Sprint 1: Python básico

Capítulo 5/5

Proyecto

**Proyecto Sprint 1**

Daniel

¡Felicidades! Estás a punto de comenzar a trabajar en tu primer proyecto. ¡Estoy muy orgulloso de ti! Revisemos la descripción del proyecto y veamos qué debes hacer.

**Descripción del proyecto**

Trabajarás como profesional de datos júnior para la empresa de comercio electrónico Store 1. El objetivo de la tienda es obtener información valiosa de su base de clientes y ha establecido varios requisitos que quiere que cumplas. Al obtener esta información, planean aprovecharla para crear campañas de marketing específicas basadas en la edad de sus clientes, sus compras previas y sus categorías de compra.

Este es un proyecto complejo que se divide en dos partes:

A diagram with black text and black circles

Description automatically generated

Este enfoque te permite incrementar tu conocimiento gradualmente y satisfacer todas las necesidades del cliente sin abrumarte. Al completar este proyecto, obtendrás experiencia práctica en el manejo de tareas de análisis de datos del mundo real, lo que mejorará tus habilidades y te preparará para retos más complejos en la segunda parte del proyecto.

**Instrucciones para completar el proyecto**

Daniel

Se te proporciona un Jupyter notebook para que escribas tu código. Contiene un conjunto de datos con una descripción e instrucciones detalladas para resolver cada uno de los pasos del proyecto. Sigue todos los pasos y llena cada celda de código requerida para completar el proyecto.

**Etapas del proyecto:**

1. Lectura del proyecto:
   * Lee cuidadosamente la descripción del conjunto de datos y revisa todos los pasos del proyecto para que te des una idea de lo que harás.
2. Resolución de los pasos del proyecto:
   * Resuelve cada paso escribiendo tu propio código en la celda de código adecuada. Recuerda agregar comentarios a tu código para que el proceso de revisión sea más fácil y rápido.
3. Revisión de tu trabajo:
   * Revisa los pasos de escritura de código para asegurarte de que todo esté resuelto y el código esté bien escrito. Agrega cualquier información adicional que consideres que puede ser de ayuda antes de enviar tu proyecto para revisión.

**Lista de comprobación**

Daniel

Si llegaste hasta aquí, significa que ya completaste tu proyecto. ¡Buen trabajo! Pero antes de entregarlo, asegúrate de que:

* Completaste todas las etapas del proyecto.
* Llenaste cada celda requerida.

Daniel

No esperes a que tu proyecto esté revisado. Avanza al segundo sprint y sigue aprendiendo sobre temas de Python más avanzados: ¡bucles, sentencias condicionales, diccionarios y funciones!

Valoración

Siguiente

Sprint 1: Python básico

**Ejercicio**

En caso de que sientas que necesitas más práctica antes de abordar el proyecto, he preparado algunas tareas adicionales para ti. Este Jupyter notebook contiene ejercicios similares a los del proyecto. Intenta resolverlos y revisa la clave de respuestas que se proporciona al final. La solución para estos ejercicios se encuentra al final del Jupyter notebook, así que puedes comprobarlos tras resolverlos. Esto es completamente opcional, puedes ir directamente al proyecto si te sientes preparado. ¡Buena suerte y disfruta trabajando en tu proyecto!

Enviar

Valoración[Siguiente](https://tripleten.com/trainer/data-scientist/lesson/d90cc5a1-f8eb-4dcd-97ed-8a3a863b7f1c/)

Los servicios en línea de la Administración empezaron a analizar el comportamiento y a tomar decisiones basadas en datos para mejorar la experiencia en línea.

Tu primera tarea consiste en evaluar la calidad de una muestra de datos recogidos y prepararla para futuros análisis.

**Paso 1**

Vamos a evaluar la calidad de los datos recogidos sobre los ciudadanos. Te han pedido que revises los datos recopilados y propongas cambios. A continuación, verás datos sobre un usuario o una usuaria en particular; los revisaremos e identificaremos cualquier posible problema.

In [ ]:

id\_card\_no **=** 3428563

name\_surname **=** 'Joseph\_Kobe\_Steeler '

age **=** '74'

subscriptions **=** ['TAXES', 'social security']

Examina las opciones que aparecen a continuación y selecciona las que reflejen los problemas que has detectado en los datos recogidos sobre el ciudadano anterior.

**Opciones:**

1. El tipo de datos de id\_card\_no no debe cambiarse de entero a cadena.
2. La variable name\_surname contiene cadenas con espacios innecesarios y guiones bajos entre las partes del nombre.
3. El tipo de datos de age es incorrecto.
4. La lista subscriptions contiene cadenas en mayúsculas y minúsculas para las distintas entradas. No hay ningún problema en que las listas sean así.

Escribe en la siguiente celda Markdown los números de las opciones que has identificado como problemas. Si has encontrado varios problemas, sepáralos con comas.

In [ ]:

"\*\*Escribe aquí tu respuesta y explica tus argumentos:\*\*

**Paso 2**

Empecemos a hacer cambios para los problemas que hemos identificado. Primero, necesitamos corregir los problemas de la variable user\_name Tenemos que eliminar los espacios y sustituir los guiones bajos por espacios.

In [4]:

name\_surname **=** 'Joseph\_Kobe\_Steeler '

name\_surname **=**name\_surname.strip()

name\_surname **=** name\_surname.replace('\_',' ')

​

print(name\_surname)

Joseph Kobe Steeler

**Paso 3**

Vamos a dividir el nombre completo almacenado en la variable name\_surname en subcadenas. Necesitamos obtener una lista que contenga los nombres por separado.

