❑ **Siga las instrucciones para crear una cuenta.** Puede optar por suscribirse con su cuenta de Google, GitHub o Facebook para acceder más rápidamente.

Explorar la plataforma de Repl.it (de 3 a 5 minutos)

Le damos la bienvenida a la plataforma de Repl.it. Exploremos algunas de las características clave disponibles para empezar.



Quizá observe varias menciones a GitHub en el navegador Repl.it. **GitHub** es un gestor de desarrollo de software popular. No usaremos esta herramienta en esta actividad, pero si quiere obtener más información, consulte el [tutorial de GitHub](#).

- **Barra de navegación.** Esta columna, situada a la izquierda de la pantalla, le permite acceder a las acciones más frecuentes que quizá usted quiera realizar, como crear o ver sus Repls.
- **Barra de búsqueda.** En lugar de usar la barra de navegación, también puede usar la barra de búsqueda para buscar y ejecutar comandos.
- **Nuevo Repl.** Hay dos formas de crear un nuevo proyecto. La primera es a través de la barra de navegación y haciendo clic en el botón **+New repl** o en el botón **+** en la parte superior derecha de la ventana.
- **Área principal.** En el centro de la pantalla; el área principal variará según la vista. La primera vez que inicie sesión en Repl.it, aparecerá la pantalla de inicio de forma predeterminada.

Puede obtener más información sobre la plataforma Repl.it y Repls usando este [recurso](#).

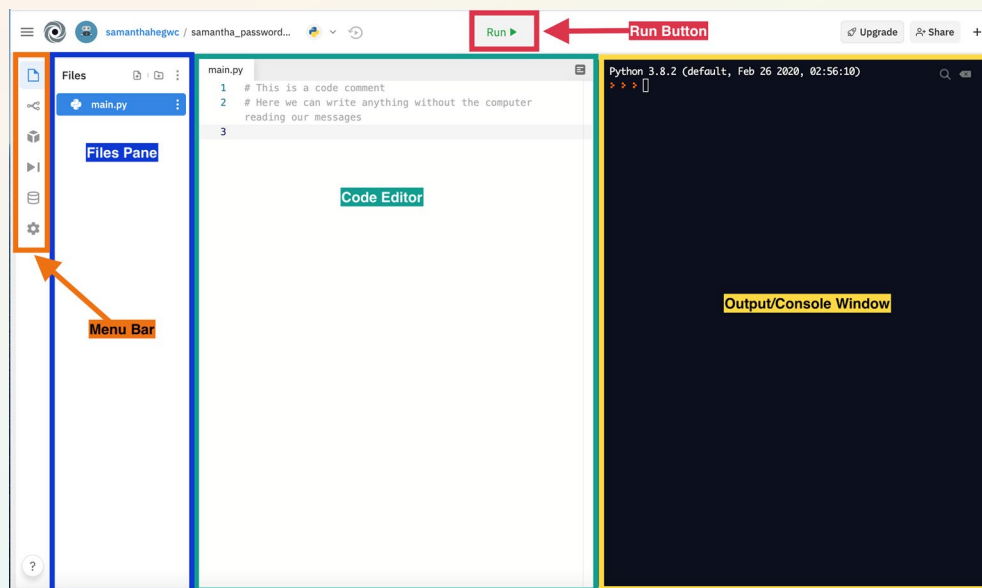
Paso 3: Comenzar con Repl.it (continuación)

Crear un proyecto nuevo con Repl.it (de 2 a 4 minutos)

- ❑ **Cree un nuevo repl.** Haga clic en el botón azul **+** en la parte superior derecha de la pantalla o en el botón **+New repl** de la barra de navegación.
- ❑ **Seleccione Python en la opción de lenguaje.** Para este proyecto, trabajaremos en Python; pero recuerde que para un proyecto futuro, usted puede programar en más de 50 lenguajes en Repl.it.
- ❑ **Desígnele un nombre a su proyecto.** Dele a su proyecto un nombre descriptivo, por ejemplo, **<yourName>_pwGenerator**. Por lo general, los proyectos no deben tener espacios ni usar **camelCase** o guiones bajos para separar las palabras.
- ❑ **Haga clic en Crear Repl.**

Explorar la vista del editor (de 5 a 8 minutos)

Veamos la vista del editor de Repl.it. Ahora que hemos creado un nuevo proyecto, tenemos que entender dónde codificar; cómo navegar entre los archivos y recursos digitales; y cómo guardar y ejecutar el código.



- **Panel de archivos.** De forma predeterminada, esta área le mostrará todos los archivos asociados al proyecto. En esta actividad, solo trabajaremos con un archivo: main.py. Es posible que en otro proyecto quiera incluir archivos de datos, imágenes u otros archivos de Python. Para cerrar este panel, haga clic en el ícono de archivos en la parte superior de la barra de menú. 📁
- **Barra de menú.** Esta barra le permitirá cambiar la vista en el panel de archivos. Entre algunas de las opciones, se incluyen cambiar el control de versiones, añadir paquetes, depurar, agregar una base de datos y actualizar la configuración. En los ajustes, puede personalizar el diseño, el tema, el tamaño de la letra y otros ajustes del texto.
- **Editor de código.** Aquí es donde escribirá su código.
- **Botón de ejecutar.** Después de realizar los cambios en el código, haga clic en el botón de ejecutar en la parte superior del editor. Debería ver el resultado en la ventana de salida/console de la derecha.
- **Ventana de resultado/console.** Muestra el resultado del código. Todos los resultados estarán visibles. Por eso, si hace clic en el botón de ejecutar varias veces, cada resultado se mostrará aquí.

