



Recuperación 1

Programación de Redes

Profesor: Gabriel Barron Rodríguez

Alumno: Alan Francisco Emmanuel Aguilar Fuentes

Laboratorios Python 2

Contenido

| | |
|---|----|
| Resúmenes de Laboratorios de Python | 1 |
| 1.1.1.11 RESUMEN DE SECCIÓN..... | 1 |
| 1.2.1.17 RESUMEN DE LA SECCIÓN | 2 |
| 1.3.1.11 RESUMEN DE LA SECCIÓN | 3 |
| 1.4.1.18 RESUMEN DE LA SECCIÓN | 4 |
| 2.1.1.4 RESUMEN DE LA SECCIÓN | 5 |
| 2.2.1.15 RESUMEN DE LA SECCIÓN | 5 |
| 2.3.1.17 RESUMEN DE LA SECCIÓN | 6 |
| 2.4.1.5 RESUMEN DE LA SECCIÓN | 7 |
| 2.5.1.5 RESUMEN DE LA SECCIÓN | 8 |
| 2.6.1.12 RESUMEN DE LA SECCIÓN | 9 |
| 2.7.1.18 RESUMEN DE LA SECCIÓN | 10 |
| 2.8.1.5 RESUMEN DE LA SECCIÓN | 11 |
| 3.1.1.8 RESUMEN DE SECCIÓN..... | 12 |
| 3.2.1.13 RESUMEN DE SECCIÓN..... | 12 |
| 3.3.1.9 RESUMEN DE SECCIÓN..... | 13 |
| 3.4.1.11 RESUMEN DE SECCIÓN..... | 14 |
| 3.5.1.22 RESUMEN DE SECCIÓN..... | 15 |
| 3.5.1.23 RESUMEN DE SECCIÓN..... | 15 |
| 3.6.1.9 RESUMEN DE SECCIÓN..... | 16 |
| 4.1.1.15 RESUMEN DE SECCIÓN..... | 17 |
| 4.2.1.12 RESUMEN DE SECCIÓN..... | 18 |
| 4.3.1.18 RESUMEN DE SECCIÓN..... | 19 |
| 4.4.1.9 RESUMEN DE SECCIÓN..... | 19 |
| 4.5.1.23 RESUMEN DE SECCIÓN..... | 20 |
| 4.6.1.14 RESUMEN DE SECCIÓN..... | 20 |

Resúmenes de Laboratorios de Python

1.1.1.11 RESUMEN DE SECCIÓN.

The screenshot shows a web-based learning environment for Python. On the left, a sidebar displays the user's profile (ALAN FRANCISCO EMMANUEL AGUILAR FUENTES) and navigation links for Área personal, Perfil, and Calificaciones. The main content area is titled 'RESUMEN DE SECCIÓN' and contains two exercises:

Ejercicio 1
Quieres invocar la función `make_money()` contenida en el módulo llamado `mint`. Tu código comienza con la siguiente linea:
`import mint`
¿Cuál es la forma adecuada de invocar a la función?
Revisar
`mint.make_money()`

Ejercicio 2
Quieres invocar la función `make_money()` contenida en el módulo llamado `mint`. Tu código comienza con la siguiente linea:
`from mint import make_money`
¿Cuál es la forma adecuada de invocar a la función?
Revisar
`make_money()`

Below the exercises, there are three numbered notes:

- Si un módulo se importa de la manera anterior y deseas acceder a cualquiera de sus entidades, debes anteponer el nombre de la entidad empleando la **notación con punto**. Por ejemplo:
`import my_module`
`result = my_module.my_function(my_module.my_data)`
El fragmento de código utiliza dos entidades que provienen del módulo `my_module`: una función llamada `my_function()` y una variable con el nombre `my_data`. Ambos nombres **deben tener el prefijo** `my_module`. Ninguno de los nombres de entidad importados entra en conflicto con los nombres idénticos existentes en el namespace de tu código.
- Se te permite no solo importar un módulo como un todo, sino también importar solo entidades individuales de él. En este caso, las entidades importadas **no deben especificar el prefijo** cuando son empleadas. Por ejemplo:

The screenshot shows a continuation of the Python learning platform. The sidebar remains the same. The main content area is titled 'RESUMEN DE SECCIÓN' and contains four exercises:

Ejercicio 3
Has escrito una función llamada `make_money` por tu cuenta. Necesitas importar una función con el mismo nombre del módulo `mint` y no deseas cambiar el nombre de ninguno de tus nombres previamente definidos. ¿Qué variante de la sentencia `import` puede ayudarte con el problema?
Revisar
`# solución de muestra`
`from mint import make_money as make_more_money`

Ejercicio 4
¿Qué forma de invocación de la función `make_money` es válida si tu código comienza con la siguiente linea?
`from mint import *`
Revisar
`make_money()`

Below the exercises, there are two numbered notes:

- La forma más general de la sentencia anterior te permite importar **todas las entidades** ofrecidas por un módulo:
`from my_module import *`
`result = my_function(my_data)`
Nota: la variante de esta importación no se recomienda debido a las mismas razones que antes (la amenaza de un conflicto de nombres es aún más peligrosa aquí).
- Puede cambiar el nombre de la entidad importada "sobre la marcha" utilizando la frase `as` del `import`. Por ejemplo:
`from module import my_function as fun, my_data as dat`
`result = fun(dat)`

