LAB-010-Exemplar_Develop an algorithm

September 20, 2024

1 Modelo: Desarrollar un algoritmo

Introducción

Un algoritmo es un conjunto de pasos que se pueden utilizar para resolver un problema. Los analistas de seguridad desarrollan algoritmos para proporcionar las soluciones que necesitan para su trabajo. Por ejemplo, un analista puede trabajar con usuarios que les traen dispositivos. Este puede necesitar un algoritmo que primero verifique si un usuario está aprobado para acceder al sistema y luego verifica si el dispositivo que le trajeron es el asignado a ellos.

En este laboratorio, desarrollarás un algoritmo en Python que automatiza este proceso.

Consejos para completar este laboratorio

Mientras recorres este laboratorio, ten en cuenta los siguientes consejos:

— `#### YOUR CODE HERE ###` indica dónde debes escribir el código. Asegúrate de reemplazarlo con tu propio código antes de ejecutar la celda de código. — Siéntete libre de abrir las pistas para obtener información adicional a medida que trabajas en cada tarea. — Para ingresar tu respuesta a una pregunta, haz doble clic en la celda de markdown para editar. Asegúrate de reemplazar "[Haz doble clic para ingresar aquí tus respuestas.]" con tu propia respuesta. — Puedes guardar tu trabajo manualmente con un clic en Archivo y luego en Guardar en la barra de menú en la parte superior del cuaderno. — Puedes descargar tu trabajo localmente con un clic en Archivo y luego en Descargar, luego puedes especificar el formato de archivo que prefieras en la barra de menú en la parte superior del cuaderno.

1.1 Situación hipotética

En este laboratorio, trabajarás como analista de seguridad y serás responsable de desarrollar un algoritmo que conecte a los usuarios con sus dispositivos asignados. Escribirás un código que indique si un usuario está aprobado en el sistema y trajo su dispositivo asignado al equipo de seguridad.

Tarea 1 Trabajarás con una lista de nombres de usuario aprobados junto con una lista de los dispositivos aprobados asignados a estos usuarios. Los elementos de las dos listas están sincronizados. En otras palabras, el usuario en la indexación 0 en `approved_users` utiliza el dispositivo en la indexación 0 en `approved_devices`. Más adelante, esto te permitirá verificar si el nombre de usuario y la ID de dispositivo ingresados por un usuario se corresponden entre sí.

En primer lugar, para explorar cómo funcionan las indexaciones en las listas, ejecuta la siguiente celda de código y observa el resultado. Luego, sustituye cada `0` por otra indexación y ejecuta la

celda para observar lo que ocurre.

[1]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "tshah", "sgilmore", "eraab"]

# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
 ↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "2ye3lzg", "4n482ts", "a307vir"]

# Muestra el elemento en la indexación especificada en `approved_users`

print(approved_users[0])