Habrá notado que no hay un botón de **guardar** en Repl.it. Mientras usted tenga conexión a Internet, todos los cambios en el código se guardarán automáticamente. Una vez que haga clic en el botón de ejecutar, su Repl se guardará; por eso, asegúrese de ejecutar siempre el código antes de cerrar el editor.

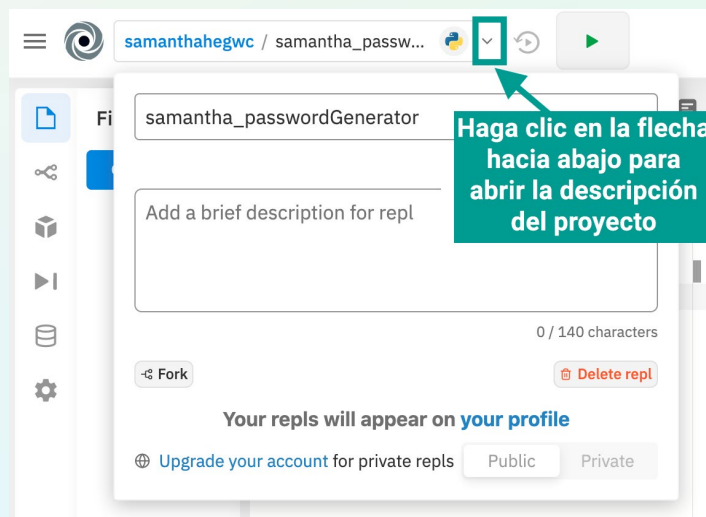
Paso 3: Comenzar con Repl.it (continuación)



Siempre lo mejor es hacer una copia de seguridad del código en caso de que el código tenga un error o no se guarde correctamente cuando usted cierre Repl.it. Puede obtener más información sobre cómo usar la opción de control de versiones en la barra de menú para configurar un repositorio Git y respaldar su trabajo con este [recurso](#). Si bien no usaremos esta opción para este proyecto, es un recurso valioso para seguir creando más proyectos con Repl.it.

Antes de empezar a codificar nuestro generador de contraseñas, vamos a añadir una breve descripción del proyecto a nuestro Repl.

- ❑ **Abra la descripción del proyecto.** Localice el nombre de su proyecto en la parte superior izquierda de la pantalla. Haga clic en la pequeña flecha hacia abajo a la derecha del nombre. Debería abrirse una nueva ventana con el nombre de su proyecto y un área para añadir una breve descripción.
- ❑ **Añada una breve descripción.** En la descripción, usted podría incluir algo como lo siguiente:
 - ❑ **Descripción general:** ¿Cómo se supone que funciona?
 - ❑ **Instrucciones:** ¿Hay instrucciones específicas para ejecutar su proyecto?
 - ❑ **Atributos:** ¿Obtuvo ayuda de otras personas o recursos adicionales? Asegúrese de dar a conocer a estas personas y estos recursos.



*Esta sección no tiene por qué terminar acá;
puede regresar y editar la descripción más adelante.*

Paso 4: Introducción a Python (de 3 a 5 minutos)



Python es un lenguaje de programación basado en texto, lo que significa que es necesario teclear todos los comandos. Muchos programadores prefieren usar Python porque es fácil de aprender y entender, dado que imita los comandos en inglés. Python es un lenguaje de **código abierto**, lo que significa que está disponible de forma gratuita para que el público lo use y lo modifique según sea necesario. Hay lineamientos estrictos para aceptar e implementar las actualizaciones al lenguaje, pero cualquier persona puede contribuir a su evolución.

Lo primero que añadiremos a nuestro proyecto son algunos **comentarios de código**. Los comentarios de código son líneas de código que la computadora no lee. Los programadores suelen usar estos comentarios para dejar notas en el proyecto mediante las cuales se explican algunas líneas de código y, así, hacer que el código sea más legible. De esta manera, es más fácil para los demás entender lo que el programador estaba tratando de lograr en su proyecto.

Para añadir un comentario de código de una sola línea en Python, debe comenzar cada línea con el símbolo **#**. Esto permite que la computadora “sepa” que la línea de código es un comentario; y no lee nada más allá del símbolo **#**. Veamos un ejemplo:

```
# Este es un comentario.  
# Aquí podemos escribir cualquier cosa sin que la  
computadora lea nuestros mensajes.  
message = "hi"
```

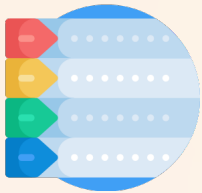
- ❑ **Tómese un momento para revisar el fragmento de código anterior. ¿Qué líneas de código son comentarios? ¿Cómo lo sabe?**



Recuerde verificar sus ideas con la Guía de referencia (página 3).