1.2.1.17 RESUMEN DE LA SECCIÓN

The screenshot shows a web-based learning platform interface. On the left, there is a sidebar with user information (ALAN FRANCISCO EMMANUEL AGUILAR FUENTES) and navigation links (Área personal, Perfil, Calificaciones). The main content area is titled "RESUMEN DE SECCIÓN". It contains several sections labeled "Ejercicio 1", "Ejercicio 2", "Ejercicio 3", and "Ejercicio 4". Each exercise includes a question, a code editor or text input field, and a "Revisar" button.

Ejercicio 1: ¿Cuál es el valor esperado de la variable `result` después de que se ejecuta el siguiente código?

```
import math  
result = math.e == math.exp(1)
```

Ejercicio 2: (Completa el enunciado) Establecer la semilla del generador con el mismo valor cada vez que se ejecuta tu programa garantiza que ...

Ejercicio 3: ¿Cuál de las funciones del módulo `platform` utilizarías para determinar el nombre del CPU que corre dentro de tu computadora?

Ejercicio 4: ¿Cuál es el resultado esperado del siguiente fragmento de código?

```
import platform  
print(len(platform.python_version_tuple()))
```

This screenshot shows the same learning platform interface, but the content in the main area has changed. The sidebar and navigation links remain the same. The main content area is also titled "RESUMEN DE SECCIÓN". It contains sections labeled "Ejercicio 1", "Ejercicio 2", "Ejercicio 3", and "Ejercicio 4". The content for each exercise is identical to the one in the previous screenshot.

Ejercicio 1: ... los valores pseudoaleatorios emitidos desde el módulo `random` serán exactamente los mismos.

Ejercicio 2: ¿Cuál de las funciones del módulo `platform` utilizarías para determinar el nombre del CPU que corre dentro de tu computadora?

Ejercicio 3: La función `processor()`.

Ejercicio 4: ¿Cuál es el resultado esperado del siguiente fragmento de código?

```
import platform  
print(len(platform.python_version_tuple()))
```

1.3.1.11 RESUMEN DE LA SECCIÓN

The screenshot shows a web-based learning platform interface. On the left, there is a sidebar with a user profile picture and the name "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". Below the profile are three menu items: "Área personal", "Perfil", and "Calificaciones". The main content area has a header "RESUMEN DE SECCIÓN >".

Ejercicio 1: Deseas evitar que el usuario de tu módulo ejecute tu código como un script ordinario. ¿Cómo lograrías tal efecto?

```
import sys
if __name__ == "__main__":
    print("No hagas eso!")
    sys.exit()
```

Ejercicio 2: Algunos paquetes adicionales y necesarios se almacenan dentro del directorio `D:\Python\Project\Modules`. Escribe un código asegurándote de que Python recorra el directorio para encontrar todos los módulos solicitados.

```
import sys
# Toma en cuenta las diagonales invertidas dobles!
sys.path.append("D:\\Python\\Project\\Modules")
```

Ejercicio 3:

Algunos paquetes adicionales y necesarios se almacenan dentro del directorio `D:\Python\Project\Modules`. Escribe un código asegurándote de que Python recorra el directorio para encontrar todos los módulos solicitados.

```
import sys
# Toma en cuenta las diagonales invertidas dobles!
sys.path.append("D:\\Python\\Project\\Modules")
```

We use cookies to improve our service. By continuing to use the site, you agree to our [privacy](#) and [cookie policies](#).

The screenshot shows a similar web-based learning platform interface. The sidebar on the left is identical, displaying the user profile "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES" and the menu items "Área personal", "Perfil", and "Calificaciones". The main content area has a header "RESUMEN DE SECCIÓN >".

Ejercicio 3: El directorio mencionado en el ejercicio anterior contiene un subárbol con la siguiente estructura:

```
abc
  def
    mymodule.py
```

Asumiendo que `D:\Python\Project\Modules` se ha adjuntado con éxito a la lista `sys.path`, escribe una directiva de importación que te permita usar todas las entidades de `mymodule`.

```
import abc.def.mymodule
```

We use cookies to improve our service. By continuing to use the site, you agree to our [privacy](#) and [cookie policies](#).

1.4.1.18 RESUMEN DE LA SECCIÓN

The screenshot shows a web-based learning platform interface. On the left, there's a sidebar with a profile picture and the name "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". Below the profile are three menu items: "Área personal", "Perfil", and "Calificaciones". The main content area has a header "RESUMEN DE SECCIÓN".

Ejercicio 1: ¿De donde proviene el nombre *The Cheese Shop*? [Revisar](#)

Es una referencia a un viejo sketch de Monty Python que lleva el mismo nombre.

Ejercicio 2: ¿Por qué deberías asegurarte de cuál *pip* o *pip3* es el correcto para ti? [Revisar](#)

Cuando Python 2 y Python 3 coexisten en el sistema operativo, es probable que *pip* identifique la instancia de *pip* que trabaja solo con paquetes de Python 2.