# Muestra el elemento en la indexación especificada en `approved_devices`

print(approved_devices[0])
```

```
elarson
8rp2k75
```

Pregunta 1 ¿Qué observaste en el resultado cuando se muestra `approved_users[0]` y `approved_devices[0]`? ¿Qué ocurre cuando se sustituye cada `0` por otra indexación?

Cuando se muestra `approved_users[0]`, el resultado es el primer nombre de usuario aprobado de `approved_users`. Cuando se muestra `approved_devices[0]`, el resultado es la primera ID de dispositivo de `approved_devices`. Cuando se sustituye cada `0` por otra indexación, el resultado es el elemento en esa indexación en `approved_users`, seguido por el elemento en esa indexación en `approved_devices`. Por ejemplo, si se sustituye cada `0` por `2`, el resultado es el elemento en la indexación 2 en `approved_users`, seguido por el elemento en la indexación 2 en `approved_devices`.

1.2 Tarea 2

Un empleado nuevo se une a la organización y se le debe proporcionar un nombre de usuario y una ID de dispositivo. En la siguiente celda de código, se te proporciona un nombre de usuario y una ID de dispositivo de este nuevo usuario, almacenados en las variables `new_user` y `new_device`, respectivamente. Utiliza el método `.append()` para agregar estas variables a los `approved_users` y `approved_devices` respectivamente. Después, muestra las variables `approved_users` y `approved_devices` para confirmar la información agregada. Asegúrate de reemplazar cada `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[2]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "tshah", "sgilmore", "eraab"]
```

```
# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "2ye3lzg", "4n482ts", "a307vir"]

# Asigna a `new_user` el nombre de usuario de un nuevo usuario aprobado

new_user = "gesparza"

# Asigna a `new_device` la ID de dispositivo del nuevo usuario aprobado

new_device = "3rcv4w6"

# Agrega el nombre de usuario y la ID de dispositivo de ese usuario a␣
↪`approved_users` y `approved_devices` respectivamente

approved_users.append(new_user)
approved_devices.append(new_device)

# Muestra el contenido de `approved_users`

print(approved_users)

# Muestra el contenido de `approved_devices`

print(approved_devices)
```

```
['elarson', 'bmoreno', 'tshah', 'sgilmore', 'eraab', 'gesparza']
['8rp2k75', 'hl0s5o1', '2ye3lzg', '4n482ts', 'a307vir', '3rcv4w6']
```

Pista 1

Utiliza el método `.append()` para agregar `new_user` a `approved_users`.

Utiliza el método `.append()` para agregar `new_device` a `approved_devices`.

Pista 2

Utiliza la función `print()` para mostrar el contenido de `approved_users`.

Utiliza la función `print()` para mostrar el contenido de `approved_devices`.

Pregunta 2 Después de agregar el nuevo usuario aprobado, ¿qué observaste en el resultado cuando se muestra `approved_users` y `approved_devices`?

Después de agregar el nuevo usuario aprobado, su nombre de usuario estará al final de `approved_users` y su ID de dispositivo estará al final de `approved_devices`.

Tarea 3 Un empleado dejó el equipo y ya no debería tener acceso al sistema. En la siguiente celda de código, se te proporciona el nombre de usuario y su ID de dispositivo que se eliminará, almacenados en las variables `removed_user` y `removed_device` respectivamente. Utiliza el método

`.remove()` para eliminar cada uno de estos elementos de la lista correspondiente. Después, muestra las variables `approved_users` y `approved_devices` para ver los usuarios eliminados. Ejecuta el código y observa los resultados. Asegúrate de reemplazar cada `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[3]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "tshah", "sgilmore", "eraab",␣
↪"gesparza"]

# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "2ye3lzg", "4n482ts", "a307vir",␣
↪"3rcv4w6"]

# Asigna a `removed_user` el nombre de usuario del empleado que dejó el equipo

removed_user = "tshah"

# Asigna a `removed_device` la ID de dispositivo del empleado que dejó el equipo

removed_device = "2ye3lzg"

# Elimina el nombre de usuario y su ID de dispositivo de `approved_users` y␣
↪`approved_devices` respectivamente

approved_users.remove(removed_user)
approved_devices.remove(removed_device)

# Muestra `approved_users`

print(approved_users)