Nota: Dado que Repl.it restringe las descripciones a solo 140 caracteres, usted también puede usar los comentarios del código de la parte superior del proyecto para añadir una descripción o instrucciones más detalladas a su proyecto.

Paso 5: Conocer las listas de Python (de 10 a 15 minutos)



Una **lista** es un conjunto de datos ordenados que se usa para almacenar diferentes tipos de información. Usaremos las **listas** en nuestro proyecto para almacenar todos los posibles caracteres para la contraseña generada aleatoriamente. Las **listas** usan índices o un número para hacer referencia a cada elemento en su interior. Esto facilita la recuperación, la adición, la eliminación y la modificación de la información dentro de la **lista**. Los índices comienzan en 0 y se incrementan en 1;

por eso optamos por comenzar en 0 en nuestro patrón de caracteres para la contraseña en el ejemplo anterior. Veamos un ejemplo:

PYTHON	DESCRIPCIÓN
<pre>programmers = ["Ada", "Grace", "Katherine", "Roya"]</pre>	<ul style="list-style-type: none">→ programmers: El nombre de la variable que almacenamos en nuestra lista.→ =: Usamos el signo igual para mostrar la asignación de un objeto/valor a una variable.→ []: Los corchetes se usan para mostrar el inicio y el final del contenido de una lista.→ "Ada", "Grace", "Katherine", "Roya": En esta lista, agregamos los nombres de las programadoras. Usamos comillas dobles o simples para indicar un tipo de dato asiento o una palabra.→ ,: Usamos comas para separar cada dato/valor almacenado en la lista.

En esta **lista**, los índices se asignan a cada nombre de la programadora empezando por el índice 0.

```
0           1           2           3  
["Ada", "Grace", "Katherine", "Roya"]
```

Podemos ver que Ada se encuentra en el índice 0 de la **lista de programadoras**; Grace está en el índice 1 de la **lista de programadoras**; y así sucesivamente.

Paso 5: Conocer las listas de Python (continuación)

Practicar la identificación del índice de la lista (de 3 a 5 minutos)

Tomémonos un momento para asegurarnos de que usted entiende la estructura de las **listas** y cómo se usan los índices para almacenar y hacer un seguimiento de información específica en una **lista**. En este proyecto de práctica, escribimos todo el código. Usted ejecutará el proyecto y responderá las preguntas en la ventana de resultado/console.

En este proyecto, tenemos una **lista** con los siguientes datos:

```
wit = ["Ada", "Grace", "Violet", "Ayanna", "Miral", "ENIAC"]
```

Se le harán una serie de preguntas, como las siguientes:

¿Qué es WIT en el índice 2?

Para responder, solo tiene que escribir su respuesta y presionar la tecla Intro en la consola. Recuerde que Python distingue entre mayúsculas y minúsculas, así que asegúrese de que todas las respuestas empiecen con mayúscula.

- ❑ Haga clic en este [enlace](#) para abrir la lista de práctica de Repl.it. Este es un proyecto de práctica para que ponga a prueba sus conocimientos de las **listas** y los índices.
- ❑ Haga clic en el botón de ejecutar (RUN) en la parte superior.
- ❑ Responda las preguntas en la ventana de la consola. Este programa seguirá haciéndole estas preguntas infinitamente. Conteste correctamente al menos 3 o 4 preguntas antes de pasar a la siguiente tarea.

Asignación de valores de la lista usando el índice (de 2 a 4 minutos)

Podemos acceder fácilmente a la información almacenada en las listas haciendo referencia al índice de los datos que queremos. Veamos la sintaxis para hacerlo.

PYTHON	DESCRIPCIÓN
<code>listName[index] = newValue</code>	<ul style="list-style-type: none">→ listName: Este es el nombre de la variable que almacena la lista de elementos.→ []: Usamos los corchetes para identificar que queremos acceder a la información de nuestra lista.→ index: Este es el índice, o el número, de los datos a los que se quiere acceder.→ =: Usamos el signo igual para mostrar la asignación o la reasignación a un nuevo valor.→ newValue: Estos son los nuevos datos que usted quiere almacenar.

Usaremos el ejemplo anterior para ver cómo podemos asignar valores a un índice específico en una lista.

```
programmers = ["Ada", "Grace", "Katherine", "Roya"]
```

- ❑ ¿Qué línea de código usaríamos para cambiar "Katherine" por "Violet" en la lista?



Recuerde verificar sus ideas con la Guía de referencia (página 3).

Paso 6: Crear todos los caracteres posibles (de 10 a 15 minutos)

Finalmente, estamos preparadas para empezar a codificar nuestro generador de contraseñas, pero ¿por dónde empezamos? Pensemos en lo primero que necesitamos: ¡los caracteres! Tenemos que decirle a nuestro programa qué caracteres son opciones válidas al generar una contraseña. Para hacerlo, crearemos tres **listas** para separar los *tipos* de caracteres posibles: letras, números y caracteres especiales o símbolos.