In [5]:

name\_surname **=** 'Joseph Kobe Steeler'

name\_split **=**name\_surname.split()

​

print(name\_split)

['Joseph', 'Kobe', 'Steeler']

**Paso 4**

Arreglemos la variable age, ya que tiene un tipo de datos incorrecto.

In [6]:

age **=** '74'

​

print(type(age))

age **=** int(age)

​

print(age)

print(type(age))

<class 'str'>

74

<class 'int'>

**Paso 5**

¿Y qué pasaría si en lugar de la edad tuviéramos otra cosa que no fuera exactamente un número? No podríamos convertirlo a int. Si lo intentáramos, nuestro sistema se bloquearía.

Vamos a escribir un fragmento de código para evitar que el sistema se bloquee, en su lugar mostrará un mensaje de error amigable.

In [ ]:

age **=** 'seventy four'

​

**try**:

age **=**int(age)

**except**:

print('Age should be a numerical value.')

Age should be a numerical value.

**Paso 6**

Supongamos ahora que la Administración lanza una cuenta corriente (CA, del inglés "current account") en línea para los ciudadanos, en la que estos pueden consultar las devoluciones de impuestos y otras subvenciones como números enteros positivos, y los impuestos adeudados y las cotizaciones a la Seguridad Social como números negativos. Nos gustaría saber cuánto tiene que recibir o pagar cada ciudadano, así como el mayor valor que el ciudadano recibirá Y pagará. Para eso tenemos lo siguiente:

* El importe total sería la suma de todos los importes. Si es negativo, el ciudadano tendrá que pagar impuestos, y si es positivo, recibirá dinero.
* El importe mínimo será el mayor impuesto/contribución que tenga que pagar el ciudadano, ya que aparecen como números negativos.
* El importe máximo será la mayor bonificación/subvención fiscal que pueda recibir el ciudadano.

Vamos a calcular estos valores y mostrarlos en la pantalla:

In [ ]:

citizen\_CA **=** [**-**465, 156, **-**567, **-**6051, 8607]

​

total\_amount **=** *# escribe tu código aquí*

max\_tax\_due **=** *# escribe tu código aquí*

max\_rebate **=** *# escribe tu código aquí*

​

print(total\_amount)

print(max\_tax\_due)

print(max\_rebate)

1680

-6051

8607

**Ejercicio 7**

La Administración quiere enviar un correo electrónico con el siguiente texto al ciudadano, notificando el inicio de la devolución de impuestos: Estimado Citizen\_name, ha gastado más de 10,000 en el mes month\_nº. Su devolución de impuestos se inicia el mes que viene, el month\_nº+1.

In [8]:

name\_surname **=** 'Joseph Kobe Steeler'

months **=** 5

​

message **=** f'Estimado {name\_surname}, ha gastado más de 10,000 en el mes de {months}. Su devolución de impuestos comienza en el mes siguiente, {months**+**1}.'

print(message)

Estimado Joseph Kobe Steeler, ha gastado más de 10,000 en el mes de 5. Su devolución de impuestos comienza en el mes siguiente, 6.

Presentaremos un caso cotidiano real. El banco ABC ha lanzado recientemente su servicio de banca privilegiada para clientes de alto poder adquisitivo.

Nuestro objetivo principal es organizar el almacenamiento de información para los nuevos clientes. A continuación, investigaremos sobre los primeros clientes para comprender mejor quiénes son.

**1  Ejercicio 1**

La información sobre cada nuevo cliente deberá facilitarse en el siguiente orden: documento de identidad, nombre, edad, ingresos netos anuales y ocupación. Esta es la lista que contiene los datos de Jack: 32456, 'Jack Wilson', 32, 150000, 'Healthcare'. Crea la variable client\_info con la información sobre Jack e imprímela.

In [ ]:

client\_info **=** *# escribe tu código aquí*

print(client\_info)

**2  Ejercicio 2**

Para organizar el almacenamiento de la información de varios clientes, hemos creado la variable clients y le hemos asignamos una lista vacía.

La información sobre Jack se almacena en la variable client\_info. El objetivo es añadir esta información a la lista existente clients como una lista anidada. Hazlo. Cuando acabes, no te olvides de imprimir la lista clients.

In [ ]:

clients **=** []

client\_info **=** [32456, 'Jack Wilson', 32, 150000, 'Healthcare']

​

*# escribe tu código aquí*

​

print(clients)

**3  Ejercicio 3**

Se ha añadido información sobre una nueva clienta, Nina Brown, a la lista de clients. El banco está interesado en su ingreso neto anual, que se sitúa en la 4ª posición de la lista. Nuestro objetivo es extraer esta información, asignarla a la variable client\_income e imprimirla.

In [ ]:

clients **=** [

[32456, 'Jack Wilson', 32, 150000, 'Healthcare'],

[34591, 'Nina Brown', 45, 250000, 'Telecom']

]

​

client\_income **=** *# escribe tu código aquí*

print(client\_income)

**4  Ejercicio 4**

El número de clientes crece rápidamente. Dos clientes más se han unido recientemente al servicio y ya figuran en la lista de clients. Sin embargo, un empleado del banco ha cometido un error tipográfico al añadir la información del cliente, concretamente en el campo de ocupación de Brian Pérez ('Transportatiion'). Dado que esta entrada no es válida, debe eliminarse de la lista de clients. Tu objetivo es eliminar toda la sublista errónea de la lista clients e imprimir esta última.