# Muestra `approved_devices`

print(approved_devices)
```

```
['elarson', 'bmoreno', 'sgilmore', 'eraab', 'gesparza']
['8rp2k75', 'hl0s5o1', '4n482ts', 'a307vir', '3rcv4w6']
```

Pista 1

Utiliza el método `.remove()` para eliminar `removed_user` de `approved_users`.

Utiliza el método `.remove()` para eliminar `removed_device` de `approved_devices`.

Pista 2

Utiliza la función `print()` para mostrar el contenido de `approved_users`.

Utiliza la función `print()` para mostrar el contenido de `approved_devices`.

Pregunta 3 Después de eliminar el usuario que abandonó el equipo, ¿qué observaste en el resultado cuando se muestra `approved_users` y `approved_devices`?

Después de eliminar el usuario que abandonó el equipo, su nombre de usuario ya no formará parte de `approved_users`; tampoco su ID de dispositivo formará parte de `approved_devices`.

1.3 Tarea 4

Como parte de la verificación de la identidad de un usuario en el sistema, deberás verificar si este es uno de los usuarios aprobados. Escribe una sentencia condicional que verifique si un nombre de usuario dado es un elemento de la lista de nombres de usuario aprobados. Si es así, muestra `"El usuario ______ está aprobado para acceder al sistema".`. De lo contrario, muestra `"El usuario ______ no está aprobado para acceder al sistema".`. Asegúrate de reemplazar cada `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[4]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
 ↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Asigna a `username` un nombre de usuario

username = "sgilmore"

# Sentencia condicional
# Si `username` pertenece a `approved_users`, muestra "El usuario ______ está␣
 ↪aprobado para acceder al sistema".
# De lo contrario, muestra "El usuario ______ no está aprobado para acceder al␣
 ↪sistema".
if username in approved_users:
    print("El nombre de usuario", username, "está aprobado para acceder al␣
 ↪sistema.")
else:
    print("El nombre de usuario", username, "no está aprobado para acceder al␣
 ↪sistema.")
```

```
El nombre de usuario sgilmore está aprobado para acceder al sistema.
```

Pista 1

En la condición `if`, asegúrate de verificar si `username` pertenece a `approved_users`.

Pista 2

Después de la sentencia `if`, utiliza la palabra clave `else` para crear una sentencia `else` que gestione el caso cuando `username` no forme parte de `approved_users`.

Pista 3

En la sentencia `else`, utiliza la función `print()` para mostrar el mensaje `"El usuario _____ no está aprobado para acceder al sistema"`..

Consulta la llamada a la función `print()` en la sentencia `if` y observa cómo las comas separan una cadena que contiene la primera parte del mensaje, la variable `username` y otra cadena que contiene la segunda parte del mensaje.

Pregunta 4 ¿Qué mensaje observas en el resultado cuando `username` es `"sgilmore"`?

Cuando `username` es `"sgilmore"`, el mensaje emitido dice `"El nombre de usuario sgilmore está aprobado para acceder al sistema."`, ya que `"sgilmore"` es un elemento de `approved_users`.

Tarea 5 En la siguiente parte del algoritmo, utilizarás el método `.index()` para encontrar la indexación de `username` en la `approved_list` y almacenarla en una variable llamada `ind`.

Cuando se utiliza en una lista, el método `.index()` devolverá la posición del valor dado en la lista.

Agrega una sentencia para mostrar `ind` en la siguiente celda de código para explorar el valor que contiene. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[5]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Asigna a `username` un nombre de usuario

username = "sgilmore"

# Asigna a `ind` la indexación de `username` en `approved_users`

ind = approved_users.index(username)

# Muestra el valor de `ind`

print(ind)
```

```
2
```

Pista 1

Utiliza la función `print()` para mostrar el valor de `ind`.

Pregunta 5 ¿Qué observas en el resultado cuando `username` es `"sgilmore"`?

Cuando `username` es `"sgilmore"`, el resultado es `2`, lo que indica que el valor de indexación de `"sgilmore"` es 2 en `approved_users`. En otras palabras, `"sgilmore"` es el tercer elemento en `approved_users`. La indexación en Python comienza en 0.

1.4 Tarea 6

Esta tarea te permitirá desarrollar tu comprensión sobre las operaciones de lista para el algoritmo que eventualmente crearás. En primer lugar, utilizarás de nuevo el método `.index()` para encontrar la indexación de `username' enapproved_usersy almacenarla en una variable llamadaind. Luego, conectarásindaapproved_devicesy mostrarás la ID del dispositivo ubicada en la indexaciónind. A continuación, ejecutarás la celda para observar el resultado. Asegúrate de reemplazar cada### YOUR CODE HERE ###‘` con tu propio código antes de ejecutar la siguiente celda.

[6]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Asigna a `username` un nombre de usuario

username = "sgilmore"

# Asigna a `ind` la indexación de `username` en `approved_users`

ind = approved_users.index(username)

# Muestra la ID del dispositivo en la indexación que coincide con el valor de␣
↪`ind` en `approved_devices`

print(approved_devices[ind])
```

```
4n482ts
```

Pista 1

Utiliza el método `.index()` para obtener el valor de indexación de `username` en `approved_users`. Asigna a `ind` el resultado.