Para simplificar, decidimos combinar las letras mayúsculas y las letras minúsculas en las mismas listas. Puede optar por hacer su patrón más complejo separando las letras en dos **listas** diferentes, una para las minúsculas y otra para las mayúsculas.

Crear las variables de la lista para almacenar los tipos de caracteres posibles (de 5 a 8 minutos)

- ❑ **Abra el proyecto del generador de contraseñas en Repl.it.**
- ❑ **Añada un comentario de código en la parte superior para identificar esta sección donde creará las listas.** El comentario del código puede ser tan sencillo como “Listas de posibles tipos de caracteres”. Recuerde que los comentarios del código deben incluir el símbolo **#** al comienzo de la línea.
- ❑ **Cree una variable para almacenar una lista de todos los posibles tipos de letras, tanto mayúsculas como minúsculas.** Asegúrese de incluir todas las letras del alfabeto en mayúsculas y minúsculas separadas por comas en la **lista**. Nombramos a esta variable **letters**, pero puede crear su propio nombre de variable. *Recuerde que las listas usan corchetes **[]** para indicar el inicio y el final del contenido y que toda la información que contengan debe estar separada por comas. Dado que estos son caracteres, cada letra debe ser un tipo de dato asiento con comillas simples o dobles.*
- ❑ **Cree una variable para almacenar una lista de todos los posibles números.** Asegúrese de incluir todos los números separados por comas en la **lista**. Recuerde que vamos a leer estos números como asientos o palabras, por lo que tendrá que incluir comillas simples **' '** o dobles **" "** con cada número. Nombramos a esta variable **numbers**, pero puede crear su propio nombre de variable.
- ❑ **Cree una variable para almacenar una lista de todos los posibles caracteres especiales o símbolos.** Siéntase libre de incluir todos los símbolos o de seleccionar solo algunos. Tenga cuidado de no incluir comillas (**' '** o **" "**) en los símbolos porque Python las lee como caracteres clave que hacen alusión a un asiento. Esto puede generar confusión en el programa. Nombramos a esta variable **sChars**, pero usted puede crear su propio nombre de variable.



Recuerde verificar sus ideas con la Guía de referencia (página 4).

Conocer la función `print()` (2 minutos)

Si hace clic en el botón de ejecutar (**Run**), verá que no sucede nada. Esto se debe a que creamos variables, pero no le dijimos a la computadora que hiciera algo con esa información. Una excelente manera de depurar o identificar errores en su programa es usar la función `print()` (imprimir) en el código para probar que las variables tengan la información correcta. Las **funciones** son procedimientos o bloques de código que pueden llamarse para ejecutar una tarea. Los programadores suelen usarlas para hacer más legible el código. La función `print()` es una función incorporada de Python que toma una entrada y la muestra en la consola.



PYTHON	DESCRIPCIÓN
<code>print(input)</code>	<ul style="list-style-type: none">→ <code>print()</code>: El nombre de la función de Python que imprime mensajes en la consola.→ <code>input</code>: Esta entrada puede tener la forma de un asiento o de una variable.

Probar su código (de 3 a 5 minutos)

Usemos la función `print()` para comprobar que nuestras listas se hayan implementado correctamente. Cada sentencia de impresión debe añadirla *debajo* de todas las variables de la **lista**.

- ❑ Añadir una sentencia `print()` para imprimir el contenido de la **lista de letras**.
- ❑ Añadir una sentencia `print()` para imprimir el contenido de la **lista de números**.
- ❑ Añadir una sentencia `print()` para imprimir el contenido de la **lista de caracteres especiales**.
- ❑ Hacer clic en el botón de ejecutar para ver el contenido de las **listas**.

Usted debería tener:

- una **lista** impresa de todas las posibles letras;
- una **lista** impresa de todos los posibles números;
- una **lista** impresa de todos los posibles caracteres especiales.

¿No funciona de la manera que usted desea?

Pruebe estos consejos de depuración:

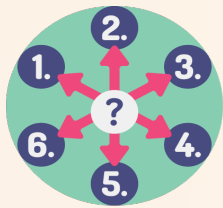
- ¿Escribió correctamente el nombre de la función?
- ¿Separó cada carácter de la **lista** con comas?
- ¿Usó comillas simples o dobles para cada carácter?
- ¿Usa corchetes para el contenido de cada **lista**?
- ¿Tiene paréntesis o llaves adicionales? Tal vez note que al escribir un símbolo (, el editor añade automáticamente el paréntesis de cierre). Esto podría provocar que usted agregue paréntesis o corchetes adicionales por accidente.

```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']
['!', '@', '#', '$', '%', '^', '&', '*', '(', ')', '-', '+']
```



Recuerde verificar sus ideas con la Guía de referencia (página 4).

Paso 7: Entender la aleatoriedad en las listas (5 minutos)



Por fin llegó el momento de crear su contraseña generada aleatoriamente. Para almacenar nuestra contraseña final, crearemos una **lista** de contraseñas. A continuación, asignaremos cada índice a uno de nuestros caracteres aleatorios. Esto puede parecer mucho, pero en realidad se puede hacer con solo algunas líneas de código.