Ejercicio 3: ¿Cómo puedes determinar si tu *pip* funciona con Python 2 o Python 3? [Revisar](#)

pip --version te lo dirá.

Ejercicio 4: [Revisar](#)

Comprueba tu mismo cuál de estos funciona en el entorno de tu sistema operativo.

3. Para verificar la versión de *pip*, se deben emitir los siguientes comandos:

```
pip --version
o
pip3 --version
```

4. La lista de las actividades principales de **pip** tiene el siguiente aspecto:

- *pip help operación_comando* → muestra una breve descripción de *pip*.
- *pip list* → muestra una lista de los paquetes instalados actualmente.
- *pip show nombre_del_paquete* → muestra información que incluyen las dependencias del paquete.
- *pip search cadena* → busca en los directorios de PyPi para encontrar paquetes cuyos nombres contengan cadena.
- *pip install nombre* → instala el paquete *nombre* en todo el sistema (espera problemas cuando no tengas privilegios de administrador).
- *pip install --user nombre* → instala *nombre* solo para ti; ningún otro usuario de la plataforma podrá utilizarlo.
- *pip install -U nombre* → actualiza un paquete previamente instalado.
- *pip uninstall nombre* → desinstala un paquete previamente instalado.

The screenshot shows the same web-based learning platform interface for the user "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". The main content area has a header "RESUMEN DE SECCIÓN".

Ejercicio 2: ¿Por qué deberías asegurarte de cuál *pip* o *pip3* es el correcto para ti? [Revisar](#)

Cuando Python 2 y Python 3 coexisten en el sistema operativo, es probable que *pip* identifique la instancia de *pip* que trabaja solo con paquetes de Python 2.

Ejercicio 3: ¿Cómo puedes determinar si tu *pip* funciona con Python 2 o Python 3? [Revisar](#)

pip --version te lo dirá.

Ejercicio 4: Desafortunadamente, no tienes privilegios de administrador. ¿Qué debes hacer para instalar un paquete en todo el sistema? [Revisar](#)

Tienes que consultar a tu administrador del sistema y no intentes hackear tu sistema operativo!

2.1.1.4 RESUMEN DE LA SECCIÓN

The screenshot shows a web browser window with two tabs. The left tab is titled 'Plataforma Educativa Institucional' and the right tab is titled 'Edube Interactive :: PE2 -- Modulo'. The main content area displays a summary for user 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES'. The summary includes:

- Ejercicio 1:** ¿Qué es BOM?
BOM (Byte Order Mark). Una Marca de Orden de Bytes es una combinación especial de bits que anuncia la codificación utilizada por el contenido de un archivo (por ejemplo, UCS-4 o UTF-8).
- Ejercicio 2:** ¿Está Python 3 internacionalizado?
Si, está completamente internacionalizado: podemos usar caracteres UNICODE dentro de nuestro código, leerlos desde la entrada y enviarlos a la salida.

2.2.1.15 RESUMEN DE LA SECCIÓN

The screenshot shows a web browser window with two tabs. The left tab is titled 'Plataforma Educativa Institucional' and the right tab is titled 'Edube Interactive :: PE2 -- Modulo'. The main content area displays a summary for user 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES'. The summary includes:

- Ejercicio 1:** ¿Cuál es la longitud de la siguiente cadena asumiendo que no hay espacios en blanco entre las comillas?
La respuesta es: 3
- Ejercicio 2:** ¿Cuál es el resultado esperado del siguiente código?
El resultado esperado es: ['t', 'e', 'r']
- Ejercicio 3:** 2. La longitud de una cadena está determinada por la función `len()`. El carácter de escape (\) no es contado. Por ejemplo:
`print(len("\\\n"))`
Su salida es: 2.

The screenshot shows a browser window with the URL <https://elearning...>. The page title is "Edube Interactive - PE2 -- Modulo 1". On the left, there's a sidebar with a profile picture and the name "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". Below the profile are links for "Área personal", "Perfil", and "Calificaciones". The main content area has a header "RESUMEN DE SECCIÓN". A blue button labeled "Revisar" is visible. Below it, there's a code editor with the following code:

```
asterisk = '*'
plus = "+"
decoration = (asterisk + plus) * 4 + asterisk
print(decoration)
```

Output: `*****`

Exercise 3: ¿Cuál es el resultado esperado del siguiente código?

```
for ch in "abc":
    print(chr(ord(ch) + 1), end="")
```

Result: `bed`

Exercise 4: El par de funciones `chr()` y `ord()` se pueden utilizar para crear un carácter utilizando su punto de código y para determinar un punto de código correspondiente a un carácter. Las dos expresiones siguientes son siempre verdaderas:

```
chr(ord(character)) == character
ord(chr(codepoint)) == codepoint
```

Exercise 5: Algunas otras funciones que se pueden aplicar a cadenas son:

- `list()` → crea una lista que consta de todos los caracteres de la cadena.
- `max()` → encuentra el carácter con el punto de código máximo.
- `min()` → encuentra el carácter con el punto de código mínimo.