In [ ]:

clients **=** [

[32456, 'Jack Wilson', 32, 150000, 'Healthcare'],

[34591, 'Nina Brown', 45, 250000, 'Telecom'],

[37512, 'Alex Smith', 39, 210000, 'IT'],

[39591, 'Brian Perez', 29, 340000, 'Transportatiion']

]

​

*# escribe tu código aquí*

​

print(clients)

**5  Ejercicio 5**

La información sobre Brian Pérez ha sido actualizada. Además, otros dos clientes se han unido recientemente al servicio. Ahora el banco quiere analizar a los clientes para determinar la distribución de edades de mayor a menor.

En el precódigo, encontrarás la variable ages que contiene las edades de todos los clientes actuales. Nuestro objetivo es ordenar esta lista en orden descendente e imprimirla. Para ordenar, utiliza el método sort().

In [ ]:

ages **=** [32, 45, 39, 29, 25, 32]

​

*# escribe tu código aquí*

​

print(ages)

**6  Ejercicio 6**

El departamento de seguridad del banco nos pidió una lista con los nombres de nuestros primeros clientes. Actualmente, esta información está disponible como una cadena en la que todos los nombres están separados por comas y que se almacena como la variable names.

Para ayudar, vamos a convertir la cadena en una lista llamada names\_split, donde cada elemento representa el nombre de un cliente. No olvides mostrar la lista resultante.

In [ ]:

names **=** 'Jack Wilson,Nina Brown,Alex Smith,Brian Perez,David Martinez,John Kim'

​

names\_split **=** *# escribe tu código aquí*

​

print(names\_split)

Los servicios en línea de la Administración empezaron a analizar el comportamiento y a tomar decisiones basadas en datos para mejorar la experiencia en línea.

Tu primera tarea consiste en evaluar la calidad de una muestra de datos recogidos y prepararla para futuros análisis.

**Paso 1**

Vamos a evaluar la calidad de los datos recogidos sobre los ciudadanos. Te han pedido que revises los datos recopilados y propongas cambios. A continuación, verás datos sobre un usuario o una usuaria en particular; los revisaremos e identificaremos cualquier posible problema.

In [ ]:

id\_card\_no **=** 3428563

name\_surname **=** 'Joseph\_Kobe\_Steeler '

age **=** '74'

subscriptions **=** ['TAXES', 'social security']

Examina las opciones que aparecen a continuación y selecciona las que reflejen los problemas que has detectado en los datos recogidos sobre el ciudadano anterior.

**Opciones:**

1. El tipo de datos de id\_card\_no no debe cambiarse de entero a cadena.
2. La variable name\_surname contiene cadenas con espacios innecesarios y guiones bajos entre las partes del nombre.
3. El tipo de datos de age es incorrecto.
4. La lista subscriptions contiene cadenas en mayúsculas y minúsculas para las distintas entradas. No hay ningún problema en que las listas sean así.

Escribe en la siguiente celda Markdown los números de las opciones que has identificado como problemas. Si has encontrado varios problemas, sepáralos con comas.

**Escribe aquí tu respuesta y explica tus argumentos:**

Veamos la primera opción. Un tarjeta de identificación suele ser un número, y ahora mismo es un entero, pero podría ser necesario escribir letras también; y las identificaciones son secuencias a las que no se aplican operaciones aritméticas, por lo que deberíamos guardarlas como cadenas. Esta opción no es correcta y no la vamos a anotar.

Para la segunda opción, la variable name\_surname realmente contiene cadenas con espacios y guiones bajos y esto es algo que tenemos que arreglar, ya que un nombre de persona normalmente no tiene guiones bajos ni tampoco espacios. Anotamos esta opción como una de las respuestas.

La tercera opción indica que el tipo de edad es incorrecto tal y como es ahora mismo, una cadena. No debemos indicar las edades como cadena, ya que las utilizamos para hacer cálculos y la edad siempre es un número. Así que anotamos esta opción como una de las respuestas.

En la cuarta opción se afirma que tenemos una lista con cadenas en mayúsculas y minúsculas, lo cual es cierto. Pero también se dice que no hay ningún problema en que las listas se almacenen así y esto no es cierto, ya que añade dificultad a la hora de seguir analizando los datos, por ejemplo, para encontrar duplicados. Deberíamos cambiar estos elementos para utilizar solo mayúsculas o minúsculas, y la mejor práctica es que todo se escriba en minúsculas. Por lo tanto, tenemos que cambiarlo. Anotamos esta opción como una de las respuestas.

Por eso, el estudiante debe escribir aquí la respuesta:

2, 3, 4.

Para el nº 2: Explicación de por qué debe corregirse

Para el nº 3: Explicación de por qué debe corregirse

Para el nº 4: Explicación de por qué debe corregirse

**Paso 2**

Empecemos a hacer cambios para los problemas que hemos identificado. Primero, necesitamos corregir los problemas de la variable user\_name Tenemos que eliminar los espacios y sustituir los guiones bajos por espacios.

In [ ]:

name\_surname **=** 'Joseph\_Kobe\_Steeler '

name\_surname **=** name\_surname.strip()

name\_surname **=** name\_surname.replace('\_', ' ')

​

print(name\_surname)

Joseph Kobe Steeler

**Paso 3**

Vamos a dividir el nombre completo almacenado en la variable name\_surname en subcadenas. Necesitamos obtener una lista que contenga los nombres por separado.