Pista 2

Para mostrar la ID de dispositivo correcta de `approved_devices`, utiliza `ind` como indexación. Coloca `ind` dentro de los corchetes para extraer el elemento correcto de `approved_devices`.

Pregunta 6 ¿Qué observas en el resultado cuando `username` es `"sgilmore"`?

Cuando `username` es `"sgilmore"`, el resultado es `4n482ts`, que es la ID de dispositivo que corresponde a `"sgilmore"`. El tercer nombre de usuario aprobado en `approved_users` es `"sgilmore"`, y de manera similar, la tercera ID de dispositivo en `approved_devices` es `"4n482ts"`.

1.5 Tarea 7

El siguiente paso para crear el algoritmo es determinar si un nombre de usuario y una ID de dispositivo se corresponden. Para ello, escribe una sentencia condicional que compruebe si el `username` es un elemento de `approved_devices` y si la `device_id` almacenada en la misma indexación que `username` coincide con la `device_id` ingresada. Utilizarás el operador lógico `and` para conectar las dos condiciones. Cuando ambas condiciones se evalúan en `True`, se muestra un mensaje de que el nombre de usuario está aprobado y otro mensaje de que tiene su dispositivo asignado. Asegúrate de reemplazar cada `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[7]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
 ↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Asigna a `username` un nombre de usuario

username = "sgilmore"

# Asigna a `device_id` una ID de dispositivo

device_id = "4n482ts"

# Asigna a `ind` la indexación de `username` en `approved_users`

ind = approved_users.index(username)

# Sentencia condicional
# Si `username` pertenece a `approved_users`, y si la ID de dispositivo en␣
 ↪`ind` en `approved_devices` coincide con `device_id`,
# entonces se muestra un mensaje de que el nombre de usuario está aprobado,
```

```
# seguido de un mensaje de que tiene el dispositivo correcto

if username in approved_users and device_id == approved_devices[ind]:
    print("El nombre de usuario", username, "está aprobado para acceder al␣
 ↪sistema.")
    print(device_id, "es el dispositivo asignado para", username)
```

```
El nombre de usuario sgilmore está aprobado para acceder al sistema.
4n482ts es el dispositivo asignado para sgilmore
```

Pista 1

Después del operador lógico `and`, escribe la segunda condición en la sentencia `if` con un operador de comparación para verificar si el elemento en `ind` en `approved_devices` coincide con `device_id`.

Pista 2

Utiliza el operador de comparación `==` para verificar si el elemento en `ind` en `approved_devices` coincide con `device_id`.

Pregunta 7 ¿Qué observas en el resultado cuando `username` es `"sgilmore"` y `device_id` es `"4n482ts"`?

Cuando `username` es `"sgilmore"` y `device_id` es `"4n482ts"`, el resultado consiste en `El nombre de usuario sgilmore está aprobado para acceder al sistema.` en la primera línea y `4n482ts es el dispositivo asignado para sgilmore` en la segunda.

1.6 Tarea 8

También sería útil que los usuarios reciban mensajes cuando su nombre de usuario no esté aprobado o su ID de dispositivo sea incorrecta.

Agrega al código escribiendo una sentencia `elif`. Esta sentencia `elif` debe ejecutarse cuando `username` forma parte de `approved_users` pero `device_id` no coincide con la ID de dispositivo correspondiente en `approved_devices`. La sentencia también debe mostrar dos mensajes que transmitan esa información.

Después, escribe una sentencia `else` que gestione el caso cuando `username` no pertenezca a `approved_users`. Esta sentencia también debe mostrar un mensaje que transmita que el usuario no está aprobado.

Asegúrate de reemplazar cada `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[8]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]
```

```
# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
 ↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Asigna a `username` un nombre de usuario

username = "sgilmore"

# Asigna a `device_id` una ID de dispositivo

device_id = "4n482ts"

# Asigna a `ind` la indexación de `username` en `approved_users`

ind = approved_users.index(username)

# Sentencia if# Si `username` pertenece a `approved_users`, y si el elemento en␣
 ↪`ind` en `approved_devices` coincide con `device_id`,
# entonces se muestra un mensaje de que el nombre de usuario está aprobado,
# seguido de un mensaje de que tiene el dispositivo correcto

if username in approved_users and device_id == approved_devices[ind]:
    print("El usuario", username, "está aprobado para acceder al sistema.")
    print(device_id, "es el dispositivo asignado para", username)

# Sentencia elif
# Se ocupa del caso cuando `username` pertenece a `approved_users` pero el␣
 ↪elemento en `ind` en `approved_devices` no coincide con `device_id`,
# y muestra dos mensajes en consecuencia

elif username in approved_users and device_id != approved_devices[ind]:
    print("El usuario", username, "está aprobado para acceder al sistema,␣
 ↪pero", device_id, "no es su dispositivo asignado.")

# Sentencia else
# Se ocupa del caso cuando `username` no pertenece a `approved_users`,
# y muestra un mensaje en consecuencia

else:
    print("El usuario", username, "no está aprobado para acceder al sistema.")
```

```
El usuario sgilmore está aprobado para acceder al sistema.
4n482ts es el dispositivo asignado para sgilmore
```