2.3.1.17 RESUMEN DE LA SECCIÓN

The screenshot shows a browser window with the URL <https://elearning...>. The page title is "Edube Interactive - PE2 -- Modulo 1". On the left, there's a sidebar with a profile picture and the name "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". Below the profile are links for "Área personal", "Perfil", and "Calificaciones". The main content area has a header "RESUMEN DE SECCIÓN". A blue button labeled "Revisar" is visible. Below it, there's a code editor with the following code:

```
for ch in "abc123XYZ":
    if ch.isupper():
        print(ch.lower(), end="")
    elif ch.islower():
        print(ch.upper(), end="")
    else:
        print(ch, end="")
```

Result: `AMC123xyz`

Exercise 2: ¿Cuál es el resultado esperado del siguiente código?

```
s1 = '¿Dónde están las nevadas de antaño?'
s2 = s1.split()
print(s2[-2])
```

Result: `de`

Ejercicio 3
¿Cuál es el resultado esperado del siguiente código?

```
the_list = ['¡Dónde', 'están', 'las', 'nevadas?']
s = '+' .join(the_list)
print(s)
```

Ejercicio 4
¿Cuál es el resultado esperado del siguiente código?

```
s = 'Es fácil o imposible'
s = s.replace('fácil', 'difícil').replace('im', '')
print(s)
```

2.4.1.5 RESUMEN DE LA SECCIÓN

Ejercicio 1
¿Cuál de las siguientes líneas describe una condición verdadera?

- 1 'smith' > 'Smith'
- 2 'Smiths' < 'Smith'
- 3 'Smith' > '1000'
- 4 '11' < '8'
- 5

Ejercicio 2
¿Cuál es el resultado esperado del siguiente código?

```
s1 = '¡Dónde están las nevadas de antaño'
s2 = s1.split()
s3 = sorted(s2)
print(s3[1])
```

de

The screenshot shows a web browser window with the URL <https://elearning...>. The page title is "Plataforma Educativa Institucional". The main content area displays a "RESUMEN DE SECCIÓN" section. It contains two code snippets under the heading "Ejercicio 3".

Exercise 1: ¿Cuál es el resultado esperado del siguiente código?

```
s1 = 'Dónde están las nevadas de antaño?'
s2 = s1.split()
s3 = sorted(s2)
print(s3[1])
```

Exercise 2: ¿Cuál es el resultado esperado del siguiente código?

```
s1 = '12.8'
i = int(s1)
s2 = str(i)
f = float(s1)
print(s1 == s2)
```

A blue "Revisar" button is visible below each code snippet.

2.5.1.5 RESUMEN DE LA SECCIÓN

The screenshot shows a web browser window with the URL <https://edube.org/learn/python-essentials-2-esp/resumen-de-secci-oacute-n-20>. The page title is "Course: Programación de Redes". The main content area displays a "2.5.1.5 RESUMEN DE SECCIÓN" section.

Puntos Claves

- Las cadenas son herramientas clave en el procesamiento de datos modernos, ya que la mayoría de los datos útiles son en realidad cadenas. Por ejemplo, el uso de un motor de búsqueda web (que parece bastante trivial en estos días) utiliza un procesamiento de cadenas extremadamente complejo, que involucra cantidades inimaginables de datos.
- El comparar cadenas de forma estricta (como lo hace Python) puede ser muy insatisfactorio cuando se trata de búsquedas avanzadas (por ejemplo, durante consultas extensas a bases de datos). En respuesta a esta demanda, se han creado e implementado una serie de algoritmos de comparación de cadenas **difusas**. Estos algoritmos pueden encontrar cadenas que no son iguales en el sentido de Python, pero que son **similares**.
- Otra forma de comparar cadenas es encontrar su similitud **acústica**, lo que significa un proceso que lleva a determinar si dos cadenas suenan similares (como "echo" y "hecho"). Esta similitud debe establecerse para cada idioma (o incluso dialecto) por separado.
- Un algoritmo utilizado para realizar una comparación de este tipo para el idioma Inglés se llama **Soundex** y se inventó, no lo creas, en 1918. Puedes encontrar más información al respecto aquí: <https://en.wikipedia.org/wiki/Soundex>.
- Debido a la precisión limitada de los datos enteros y flotantes nativos, a veces es razonable almacenar y procesar valores numéricos enormes como cadenas. Esta es la técnica que usa Python cuando se le fuerza a operar con un número entero que consta de una gran cantidad de dígitos.

2.6.1.12 RESUMEN DE LA SECCIÓN

The screenshot shows a web-based programming environment. On the left, there's a sidebar with a user profile picture and the name "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". Below the profile are three menu items: "Área personal", "Perfil", and "Calificaciones". The main area is titled "SUMEN DE SECCIÓN >". It contains two exercises:

Ejercicio 1
¿Cuál es el resultado esperado del siguiente código?

```
try:  
    print("Tratemos de hacer esto")  
    print("#[2])  
    print("Tuvimos éxito!")  
except:  
    print("Remo fallido")  
print("Remo terminado")
```

Below the code is a "Revisar" button and a text input field containing the expected output: "Tratemos de hacer esto", "Remo fallido", and "Remo terminado".