In [ ]:

name\_surname **=** 'Joseph Kobe Steeler'

name\_split **=** name\_surname.split()

​

print(name\_split)

['Joseph', 'Kobe', 'Steeler']

**Paso 4**

Arreglemos la variable age, ya que tiene un tipo de datos incorrecto.

In [ ]:

age **=** '74'

​

print(type(age))

age **=** int(age)

​

print(age)

print(type(age))

<class 'str'>

74

<class 'int'>

**Paso 5**

¿Y qué pasaría si en lugar de la edad tuviéramos otra cosa que no fuera exactamente un número? No podríamos convertirlo a int. Si lo intentáramos, nuestro sistema se bloquearía.

Vamos a escribir un fragmento de código para evitar que el sistema se bloquee, en su lugar mostrará un mensaje de error amigable.

In [ ]:

age **=** 'seventy four'

​

**try**:

age **=** int(age)

**except**:

print('La edad debe ser un valor numérico')

Age should be a numerical value.

**Paso 6**

Supongamos ahora que la Administración lanza una cuenta corriente (CA, del inglés "current account") en línea para los ciudadanos, en la que estos pueden consultar las devoluciones de impuestos y otras subvenciones como números enteros positivos, y los impuestos adeudados y las cotizaciones a la Seguridad Social como números negativos. Nos gustaría saber cuánto tiene que recibir o pagar cada ciudadano, así como el mayor valor que el ciudadano recibirá Y pagará. Para eso tenemos lo siguiente:

* El importe total sería la suma de todos los importes. Si es negativo, el ciudadano tendrá que pagar impuestos, y si es positivo, recibirá dinero.
* El importe mínimo será el mayor impuesto/contribución que tenga que pagar el ciudadano, ya que aparecen como números negativos.
* El importe máximo será la mayor bonificación/subvención fiscal que pueda recibir el ciudadano.

Vamos a calcular estos valores y mostrarlos en la pantalla:

In [ ]:

citizen\_CA **=** [**-**465, 156, **-**567, **-**6051, 8607]

​

total\_amount **=** sum(citizen\_CA)

max\_tax\_due **=** min(citizen\_CA)

max\_rebate **=** max(citizen\_CA)

​

print(total\_amount)

print(max\_tax\_due)

print(max\_rebate)

1680

-6051

8607

**Ejercicio 7**

La Administración quiere enviar un correo electrónico con el siguiente texto al ciudadano, notificando el inicio de la devolución de impuestos: Estimado Citizen\_name, ha gastado más de 10,000 en el mes month\_nº. Su devolución de impuestos se inicia el mes que viene, el month\_nº+1.

In [7]:

name\_surname **=** 'Joseph Kobe Steeler'

months **=** 5

​

message **=** f'Estimado {name\_surname}, ha gastado más de 10,000 en el mes de {months}. Su devolución de impuestos comienza en el mes siguiente, {months**+**1}.'

print(message)

Estimado Joseph Kobe Steeler, ha gastado más de 10,000 en el mes de 5. Su devolución de impuestos comienza en el mes siguiente, 6.

Presentaremos un caso cotidiano real. El banco ABC ha lanzado recientemente su servicio de banca privilegiada para clientes de alto poder adquisitivo.

Nuestro objetivo principal es organizar el almacenamiento de información para los nuevos clientes. A continuación, investigaremos sobre los primeros clientes para comprender mejor quiénes son.

**7  Ejercicio 1**

La información sobre cada nuevo cliente deberá facilitarse en el siguiente orden: documento de identidad, nombre, edad, ingresos netos anuales y ocupación. Esta es la lista que contiene los datos de Jack: 32456, 'Jack Wilson', 32, 150000, 'Healthcare'. Crea la variable client\_info con la información sobre Jack e imprímela.

In [ ]:

lient\_info **=** [32456, 'Jack Wilson', 32, 150000, 'Healthcare']

print(client\_info)

**8  Ejercicio 2**

Para organizar el almacenamiento de la información de varios clientes, hemos creado la variable clients y le hemos asignamos una lista vacía.

La información sobre Jack se almacena en la variable client\_info. El objetivo es añadir esta información a la lista existente clients como una lista anidada. Hazlo. Cuando acabes, no te olvides de imprimir la lista clients.

In [ ]:

clients **=** []

client\_info **=** [32456, 'Jack Wilson', 32, 150000, 'Healthcare']

​

clients.append(client\_info)

​

print(clients)

**9  Ejercicio 3**

Se ha añadido información sobre una nueva clienta, Nina Brown, a la lista de clients. El banco está interesado en su ingreso neto anual, que se sitúa en la 4ª posición de la lista. Nuestro objetivo es extraer esta información, asignarla a la variable client\_income e imprimirla.

In [ ]:

clients **=** [

[32456, 'Jack Wilson', 32, 150000, 'Healthcare'],

[34591, 'Nina Brown', 45, 250000, 'Telecom']

]

​

client\_income **=** clients[1][3]

print(client\_income)

**10  Ejercicio 4**

El número de clientes crece rápidamente. Dos clientes más se han unido recientemente al servicio y ya figuran en la lista de clients. Sin embargo, un empleado del banco ha cometido un error tipográfico al añadir la información del cliente, concretamente en el campo de ocupación de Brian Pérez ('Transportatiion'). Dado que esta entrada no es válida, debe eliminarse de la lista de clients. Tu objetivo es eliminar toda la sublista errónea de la lista clients e imprimir esta última.