Conocer la biblioteca aleatoria (2 minutos)

Una **biblioteca** es una colección de funciones y variables. Los programadores usan las bibliotecas porque muchas funciones ya están escritas para que se puedan usar. La biblioteca integrada de Python nos ayudará a seleccionar caracteres aleatoriamente de nuestras **listas**. Antes de usar cualquiera de las funciones de una biblioteca, primero tenemos que decirle a la computadora que **importe** o cargue la biblioteca.

PYTHON	DESCRIPCIÓN
<code>import random</code>	<ul style="list-style-type: none">→ import: Esta palabra clave permite a la computadora saber que debe cargar la información desde una biblioteca de Python.→ random: La biblioteca de Python se usa para generar números aleatorios.

La biblioteca **random** contiene muchas funciones, pero usaremos la función **choice()** en nuestro proyecto. Esta función nos permite seleccionar aleatoriamente un elemento de una **lista** o secuencia. Veamos la sintaxis de esta función.

PYTHON	DESCRIPCIÓN
<code>random.choice(listName)</code>	<ul style="list-style-type: none">→ random: Esto le permite a la computadora saber que vamos a usar una función de la biblioteca random.→ .: Este símbolo se usa a menudo para indicar la llamada a una función desde una biblioteca u otro archivo.→ choice(): Este es el nombre de la función que queremos usar.→ listName: El nombre de la lista de la cual queremos seleccionar.

Paso 8: Generar el primer carácter aleatorio (de 5 a 10 minutos)



Ya estamos preparadas para empezar a codificar nuestras contraseñas generadas aleatoriamente. Quizá quiera tener su patrón planificado para la contraseña (en la página 7) impreso o fácilmente accesible para este paso.

En una sección anterior, añadimos las sentencias `print()` para depurar el código y verificar que las **listas** se implementaron correctamente. Quizá opte por *eliminar* o *comentar* estas líneas de código dado que no contribuyen al objetivo general del proyecto.

- ❑ **Añada una nueva línea de código en la parte superior del programa, arriba de las variables, para importar la biblioteca `random`.** También puede añadir un comentario arriba para recordar lo que hace esta línea de código. Normalmente, los programadores añaden un recurso adicional, como la importación de bibliotecas, en las primeras líneas de código de su programa.
- ❑ **Cree una nueva variable de contraseña y asigne una lista de 10 valores vacíos en la parte inferior del código.** Nombramos a esta variable `pw`, pero puede crear su propio nombre de variable. Puede optar por crear una lista que contenga varios `0` o asientos `" "` vacíos. En los próximos pasos, reemplazaremos cada uno de estos valores.
- ❑ **Asigne el índice cero de la lista de contraseñas a un elemento aleatorio en una de las listas de caracteres con la función `random.choice()`.** Use su patrón de contraseñas para determinar la lista que debería elegir para cada índice. Por ejemplo, si quiere que una letra aparezca primero en su contraseña, asegúrese de indicar las listas de letras como una entrada en la función `random.choice()`.
- ❑ **Añada una línea de código para imprimir la lista de contraseñas.** Antes de avanzar, imprimiremos la lista para asegurarnos de haber recuperado correctamente un carácter y reasignado el valor en el índice 0.



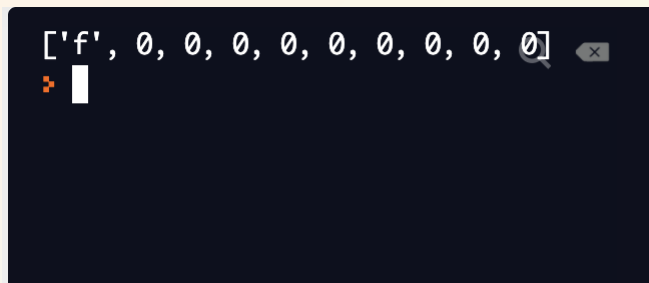
Recuerde verificar sus ideas con la Guía de referencia (página 5).

Paso 8: Generar el primer carácter aleatorio (continuación)

Probar su código (de 3 a 5 minutos)

Pongamos a prueba lo que hemos escrito hasta ahora para asegurarnos de que nuestro programa funcione de la manera que queremos. Haga clic en el botón de **ejecutar** para ejecutar su código. Usted debería tener:

- Una **lista** de contraseñas impresa en la que el primer carácter sea una letra, un número o un carácter especial aleatorios y los demás caracteres sean 0 o espacios " " vacíos.



```
[ 'f', 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
```

¿No funciona de la manera que usted desea? **Pruebe estos consejos de depuración:**

- ¿Escribió correctamente los nombres de la función y de la biblioteca?
- ¿Separó cada carácter de la **lista** con comas?
- ¿Usó comillas simples o dobles para cada carácter?
- ¿Usa corchetes **[]** para llamar o asignar valores a una **lista**?
- ¿Tiene paréntesis o llaves adicionales? Tal vez note que al escribir un símbolo (, el editor añade automáticamente el paréntesis de cierre). Esto podría provocar que usted agregue paréntesis o corchetes adicionales por accidente.