Ejercicio 2
¿Cuál es el resultado esperado del siguiente código?

```
try:  
    print("alpha"[1/0])  
except ZeroDivisionError:  
    print("cero")  
except IndexError:  
    print("indice")  
except:  
    print("algo")
```

Below the code is a "Revisar" button and a text input field containing the expected output: "cero".

This screenshot shows the same web-based programming environment as the previous one, but with different exercise content.

The sidebar on the left remains the same, showing the user profile and menu items.

The main area is titled "SUMEN DE SECCIÓN >". It contains two exercises:

Ejercicio 2
¿Cuál es el resultado esperado del siguiente código?

```
try:  
    print("alpha"[1/0])  
except ZeroDivisionError:  
    print("cero")  
except IndexError:  
    print("indice")  
except:  
    print("algo")
```

Below the code is a "Revisar" button and a text input field containing the expected output: "cero".

At the bottom of the page, there is some explanatory text and a numbered list:

En el mejor caso, se ejecuta uno de los bloques `except`; ninguno de los bloques se ejecuta cuando la excepción generada no coincide con ninguna de las excepciones especificadas.

3. No se puede agregar más de un bloque de `excepción sin nombre` después de los bloques con nombre.

```
# El código que siempre corre suavemente.  
# De vuelta a la normalidad.  
#
```

2.7.1.18 RESUMEN DE LA SECCIÓN

The screenshot shows a browser window with two tabs. The left tab is a user profile page for 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES'. The right tab is titled 'Edube Interactive - P2 -- Modulo 2' and displays 'SUMEN DE SECCIÓN >'.
Ejercicio 1:
¿Cuál es la salida esperada del siguiente código?

```
try:  
    print(1/0)  
except ZeroDivisionError:  
    print("cero")  
except ArithmeticError:  
    print("arit")  
except:  
    print("algo")
```

Review button: cero

Ejercicio 2:
¿Cuál es la salida esperada del siguiente código?

```
try:  
    print(1/0)  
except ArithmeticError:  
    print("arit")  
except ZeroDivisionError:  
    print("cero")  
except:  
    print("algo")
```

The screenshot shows a browser window with two tabs. The left tab is a user profile page for 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES'. The right tab is titled 'Edube Interactive - P2 -- Modulo 2' and displays 'SUMEN DE SECCIÓN >'.
Ejercicio 2:
¿Cuál es la salida esperada del siguiente código?

```
try:  
    print(1/0)  
except ArithmeticError:  
    print("arit")  
except ZeroDivisionError:  
    print("cero")  
except:  
    print("algo")
```

Review button: arit

Ejercicio 3:
¿Cuál es la salida esperada del siguiente código?

```
def foo(x):  
    assert x  
    return 1/x  
  
try:  
    print(foo(0))  
except ZeroDivisionError:
```

The screenshot shows a dual-pane interface. On the left is a "Plataforma Educativa Institucional" window displaying a user profile for "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". The right pane is titled "Edube Interactive :: PE2 -- Modulo 1" and is titled "SUMEN DE SECCIÓN". It contains a code editor with Python code and several exercise sections:

- Ejercicio 1:** ¿Cuál es la salida esperada del siguiente código?
Code:

```
print("algo")
```

Result:
- Ejercicio 3:** ¿Cuál es la salida esperada del siguiente código?
Code:

```
def foo(x):
    assert x
    return 1/x

try:
    print(foo(0))
except ZeroDivisionError:
    print("zero")
except:
    print("algo")
```

Result:

2.8.1.5 RESUMEN DE LA SECCIÓN

The screenshot shows a dual-pane interface. On the left is a "Plataforma Educativa Institucional" window displaying a user profile for "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES". The right pane is titled "Edube Interactive :: PE2 -- Modulo 1" and is titled "SUMEN DE SECCIÓN". It contains three exercises:

- Ejercicio 1:** ¿Cuál de las excepciones se utilizará para proteger al código de ser interrumpido por el uso del teclado?
Result:
- Ejercicio 2:** ¿Cuál es el nombre de la más general de todas las excepciones de Python?
Result:
- Ejercicio 3:** ¿Cuál de las excepciones será generada a través de la siguiente evaluación fallida?
Code:

```
huge_value = 1E250 ** 2
```

Result:

3.1.1.8 RESUMEN DE SECCIÓN.

The screenshot shows a web-based educational platform. On the left, there's a sidebar with a profile picture of Alan Francisco Emmanuel Aguilar Fuentes, followed by links for Área personal, Perfil, and Calificaciones. The main content area has a header "RESUMEN DE SECCIÓN".

Ejercicio 1:
Si asumimos que pitones, víboras y cobras son subclases de la misma superclase, ¿cómo las llamarías?
Revisar
Serpiente, reptil, vertebrado, animal: todas estas respuestas son aceptables.

Ejercicio 2:
Intenta nombrar algunas subclases de la clase Pitón.
Revisar
Pitón India, Pitón de Roca Africana, Pitón Bola, Pitón Birmana: la lista es larga.