In [ ]:

clients **=** [

[32456, "Jack Wilson", 32, 150000, "Healthcare"],

[34591, "Nina Brown", 45, 250000, "Telecom"],

[37512, "Alex Smith", 39, 210000, "IT"],

[39591, "Brian Perez", 29, 340000, "Transportatiion"],

]

​

clients.pop(3)

​

print(clients)

**11  Ejercicio 5**

La información sobre Brian Pérez ha sido actualizada. Además, otros dos clientes se han unido recientemente al servicio. Ahora el banco quiere analizar a los clientes para determinar la distribución de edades de mayor a menor.

En el precódigo, encontrarás la variable ages que contiene las edades de todos los clientes actuales. Nuestro objetivo es ordenar esta lista en orden descendente e imprimirla. Para ordenar, utiliza el método sort().

In [ ]:

ages **=** [32, 45, 39, 29, 25, 32]

​

ages.sort(reverse**=True**)

print(ages)

**12  Ejercicio 6**

El departamento de seguridad del banco nos pidió una lista con los nombres de nuestros primeros clientes. Actualmente, esta información está disponible como una cadena en la que todos los nombres están separados por comas y que se almacena como la variable names.

Para ayudar, vamos a convertir la cadena en una lista llamada names\_split, donde cada elemento representa el nombre de un cliente. No olvides mostrar la lista resultante.

In [ ]:

names **=** 'Jack Wilson,Nina Brown,Alex Smith,Brian Perez,David Martinez,John Kim'

​

names\_split **=** names.split(',')

​

print(names\_split)

Siguiente

[{"metadata":{},"cell\_type":"markdown","source":"# ¡Hola !🙋🏻‍♂️\n\nTe escribe Lisandro Saez, soy revisor de código en Tripleten y tengo el agrado de revisar el proyecto que entregaste.\n\nPara simular la dinámica de un ambiente de trabajo, si veo algún error, en primer instancia solo los señalaré, dándote la oportunidad de encontrarlos y corregirlos por tu cuenta. En un trabajo real, el líder de tu equipo hará una dinámica similar. En caso de que no puedas resolver la tarea, te daré una información más precisa en la próxima revisión.\n\nEncontrarás mis comentarios más abajo - \*\*por favor, no los muevas, no los modifiques ni los borres\*\*.\n\n¿Cómo lo voy a hacer? Voy a leer detenidamente cada una de las implementaciones que has llevado a cabo para cumplir con lo solicitado. Verás los comentarios de esta forma:\n\n

\n**Comentario del revisor** \n\nSi todo está perfecto.\n

\n\n\n

\n**Comentario del revisor** \n\nSi tu código está bien pero se puede mejorar o hay algún detalle que le hace falta. Se aceptan uno o dos comentarios de este tipo en el borrador, pero si hay más, deberías hacer las correcciones. Es como una tarea de prueba al solicitar un trabajo: muchos pequeños errores pueden hacer que un candidato sea rechazado.\n

\n\n

\n\n**Comentario del revisor** \n\nSi de pronto hace falta algo o existe algún problema con tu código o conclusiones.\n

\n\nPuedes responderme de esta forma (no te preocupes, no es obligatorio):\n\n

\n**Respuesta del estudiante** \n\nHola, muchas gracias por tus comentarios y la revisión.\n

\n\n¡Empecemos!\n"},{"metadata":{"id":"UCjlyTyyJm9R"},"cell\_type":"markdown","source":"\n\n

Como parte del equipo de análisis, lo primero que debes hacer es evaluar la calidad de una muestra de datos recopilados y prepararla para analizarla posteriormente. Después, en la segunda parte de este proyecto en el segundo sprint, desarrollarás más tus habilidades y harás tu primer análisis completo, respondiendo a las necesidades del cliente.

Estos son los datos que el cliente nos proporcionó. Tienen el formato de una lista de Python, con las siguientes columnas de datos:

* **user\_id:** Identificador único para cada usuario.
* **user\_name:** El nombre del usuario.
* **user\_age:** La edad del usuario.
* **fav\_categories:** Categorías favoritas de los artículos que compró el usuario, como 'ELECTRONICS', 'SPORT' y 'BOOKS' (ELECTRÓNICOS, DEPORTES y LIBROS), etc.
* **total\_spendings:** Una lista de números enteros que indican la cantidad total gastada en cada una de las categorías favoritas.

In [ ]:

users **=** [

['32415', ' mike\_reed ', 32.0, ['ELECTRONICS', 'SPORT', 'BOOKS'], [894, 213, 173]],

['31980', 'kate morgan', 24.0, ['CLOTHES', 'BOOKS'], [439, 390]],

['32156', ' john doe ', 37.0, ['ELECTRONICS', 'HOME', 'FOOD'], [459, 120, 99]],

['32761', 'SAMANTHA SMITH', 29.0, ['CLOTHES', 'ELECTRONICS', 'BEAUTY'], [299, 679, 85]],

['32984', 'David White', 41.0, ['BOOKS', 'HOME', 'SPORT'], [234, 329, 243]],

['33001', 'emily brown', 26.0, ['BEAUTY', 'HOME', 'FOOD'], [213, 659, 79]],

['33767', ' Maria Garcia', 33.0, ['CLOTHES', 'FOOD', 'BEAUTY'], [499, 189, 63]],

['33912', 'JOSE MARTINEZ', 22.0, ['SPORT', 'ELECTRONICS', 'HOME'], [259, 549, 109]],

['34009', 'lisa wilson ', 35.0, ['HOME', 'BOOKS', 'CLOTHES'], [329, 189, 329]],

['34278', 'James Lee', 28.0, ['BEAUTY', 'CLOTHES', 'ELECTRONICS'], [189, 299, 579]],

]

​

**Paso 1**

Store 1 tiene como objetivo garantizar la coherencia en la recopilación de datos. Como parte de esta iniciativa, se debe evaluar la calidad de los datos recopilados sobre los usuarios y las usuarias. Te han pedido que revises los datos recopilados y propongas cambios. A continuación verás datos sobre un usuario o una usuaria en particular; revisa los datos e identifica cualquier posible problema.