Pista 1

En la sentencia `elif`, utiliza el operador `in` para verificar si `username` pertenece a `approved_users`, utiliza un operador de comparación para verificar si el elemento en `ind` en `approved_devices` no coincide con `device_id`, y utiliza un operador lógico para conectar estas dos condiciones para verificar si se cumplen.

Pista 2

En la sentencia `elif`, utiliza el operador `in` para verificar si `username` pertenece a `approved_users`, utiliza un operador de comparación `!=` para verificar si el elemento en `ind` en `approved_devices` no coincide con `device_id`, y utiliza el operador lógico `and` para conectar estas dos condiciones para verificar si se cumplen.

Pregunta 8 ¿Qué observas en el resultado cuando `username` es `"sgilmore"` y `device_id` es `"4n482ts"`?

Cuando `username` es `"sgilmore"` y `device_id` es `"4n482ts"`, el resultado consiste en `El nombre de usuario sgilmore está aprobado para acceder al sistema.` en la primera línea y `4n482ts es el dispositivo asignado para sgilmore` en la segunda. Si `username` no estaba en la lista `approved_devices`, el resultado sería un mensaje de que el usuario no está aprobado para acceder al sistema.

Si `username` estaba en la lista `approved_devices` pero `device_id` no correspondía con `username`, el resultado sería un mensaje de que el usuario está aprobado para acceder al sistema pero la ID del dispositivo no está asignada a ellos.

1.7 Tarea 9

En esta tarea, completarás tu algoritmo desarrollando una función que utilice parte del código que escribiste en tareas anteriores. Esto automatizará el proceso de inicio de sesión.

Existen varias formas de utilizar condicionales para automatizar el proceso de inicio de sesión. En el siguiente código, se utiliza una sentencia condicional anidada para lograr los objetivos del algoritmo. Hay una sentencia condicional dentro de otra. La sentencia condicional externa se ocupa del caso cuando se aprueba el `username` y el caso cuando no se aprueba el `username`. La sentencia condicional interna, que se coloca dentro de la primera sentencia `if`, se ocupa del caso cuando se aprueba el `username' y la`device_i`des correcta, así como el caso cuando se aprueba el`usernamey `la`device_id‘ es incorrecta.