Ejercicio 3:
¿Puedes usar la palabra "class" para darle nombre a alguna de tus clases?
Revisar
¡No, no puedes. class es una palabra clave reservada!

Below the exercises, there are numbered steps and code snippets:

- Las relaciones entre clases se muestran como flechas dirigidas **desde la subclase hacia su superclase**.
- Los objetos están equipados con:
 - Un **nombre** que los identifica y nos permite distinguirlos.
 - Un conjunto de **propiedades** (el conjunto puede estar vacío).
 - Un conjunto de **métodos** (también puede estar vacío).
- Para definir una clase en Python, se necesita usar la palabra clave reservada `class`. Por ejemplo:

```
class This_Is_A_Class:  
    pass
```
- Para crear un objeto de la clase previamente definida, se necesita usar la clase como si fuera una función. Por ejemplo:

```
this_is_an_object = This_Is_A_Class()
```

3.2.1.13 RESUMEN DE SECCIÓN.

The screenshot shows a web-based educational platform. On the left, there's a sidebar with a profile picture of Alan Francisco Emmanuel Aguilar Fuentes, followed by links for Área personal, Perfil, and Calificaciones. The main content area has a header "RESUMEN DE SECCIÓN".

Ejercicio 1:
Suponiendo que hay una clase llamada `Snakes`, escribe la primera línea de la declaración de clase `Python`, expresando el hecho de que la nueva clase es en realidad una subclase de `Snake`.
Revisar
`class Python(Snakes):`

Ejercicio 2:
Algo falta en la siguiente declaración. ¿Qué es?
Revisar
`class Snakes:
 def __init__():
 self.sound = 'Sssssss'`
El constructor `__init__()` carece del parámetro obligatorio (deberíamos llamarlo `self` para cumplir con los estándares).

Below the exercises, there are numbered steps and code snippets:

- la clase.
- La parte de la clase en Python responsable de crear nuevos objetos se llama **constructor** y se implementa como un método de nombre `__init__`.
- Cada declaración de método de clase debe contener al menos un parámetro (siempre el primero) generalmente denominado `self`, y es utilizado por los objetos para identificarse a sí mismos.
- Si queremos ocultar alguno de los componentes de una clase del mundo exterior, debemos comenzar su nombre con `_`. Estos componentes se denominan **privados**.

The screenshot shows a dual-browser setup. The left browser window displays a user profile for "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES" with links for "Área personal", "Perfil", and "Calificaciones". The right browser window is titled "RESUMEN DE SECCIÓN" and contains a code editor. The code in the editor is:

```
class Snakes:
    def __init__(self):
        self.sound = 'Sssss'
```

Below the code, a message states: "El constructor `__init__()` carece del parámetro obligatorio (deberíamos llamarlo `self` para cumplir con los estándares)." A "Revisar" button is present.

Ejercicio 3

Modifica el código para garantizar que la propiedad `venomous` sea privada.

```
class Snakes:
    def __init__(self):
        self.venomous = True
```

A "Revisar" button is present. The message below the code states: "El código debería verse como sigue:" followed by the corrected code:

```
class Snakes:
    def __init__(self):
        self.__venomous = True
```

3.3.1.9 RESUMEN DE SECCIÓN.

The screenshot shows a dual-browser setup. The left browser window displays a user profile for "ALAN FRANCISCO EMMANUEL AGUILAR FUENTES" with links for "Área personal", "Perfil", and "Calificaciones". The right browser window is titled "RESUMEN DE SECCIÓN" and contains a code editor. The code in the editor is:

```
class Python:
    population = 1
    victims = 0
    def __init__(self):
        self.length_ft = 3
        self.__venomous = False
```

Below the code, a message asks: "¿Cuáles de las propiedades de la clase `Python` son variables de instancia y cuáles son variables de clase? ¿Cuáles de ellos son privados?" A "Revisar" button is present.

Ejercicio 1

population y victims son variables de clase, mientras que length y __venomous son variables de instancia (esta última también es privada).

Ejercicio 2

Vas a negar la propiedad `__venomous` del objeto `version_2`, ignorando el hecho de que la propiedad es privada. ¿Cómo vas a hacer esto?

```
version_2 = Python()
```

A "Revisar" button is present. The message below the code states: "version_2.__Python__venomous = not version_2.__Python__venomous"

3.4.1.11 RESUMEN DE SECCIÓN.

The screenshot shows a web browser window with two tabs. The left tab is from 'Plataforma Educativa Institucional' and the right tab is from 'Edube Interactive'. The right tab displays a 'RESUMEN DE SECCIÓN' (Section Summary) page. It includes a sidebar with user information (ALAN FRANCISCO EMMANUEL AGUILAR FUENTES), navigation links (Área personal, Perfil, Calificaciones), and a message about class properties. Below the sidebar, there are three exercises:

- Ejercicio 1:** A code editor with the following Python code:

```
class Sample:  
    def __init__(self):  
        self.name = Sample.__name__  
    def __repr__(self):  
        print("Mi nombre es " + self.name + " y vivo en " + Sample.__module__)
```
- Ejercicio 2:** A code editor with the following Python code:

```
class Snake:  
    def __init__(self):  
        self.victims = 0  
    def increment(self):  
        self.victims += 1
```
- Ejercicio 3:** A code editor with the following Python code:

```
class Snake:  
    pass  
class Python(Snake):  
    pass  
print(Python.__name__, 'es una', Snake.__name__)  
print(Python.__bases__[0].__name__, 'puede ser una', Python.__name__)
```