In [1]:

user\_id **=** '32415'

user\_name **=** ' mike\_reed '

user\_age **=** 32.0

fav\_categories **=** ['ELECTRONICS', 'SPORT', 'BOOKS']

**Opciones:**

1. Se debe cambiar el tipo de dato de user\_id de cadena a entero.
2. La variable user\_name contiene una cadena que tiene espacios innecesarios y un guion bajo entre el nombre y el apellido.
3. El tipo de dato de user\_age es correcto y no hay necesidad de convertirlo.
4. La lista fav\_categories contiene cadenas en mayúsculas. En lugar de ello, debemos convertir los valores de la lista en minúsculas.

Para cada una de las opciones, escribe en la siguiente celda markdown si la identificaste como un problema real en los datos o no. Justifica tu razonamiento. Por ejemplo, si crees que la primera opción es correcta, escríbelo y explica por qué piensas que es correcta.

#Ob1. No es necesario cambiar el dato a entero debido a que es información asignada, que no participa en operaciones matemáticas. #Ob2. Eliminar espacios innecesarios y guiones para uniformidad de formato de la información. #Ob3. Posiblemente no representa problemas el tipo floot sin embargo la edad no suele ser representada así por lo que sería conveniente cambiar a int. #Ob4. Es recomendable mantener el formato de la información uniforme por lo que, lo conveniente es utilizar minúsculas.

**Comentario del revisor**

Gran trabajo! Es importante que antes de encarar cualquier problema que queramos resolver con Python, nos paremos a pensar bien en qué consisten los datos con los que vamos a trabajar!

**Paso 2**

Vamos a implementar los cambios que identificamos. Primero, necesitamos corregir los problemas de la variable user\_name Como vimos, tiene espacios innecesarios y un guion bajo como separador entre el nombre y el apellido; tu objetivo es eliminar los espacios y luego reemplazar el guion bajo con el espacio.

In [2]:

user\_name **=** ' mike\_reed '

user\_name **=**user\_name.strip()

user\_name **=**user\_name.replace('\_',' ')

​

print(user\_name)

mike reed

**\*\*\*\***Hint\*\***\*\***

Existe un método, strip(), que puede eliminar espacios al principio y al final de una cadena. Además, el método replace() se puede usar para reemplazar una parte de una cadena. En este caso, queremos reemplazar los guiones bajos (\_) con espacios.

**Paso 3**

Luego, debemos dividir el user\_name (nombre de usuario o usuaria) actualizado en dos subcadenas para obtener una lista que contenga dos valores: la cadena para el nombre y la cadena para el apellido.

In [3]:

user\_name **=** 'mike reed'

name\_split **=**user\_name.split()

​

print(name\_split)

['mike', 'reed']

**\*\*\*\***Hint\*\***\*\***

El método split() se utiliza para dividir una cadena. Por defecto, utiliza un espacio como separador.

**Comentario del revisor**

Gran uso de los métodos split() y strip()!

**Paso 4**

¡Genial! Ahora debemos trabajar con la variable user\_age. Como ya mencionamos, esta tiene un tipo de datos incorrecto. Arreglemos este problema transformando el tipo de datos y mostrando el resultado final.

In [4]:

user\_age **=** 32.0

user\_age **=** int(user\_age)

​

print(user\_age)

32

**\*\*\*\***Hint\*\***\*\***

¿Qué tipo de datos eliminará la parte de coma flotante?

**Paso 5**

Como sabemos, los datos no siempre son perfectos. Debemos considerar escenarios en los que el valor de user\_age no se pueda convertir en un número entero. Para evitar que nuestro sistema se bloquee, debemos tomar medidas con anticipación.

Escribe un código que intente convertir la variable user\_age en un número entero y asigna el valor transformado a user\_age\_int. Si el intento falla, mostramos un mensaje pidiendo al usuario o la usuaria que proporcione su edad como un valor numérico con el mensaje: Please provide your age as a numerical value. (Proporcione su edad como un valor numérico.)

In [5]:

user\_age **=** 'treinta y dos'

​

**try**:

int(user\_age)

**except**:

print("Please provide your age as a numerical value.")

Please provide your age as a numerical value.

**\*\*\*\***Hint\*\***\*\***

Utiliza un bloque try-except para intentar la conversión; si falla, proporciona un mensaje claro indicando que la entrada debe ser numérica.

**Paso 6**

El equipo de dirección de Store 1 te pidió ayudarles a organizar los datos de sus clientes para analizarlos y gestionarlos mejor.

Tu tarea es ordenar esta lista por ID de usuario de forma ascendente para que sea más fácil acceder a ella y analizarla.