Paso 9: Crear su contraseña aleatoria (de 5 a 10 minutos)



Ahora que entendemos cómo generar el primer carácter aleatorio, continuemos y codifiquemos el resto del patrón de nuestra contraseña. Cuando asigne los índices remanentes de la lista de contraseñas, debería hacerlo **encima** de la sentencia de impresión de la contraseña.

- ❏ **Asigne los índices restantes de la lista de contraseñas a un elemento aleatorio en una de las listas de caracteres con la función `random.choice()`.** Tendrá que repetir este paso 9 veces, y deberá asignar a cada índice restante de 1 a 9 un carácter aleatorio de una de las listas de tipos de caracteres.

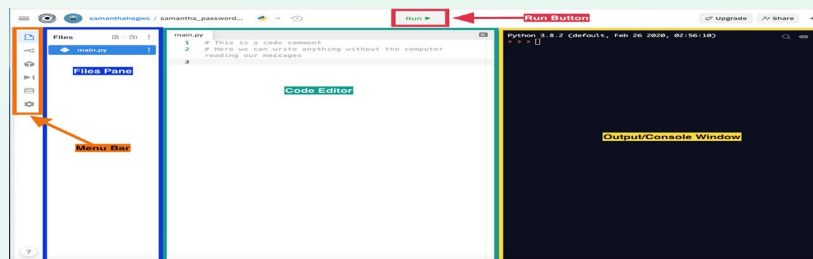


Recuerde verificar sus ideas con la Guía de referencia (página 6).

Probar su código (de 3 a 8 minutos)

Pongamos a prueba lo que hemos escrito hasta ahora para asegurarnos de que nuestro programa funcione de la manera que queremos. Haga clic en el botón de **ejecutar** para ejecutar su bordado. Usted debería tener:

- Una **lista** de contraseñas impresa en la que cada letra, número o carácter especial sea aleatorio y siga un patrón planificado de su contraseña.



¿No funciona de la manera que usted desea? **Pruebe estos consejos de depuración:**

- ¿Escribió correctamente los nombres de la función y de la biblioteca?
- ¿Separó cada carácter de la **lista** con comas?
- ¿Usó comillas simples o dobles para cada carácter?
- ¿Usa corchetes **[]** para llamar o asignar valores a una **lista**?
- ¿Tiene paréntesis o llaves adicionales? Tal vez note que al escribir un símbolo (, el editor añade automáticamente el paréntesis de cierre). Esto podría provocar que usted agregue paréntesis o corchetes adicionales por accidente.
- ¿Sus líneas de código están en la ubicación y secuencia correctas? ¡Recuerde que el orden es importante en el flujo del programa!

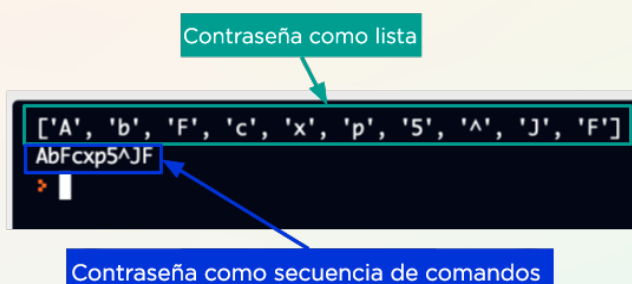
Paso 10: Extensiones (de 2 a 20 minutos)



Use estas extensiones para añadir más funciones y complejidad a su generador de contraseñas. Estos desafíos están pensados para ser más difíciles, ¡pero no se desanime! Use los Recursos de extensión para informarse más sobre los conceptos tratados en cada desafío. También puede ser útil mirar el proyecto de ejemplo y hacer ingeniería inversa de los pasos mirando el código.

Extensión 1: Imprimir la contraseña como un asiento (de 2 a 5 minutos)

Quizá note que al imprimir los resultados de las **listas** de contraseñas, estas aparecen con comas entre cada carácter. Si queremos representar nuestra contraseña como un asiento, podemos usar la función `join()`.



Haga clic [aquí](#) para ver un bordado de ejemplo de esta extensión.

La función `join()` en Python le permite tomar un objeto iterable o estructuras como **listas** que contienen múltiple información y en las que se puede acceder fácilmente a cada elemento, y combina o *une* cada elemento con un **asiento** o una palabra.

PYTHON	DESCRIPCIÓN
<code>stringName.join(itrObject)</code>	<ul style="list-style-type: none">→ stringName: Esta es una variable que contiene un asiento o una palabra.→ .: Este símbolo se usa a menudo para indicar la llamada a una función.→ join(): El nombre de la función que combina un objeto iterable y un asiento en un asiento.→ itrObject: Esta es una variable que contiene un objeto iterable, como una lista.