Para completar esta tarea, debes definir una función llamada `login` que reciba dos parámetros, `username` y `device_id`. Después, llama a la función y pasa diferentes combinaciones de nombre de usuario e ID de dispositivo para experimentar y observar su comportamiento. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[9]:
```
# Asigna a `approved_users` una lista de nombres de usuario aprobados

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]
```

```
# Asigna a `approved_devices` una lista de ID de dispositivo que corresponden a␣
 ↪los nombres de usuario en `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Define una función llamada `login` que reciba dos parámetros, `username` y␣
 ↪`device_id`

def login(username, device_id):

    # Si `username` pertenece a `approved_users`,

    if username in approved_users:

        # entonces muestra "El usuario ______ está aprobado para acceder al␣
 ↪sistema.",

    print("El usuario", username, "está aprobado para acceder al sistema.")

# asigna a `ind` la indexación de `username` en `approved_users`

        ind = approved_users.index(username)

        # y ejecuta la siguiente sentencia condicional
        # Si `device_id` coincide con el elemento en la indexación `ind` en␣
 ↪`approved_devices`,

        if device_id == approved_devices[ind]:

          # entonces muestra "______ es el dispositivo asignado para ______"

    print(device_id, "es el dispositivo asignado para", username)

        # De lo contrario,

        else:

          # muestra "______ no es su dispositivo asignado"

          print(device_id, "no es su dispositivo asignado.")

    # De lo contrario (parte de la sentencia condicional externa y gestiona el␣
 ↪caso cuando `username` no pertenece a `approved_users`),

    else:

        # Muestra "El usuario ______ no está aprobado para acceder al sistema."
```

```
        print("El nombre de usuario", username, "no está aprobado para acceder␣
 ↪al sistema.")

# Llama a la función que acabas de definir para experimentar con diferentes␣
 ↪combinaciones de nombre de usuario y device_id

login("bmoreno", "hl0s5o1")
login("elarson", "r2s5r9g")
login("abernard", "4n482ts")
```

```
El usuario bmoreno está aprobado para acceder al sistema.
hl0s5o1 es el dispositivo asignado para bmoreno
El usuario elarson está aprobado para acceder al sistema.
r2s5r9g no es su dispositivo asignado.
El nombre de usuario abernard no está aprobado para acceder al sistema.
```

Pista 1

Utiliza la palabra clave `def` para iniciar la definición de la función.

Pista 2

Después de la palabra clave `def`, especifica el nombre de la función, seguido de paréntesis y dos puntos. Dentro de los paréntesis, especifica los parámetros que toma la función.

Para llamar a la función, escribe el nombre de la función, seguido de paréntesis, y pasa el nombre de usuario y la ID del dispositivo con los que deseas experimentar.

Pista 3

Después de la palabra clave `def`, escribe `login(username, device_id):` para completar el encabezado de definición de la función.

Para llamar a la función, escribe `login()` y pasa el nombre de usuario y la ID del dispositivo con los que deseas experimentar, separados por una coma. Ten en cuenta que los argumentos que pasas son datos de tipo cadena.

Pregunta 9 Después de que Python ingresa en la sentencia condicional interna, ¿qué sucede cuando la `device_id` es correcta? ¿Y cuando la `device_id` es incorrecta?

Lo que sucede después de que Python ingresa en la sentencia condicional interna es lo siguiente:

Cuando la `device_id` es correcta, la condición interna `if` se evalúa a `True`, y se muestra un mensaje de que la ID de dispositivo está asignada al usuario.

Cuando la `device_id` es incorrecta, la condición interna `if` se evalúa a `False`, Python ingresa en el caso `else` y se muestra un mensaje de que la ID de dispositivo no es el dispositivo asignado al usuario.

1.8 Conclusión

¿A qué conclusiones clave llegaste con este laboratorio?

— La indexación de una lista es similar a la indexación de una cadena. Los valores de indexación comienzan en 0. — El método `.append()` te ayuda a agregar nuevos elementos al final de las listas. — El método `.remove()` te ayuda a eliminar elementos de las listas. — El método `.index()` se puede utilizar en diferentes tipos de secuencias. Se pueden utilizar no solo con cadenas, sino también con listas. — Con una lista, el método `.index()` te permite identificar la posición en la que se encuentra un elemento especificado en ella. — Si dos listas contienen información que se corresponden entre sí en un orden específico, puedes utilizar índices para emparejar elementos de ambas listas. — Las funciones se pueden utilizar para desarrollar algoritmos. Al definir una función, debes especificar los parámetros que toma y las acciones que debe ejecutar.