In [6]:

users **=** [

['32415', ' mike\_reed ', 32.0, ['ELECTRONICS', 'SPORT', 'BOOKS'], [894, 213, 173]],

['31980', 'kate morgan', 24.0, ['CLOTHES', 'BOOKS'], [439, 390]],

['32156', ' john doe ', 37.0, ['ELECTRONICS', 'HOME', 'FOOD'], [459, 120, 99]],

['32761', 'SAMANTHA SMITH', 29.0, ['CLOTHES', 'ELECTRONICS', 'BEAUTY'], [299, 679, 85]],

['32984', 'David White', 41.0, ['BOOKS', 'HOME', 'SPORT'], [234, 329, 243]],

['33001', 'emily brown', 26.0, ['BEAUTY', 'HOME', 'FOOD'], [213, 659, 79]],

['33767', ' Maria Garcia', 33.0, ['CLOTHES', 'FOOD', 'BEAUTY'], [499, 189, 63]],

['33912', 'JOSE MARTINEZ', 22.0, ['SPORT', 'ELECTRONICS', 'HOME'], [259, 549, 109]],

['34009', 'lisa wilson ', 35.0, ['HOME', 'BOOKS', 'CLOTHES'], [329, 189, 329]],

['34278', 'James Lee', 28.0, ['BEAUTY', 'CLOTHES', 'ELECTRONICS'], [189, 299, 579]],

]

​

users.sort()

​

print(users)

[['31980', 'kate morgan', 24.0, ['CLOTHES', 'BOOKS'], [439, 390]], ['32156', ' john doe ', 37.0, ['ELECTRONICS', 'HOME', 'FOOD'], [459, 120, 99]], ['32415', ' mike\_reed ', 32.0, ['ELECTRONICS', 'SPORT', 'BOOKS'], [894, 213, 173]], ['32761', 'SAMANTHA SMITH', 29.0, ['CLOTHES', 'ELECTRONICS', 'BEAUTY'], [299, 679, 85]], ['32984', 'David White', 41.0, ['BOOKS', 'HOME', 'SPORT'], [234, 329, 243]], ['33001', 'emily brown', 26.0, ['BEAUTY', 'HOME', 'FOOD'], [213, 659, 79]], ['33767', ' Maria Garcia', 33.0, ['CLOTHES', 'FOOD', 'BEAUTY'], [499, 189, 63]], ['33912', 'JOSE MARTINEZ', 22.0, ['SPORT', 'ELECTRONICS', 'HOME'], [259, 549, 109]], ['34009', 'lisa wilson ', 35.0, ['HOME', 'BOOKS', 'CLOTHES'], [329, 189, 329]], ['34278', 'James Lee', 28.0, ['BEAUTY', 'CLOTHES', 'ELECTRONICS'], [189, 299, 579]]]

**\*\*\*\***Hint\*\***\*\***

Puedes utilizar el método sort() en la lista de usuarios para ordenarla de forma ascendente.

**Paso 7**

Tenemos la información de los hábitos de consumo de nuestros usuarios, incluyendo la cantidad gastada en cada una de sus categorías favoritas. La dirección está interesada en conocer la cantidad total gastada por el usuario.

Calculemos este valor y despleguémoslo.

In [7]:

fav\_categories\_low **=** ['electronics', 'sport', 'books']

spendings\_per\_category **=** [894, 213, 173]

​

total\_amount **=** sum(spendings\_per\_category)

​

print(total\_amount)

1280

**\*\*\*\***Hint\*\***\*\***

¿Cuáles son los tres métodos que se pueden aplicar a una lista para calcular sus valores mínimo, máximo y total?

**Comentario del Revisor**

Otra forma de hacer esto mismo podía ser utilizando total\_amount = spendings\_per\_category[0] + spendings\_per\_category[1] + spendings\_per\_category[2]

Te felicito por haber intentado la manera más rapida de hacerlo, pero ten en cuenta también esta forma de sumar ya que puede servirte en algún momento donde sum() quizás no sea el mejor acercamiento al problema.

**Paso 8**

La dirección de la empresa nos pidió pensar en una manera de resumir toda la información de un usuario. Tu objetivo es crear una cadena formateada que utilice información de las variables user\_id, user\_name y user\_age.

Esta es la cadena final que queremos crear: User 32415 is mike who is 32 years old. (El usuario 32415 es Mike, quien tiene 32 años).

In [8]:

user\_id **=** '32415'

user\_name **=** ['mike', 'reed']

user\_age **=** 32

​

user\_info **=** f"User {user\_id} is {user\_name[0]} who is {user\_age} years old."

print(user\_info)

User 32415 is mike who is 32 years old.

**\*\*\*\***Hint\*\***\*\***

Para crear una cadena, puedes utilizar el método format() o f-string. Para extraer el nombre de la lista user\_name, puedes utilizar la segmentación.

**Paso 9**

La dirección también quiere una forma fácil de conocer la cantidad de clientes con cuyos datos contamos. Tu objetivo es crear una cadena formateada que muestre la cantidad de datos de clientes registrados.

Esta es la cadena final que queremos crear: Hemos registrado datos de X clientes.