- ❑ Use la función `join()` para combinar un asiento "" vacío con la lista de contraseñas y almacenar esto en una **variable**. Nombramos a esta variable `pws`, pero puede crear su propio nombre de variable. También puede añadir un comentario arriba para recordar lo que hace esta línea de código.
- ❑ **Imprima el asiento de su contraseña.**

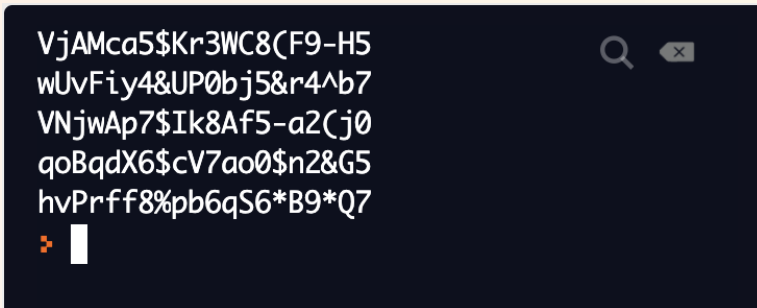
Recursos de extensión

A continuación, encontrará algunos recursos útiles que utilizamos para crear esta extensión. Esto será de ayuda para comenzar, pero recuerde que hay muchos más recursos usando simplemente el motor de búsqueda.

- [join\(\)](#)
- [Objetos iterables](#)

Extensión 2: Incremento de la longitud de la contraseña (de 10 a 15 minutos)

Ahora nuestra contraseña tiene una longitud de 10 caracteres, pero ¿qué pasa si queremos que sea aún más segura? La mayoría de los administradores de contraseñas le recomendarán que cree una contraseña de al menos 15 caracteres o incluso de 20 caracteres.



Haga clic [aquí](#) para ver un bordado de ejemplo de esta extensión.

Haga su contraseña más compleja añadiendo caracteres adicionales a la **lista** de contraseñas y asignando los caracteres restantes a valores aleatorios.

- ❑ **Planifique el patrón restante de la contraseña.**
- ❑ **En la variable de la lista de contraseñas, añada elementos adicionales para que la lista tenga la longitud deseada.** Para aumentar nuestra contraseña de 10 a 20 caracteres, añadimos solo diez **0** más a la **lista** entre corchetes `[]`.
- ❑ **Asigne los índices restantes de la lista de contraseñas a un elemento aleatorio en una de las listas de caracteres con la función `random.choice()`.**

Extensión 3: Generar contraseñas con un patrón de carácter aleatorio (de 10 a 15 minutos)

En nuestro programa, usted elige el patrón de su contraseña, pero ¿qué pasa si queremos aleatorizar el patrón de nuestras contraseñas? Podemos usar la biblioteca **random** para seleccionar aleatoriamente de cualquier lista de caracteres. Para esta extensión, deberá usar un **bucle** y **sentencias condicionales** para generar nuestra contraseña.

```
3$+l0lv3Dq
%x6812%#9k
)1h)%K6c*q
5U@d4f$P7$
ek4114$491
> |
```

Haga clic [aquí](#) para ver un bordado de ejemplo de esta extensión.

En esta extensión, intente usar un bucle para asignar aleatoriamente cada carácter de su contraseña a un tipo de carácter aleatorio en lugar de asignar un patrón específico. Estos son los pasos básicos que necesita para completar esta extensión:

- ❑ Cree una nueva variable de contraseña y asigne una lista de 10 valores vacíos en la parte inferior del código.
- ❑ Cree un bucle **for** con la función **range()** que itera 10 veces. Usamos la variable **i** para almacenar el valor de iteración de la función **range**, pero puede crear su propio nombre de variable.
- ❑ Dentro del bucle **for**, use la función **random.randint()** para elegir aleatoriamente un número entre 1 y 3 y almacenar el resultado en una variable. Elegiremos un número al azar para seleccionar qué tipo de carácter debe ser seleccionado. Como tenemos 3 tipos (letras, números y caracteres especiales), restringimos el intervalo para que sean solo los números 1, 2 y 3. Nombramos a esta variable **charType**, pero puede crear su propio nombre de variable.
- ❑ En el bucle **for**, añada sentencias condicionales que asignen el carácter en la lista de contraseñas en función del tipo de carácter aleatoriamente seleccionado por la función **randint()**. En nuestro ejemplo, dijimos que si **charType** era 1, seleccionaríamos aleatoriamente una letra. Si **charType** fuera 2, seleccionaríamos aleatoriamente un número. Por último, si **charType** fuera 3, seleccionaríamos aleatoriamente un carácter especial. Siéntase libre de asignar sus propios valores en un orden diferente de tipos de caracteres.

*Tendremos que usar la variable **i** del bucle **for** para asignar cada carácter en la lista de contraseñas. La variable **i** se actualizará después de cada iteración del bucle **for** y tendrá valores del 0 al 9.*

- ❑ Fuera del bucle **for**, imprima la contraseña generada aleatoriamente.

Recursos de extensión

A continuación, encontrará algunos recursos útiles que utilizamos para crear esta extensión. Esto será de ayuda para comenzar, pero recuerde que hay muchos más recursos usando simplemente el motor de búsqueda.