This screenshot shows the same web browser setup as the previous one, but the content has changed. The right tab now displays a 'RESUMEN DE SECCIÓN' (Section Summary) page. It includes a sidebar with user information (ALAN FRANCISCO EMMANUEL AGUILAR FUENTES), navigation links (Área personal, Perfil, Calificaciones), and a message about class properties. Below the sidebar, there are three exercises:

- Ejercicio 1:** A code editor with the following Python code:

```
class Snake:  
    def __init__(self):  
        self.victims = 0  
    def increment(self):  
        self.victims += 1
```
- Ejercicio 2:** A code editor with the following Python code:

```
class Snake:  
    def __init__(self, victims):  
        self.victims = victims
```
- Ejercicio 3:** A code editor with the following Python code:

```
class Snake:  
    pass  
class Python(Snake):  
    pass  
print(Python.__name__, 'es una', Snake.__name__)  
print(Python.__bases__[0].__name__, 'puede ser una', Python.__name__)
```

3.5.1.22 RESUMEN DE SECCIÓN.

The screenshot shows a dual-pane interface. On the left is a user profile for 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES' with links for Área personal, Perfil, and Calificaciones. On the right is a code editor titled 'SUMEN DE SECCIÓN 1/2' containing several examples of Python code related to inheritance:

- Una función llamada `issubclass(Class_1, Class_2)` es capaz de determinar si `Class_1` es una subclase de `Class_2`. Por ejemplo:

```
1 class Mouse:
2     pass
3
4 class LabMouse(Mouse):
5     pass
6
7
8 print(issubclass(Mouse, LabMouse), issubclass(LabMouse, Mouse)) # Imprime "False True"
```
- Una función llamada `isinstance(Object, Classe)` comprueba si un objeto proviene de una clase indicada. Por ejemplo:

```
1 class Mouse:
2     pass
3
4 class LabMouse(Mouse):
5     pass
6
7
8 mickey = Mouse()
9 print(isinstance(mickey, Mouse), isinstance(mickey, LabMouse)) # Imprime "True False".
```
- Un operador llamado `:a` comprueba si dos variables hacen referencia al mismo objeto. Por ejemplo:

```
1
2 Ahora responde las preguntas de los ejercicios 1-4.
```

On the right side, there is also a section for exercise answers:

- Para encontrar cualquier propiedad de objeto/clase, Python la busca dentro:
 - Del objeto mismo.
 - Todas las clases involucradas en la línea de herencia del objeto de abajo hacia arriba.
 - Si existe más de una clase en una ruta de herencia en particular, Python las examina de izquierda a derecha.
 - Si lo mencionado anteriormente falla, la excepción `AttributeError` es generada.
- Si alguna de las subclases define un método, variable de clase o variable de instancia del mismo nombre que existe en la superclase, el nuevo nombre anula cualquiera de las instancias anteriores del nombre. Por ejemplo:

```
1 class Mouse:
2     pass
3
4 class LabMouse(Mouse):
5     def __str__(self):
6         return "Hola, mi nombre es " + self.name
7
8
9 class Professor(Mouse):
10    pass
11
12 professor_mouse = LabMouse("Professor Mouse")
13 print(professor_mouse, Mouse.Population) # Imprime "Hola, mi nombre es Professor Mouse 1"
```

3.5.1.23 RESUMEN DE SECCIÓN.

The screenshot shows a dual-pane interface. On the left is a user profile for 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES' with links for Área personal, Perfil, and Calificaciones. On the right is a code editor titled 'SUMEN DE SECCIÓN 2/2' containing several exercises:

- Ejercicio 1:

¿Cuál es el resultado esperado del siguiente código?

```
print(roxy)
print(luna)
```

Roxy dice: ¡Guau! ¡No bajes, corderito!
Dobermann dice: ¡Guau! ¡Quédese donde está, intruso!
- Ejercicio 2:

¿Cuál es el resultado esperado del siguiente código?

```
print(issubclass(SleepDog, Dog), issubclass(SleepDog, GuardDog))
print(isinstance(roxy, GuardDog), isinstance(luna, GuardDog))
```

True False
False True
- Ejercicio 3:

¿Cuál es el resultado esperado del siguiente código?

```
print(luna is luna, rocky is luna)
print(roxy.kennel)
```

True False
2

Ejercicio 2
¿Cuál es el resultado esperado del siguiente código?

```
print(isinstance(SheepDog, Dog), isinstance(ShaggyDog, GuardDog))
```

 True False
False True

Ejercicio 3
¿Cuál es el resultado esperado del siguiente código?

```
print(luna is luna, rocky is luna)
print(rocky is rocky)
```