In [9]:

users **=** [

['32415', ' mike\_reed ', 32.0, ['ELECTRONICS', 'SPORT', 'BOOKS'], [894, 213, 173]],

['31980', 'kate morgan', 24.0, ['CLOTHES', 'BOOKS'], [439, 390]],

['32156', ' john doe ', 37.0, ['ELECTRONICS', 'HOME', 'FOOD'], [459, 120, 99]],

['32761', 'SAMANTHA SMITH', 29.0, ['CLOTHES', 'ELECTRONICS', 'BEAUTY'], [299, 679, 85]],

['32984', 'David White', 41.0, ['BOOKS', 'HOME', 'SPORT'], [234, 329, 243]],

['33001', 'emily brown', 26.0, ['BEAUTY', 'HOME', 'FOOD'], [213, 659, 79]],

['33767', ' Maria Garcia', 33.0, ['CLOTHES', 'FOOD', 'BEAUTY'], [499, 189, 63]],

['33912', 'JOSE MARTINEZ', 22.0, ['SPORT', 'ELECTRONICS', 'HOME'], [259, 549, 109]],

['34009', 'lisa wilson ', 35.0, ['HOME', 'BOOKS', 'CLOTHES'], [329, 189, 329]],

['34278', 'James Lee', 28.0, ['BEAUTY', 'CLOTHES', 'ELECTRONICS'], [189, 299, 579]],

]

​

​

user\_info **=**f"We have registered data on {len(users)} clients."

print(user\_info)

We have registered data on 10 clients.

**\*\*\*\***Hint\*\***\*\***

Para crear una cadena, puedes utilizar el método format() o f-string. Para extraer la cantidad de clientes en la lista, puedes utilizar la función que devuelve la longitud de la lista.

**Paso 10**

Apliquemos ahora todos los cambios a la lista de clientes. Para simplificar las cosas, te proporcionaremos una más corta. Debes:

1. Eliminar todos los espacios iniciales y finales de los nombres, así como cualquier guion bajo.
2. Convertir todas las edades en números enteros.
3. Separar todos los nombres y apellidos en una sublista.

Guarda la lista modificada como una nueva lista llamada users\_clean y muéstrala en la pantalla.

In [10]:

users **=** [

['32415', ' mike\_reed ', 32.0, ['ELECTRONICS', 'SPORT', 'BOOKS'], [894, 213, 173]],

['31980', 'kate morgan', 24.0, ['CLOTHES', 'BOOKS'], [439, 390]],

['32156', ' john doe ', 37.0, ['ELECTRONICS', 'HOME', 'FOOD'], [459, 120, 99]],

]

​

users\_clean **=** [user\_clean\_1,user\_clean\_2,user\_clean\_3]

​

​

*# Procesa al primer usuario*

user\_name\_1 **=**users[0]

user\_age\_1 **=**int(user\_name\_1[2])

user\_name\_1\_2 **=**(user\_name\_1[1].strip()).replace('\_',' ')

user\_clean\_1**=**f"{users[0]}{[user\_name\_1\_2,user\_age\_1]}"

​

*#test*

print(user\_name\_1)

print(user\_age\_1)

print(user\_name\_1\_2)

print(user\_clean\_1)

​

​

*# Procesa al segundo usuario*

user\_name\_2 **=**users[1]

user\_age\_2 **=**int(user\_name\_2[2])

user\_name\_2\_2 **=**(user\_name\_2[1].strip()).replace('\_',' ')

user\_clean\_2**=**f"{users[1]}{[user\_name\_2\_2,user\_age\_2]}"

​

*#test*

print(user\_name\_2)

print(user\_age\_2)

print(user\_name\_2\_2)

print(user\_clean\_2)

​

​

*# Procesa al tercer usuario*

user\_name\_3 **=**users[2]

user\_age\_3 **=**int(user\_name\_3[2])

user\_name\_3\_2 **=**(user\_name\_3[1].strip()).replace('\_',' ')

user\_clean\_3**=**f"{users[2]}{[user\_name\_3\_2,user\_age\_3]}"

​

*#test*

print(user\_name\_3)

print(user\_age\_3)

print(user\_name\_3\_2)

print(user\_clean\_3)

​

print(users\_clean)

​

*#Dudas:*

​

*# Procesa al primer usuario - (dudas en el caso de cualquier usuario)*

user\_name\_1 **=** *# escribe tu código aquí - (traje la lista del usuario a corregir)*

user\_age\_1 **=** *# escribe tu código aquí - (se corrige el formato de edad)*

user\_name\_1 **=** *# escribe tu código aquí - (se corrigen espacios y guiones), (por qué repetir el nombre de la variable "user\_name\_1"?. Las renombré agregando al final "\_2")*

users\_clean.*# escribe tu código aquí - (las tomé como variables en cada usuario agregando "\_1", 2 o 3 según el usuario)(es dificil apegarse a las instrucciones ya que la función append sólo puede agregar un dato al final y el resultado esperado anuncia la sustitución de los latos en el orden original lo cual requeriría las funciones pop e insert)*

​

users\_clean **-** (a partir **del** punto anterior, el objetivo presentar la lista definitiva no es eficaz)

​

Cell In[10], line 53

user\_name\_1 = # escribe tu código aquí - (traje la lista del usuario a corregir)

^

SyntaxError: invalid syntax

**Comentario del Revisor**

Prar próximos sprints, recuerda que todos tus bloques de código deben correr sin errores.

**\*\*\*\***Hint\*\***\*\***

Para procesar a cada usuario, comienza por acceder a los elementos requeridos de la lista de usuarios. Utiliza el método strip() para eliminar los espacios iniciales y finales y el método replace('\_',' ') para reemplazar los guiones bajos por espacios en los nombres. Convierte la edad a un número entero utilizando int(). Separa el nombre completo en nombre y apellido utilizando el método split(). Por último, append (agrega) los datos limpios a la lista users\_clean.

**Comentario del revisor**

Felicitaciones por el excelente trabajo que hiciste durante todo el sprint!