- [Bucles For](#). También puede usar el [bucle while](#) para este ejemplo, pero recomendamos un bucle for, ya que conocemos el número concreto de iteraciones.
- [función range\(\) con bucles for](#).
- [random.randint\(\)](#)
- [Sentencias condicionales](#)

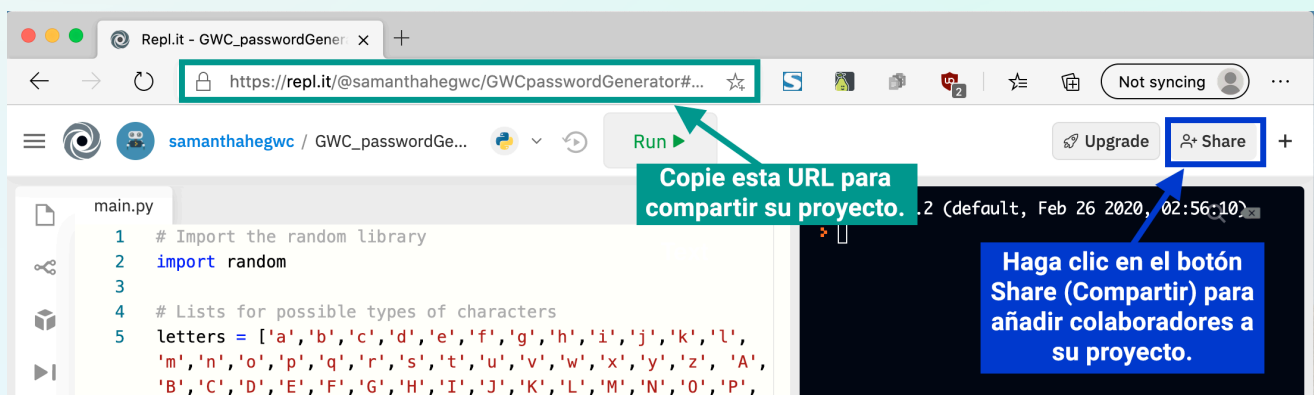
Paso 11: Compartir su proyecto de Girls Who Code en casa (de 5 a 10 minutos)

Nos encantaría ver su trabajo y sabemos que a otros también les gustaría. Comparta su proyecto final con nosotros. No olvide etiquetar [@girlswhocode](#) [#codefromhome](#), ¡y quizá la destaquemos en nuestra cuenta!

Acceso de solo lectura (1 minuto)

Compartir su trabajo en Repl.it es fácil. Solo tiene que copiar y pegar la dirección URL de su proyecto Repl en la barra de direcciones web de la parte superior. Esta acción permitirá que otros ejecuten su proyecto, vean su código y **bifurquen** el proyecto y lo remezclen por su cuenta.

Comparta este enlace en sus cuentas de redes sociales y recuerde usar las etiquetas [@girlswhocode](#) [#codefromhome](#) para que podamos mencionarla en nuestra cuenta.



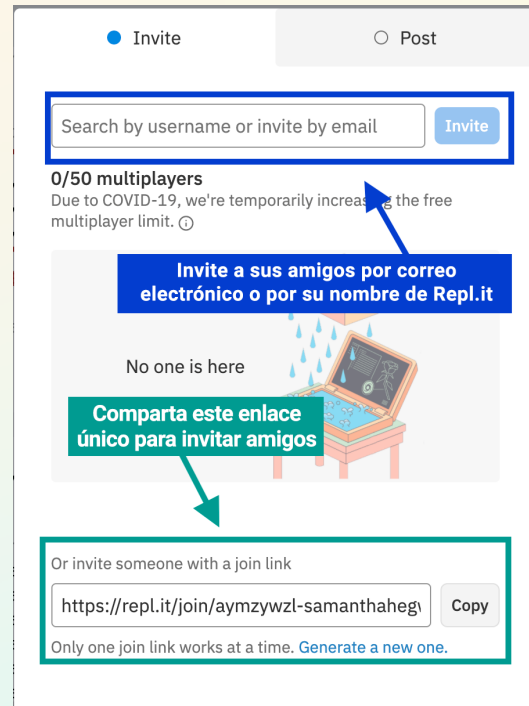
Paso 11: Compartir su proyecto de Girls Who Code en casa (continuación)

Agregar colaboradores (2 minutos)

Si quiere trabajar con un grupo de amigos en un proyecto, puede invitarlos fácilmente a colaborar mediante el botón de compartir (botón **Share**), que se encuentra en la parte superior derecha de la ventana. Debería aparecer una nueva ventana con dos opciones para invitar a otros a colaborar en su proyecto.

- **Invitar por correo electrónico o nombre de usuario de Repl.it.** Esta opción permite compartir su proyecto con personas específicas escribiendo la dirección de correo electrónico o el nombre de usuario de Repl.it si estas personas ya tienen cuenta en Repl.it. Se recomienda esta opción para que usted se asegure de que compartirá su proyecto con las personas adecuadas.
- **Comparta el enlace de invitación.** En la parte inferior de la ventana, hay un enlace de invitación único. Puede copiar y pegar este enlace para sus amigos y, de esa manera, accederán a su proyecto.

Nota acerca de colaboradores: Recuerde que, si agrega colaboradores, le dará a otros acceso de edición a su proyecto. De esta manera, los colaboradores podrán cambiar su código, el nombre y la descripción. **NO comparta el enlace de invitación en las redes sociales.** Seleccione con cuidado con quién comparte los derechos de edición.



Siga en contacto para recibir más información sobre Depuración de código faltante, parte 2.