 True False
2

Ejercicio 4
Define una subclase de SheepDog llamada LowlandDog, y tráigale con un método `__str__(self)` que anule un método heredado del mismo nombre. El método `__str__(self)` debe retornar la cadena "¡Guau! ¡No me gustan las montañas!"

```
class LowlandDog(SheepDog):
    def __str__(self):
        return Dog.__str__(self) + " ¡No me gustan las montañas!"
```

3.6.1.9 RESUMEN DE SECCIÓN.

Ejercicio 1
¿Cuál es el resultado esperado del siguiente código?

```
import math

try:
    print(math.sqrt(8))
except ValueError:
    print("Error")
else:
    print("ok")
```

 3.0
ok

Ejercicio 2
¿Cuál es el resultado esperado del siguiente código?

```
import math

try:
    print(math.sqrt(-9))
except ValueError:
    print("Error")
else:
    print("ok")
finally:
    print("fin")
```

 inf
fin

Ejercicio 2
¿Cuál es el resultado esperado del siguiente código?

```
import math
try:
    print(math.sqrt(-9))
except ValueError:
    print("Error")
else:
    print("ok")
finally:
    print("fin")
```

Ejercicio 3
¿Cuál es el resultado esperado del siguiente código?

```
import math
class NewValueError(ValueError):
    def __init__(self, name, color, state):
        self.name = name
        self.color = color
        self.state = state
    try:
        raise NewValueError("Advertencia enemiga", "Alerta roja", "Alta disponibilidad")
    except NewValueError as evo:
        for arg in evo.args:
            print(arg, end=" ")
```

inf
fin

Advertencia enemiga! Alerta roja! Alta disponibilidad!

4.1.1.15 RESUMEN DE SECCIÓN.

De como saldrá: True .

4. Una lista por comprensión se convierte en un generador cuando se emplea dentro de paréntesis (usado entre paréntesis, produce una lista regular). Por ejemplo:

```
for n in (el ** 2 for el in range(5)):
    print(n)
```

De como saldrá: 02448 .

4. Una función lambda es una herramienta para crear funciones anónimas. Por ejemplo:

```
def dois(x, f):
    return f(x)
print(dois(5, lambda x: x ** 0.5))
```

De como saldrá: 5.0 .

5. La función map(func, lista) dota una copia del argumento lista, y aplica la función func a todos sus elementos, retornando un generador que proporciona el nuevo contenido de la lista elemento por elemento. Por ejemplo:

```
short_list = ['mythos', 'python', 'feil', 'on', 'the', 'floss']
```

Ejercicio 1
¿Cuál es el resultado esperado del siguiente código?

```
class Vowel:
    def __init__(self):
        self.vow = "seliny" # Si, sabemos que y no siempre se considera una vocal.
        self.pos = 0
    def __iter__(self):
        return self
    def __next__(self):
        if self.pos <= len(self.vow):
            raise StopIteration
        self.pos += 1
        return self.vow[self.pos - 1]
vowels = Vowel()
for v in vowels:
    print(v, end=" ")
```

s e i o u y

Ejercicio 2
Escribe una función lambda, estableciendo a 1 su argumento entero, y aplicalo a la función map() para producir la cadena 1 3 5 7 en la consola.

```
eng_list = [1, 2, 3, 4]
even_list = # Completa las líneas aquí.
print(even_list)
```

list(map(lambda n: n * 1, eng_list))

Ejercicio 3

¿Cuál es el resultado esperado del siguiente código?

```
def replace_spaces(replacement=" "):
    def new_replace(text):
        return text.replace(" ", replacement)
    return new_replace

spaces = replace_spaces()
print(spaces("And Now for Something Completely Different"))
```

Resumen

And Now for Something Completely Different

Nota: PEP 8, la Guía de Estilo para Código Python, recomienda que las funciones lambda no deben asignarse a variables, sino que deben definirse como funciones.

Esto significa que es mejor utilizar una sentencia `def`, y evita usar una sentencia de asignación que vincula una expresión lambda a un identificador. Analiza el código a continuación:

```
# Recomendado:
def f(x): return 3*x

# No recomendado:
f = lambda x: 3*x
```

La invocación de lambdas a identificadores generalmente duplica la funcionalidad de la declaración `def`. El uso de sentencias `def`, por otro lado, genera más líneas de código.

Es importante comprender que a la realidad a menudo le gusta doblar sus propios escenarios, que no necesariamente siguen las convenciones o recomendaciones formales. Si decides seguirlo o no, dependerá de muchas cosas: tus preferencias, otras convenciones adoptadas, las pautas internas de la empresa, la compatibilidad con el código existente, etc. Toma en cuenta esto.

4.2.1.12 RESUMEN DE SECCIÓN.

Ejercicio 1

¿Cómo se codifica el valor del argumento modo de la función `open()` si se va a crear un nuevo archivo de texto?

Resumen

"w" o "w+"; "w"

Ejercicio 2

¿Cuál es el significado del valor representado por `errno.EACCES`?

Resumen

Permito denegado: no se permite acceder al contenido del archivo.

Ejercicio 3

¿Cuál es la salida esperada del siguiente código, asumiendo que el archivo llamado file no existe?

Resumen

```
import errno

try:
    stream = open("file", "rb")
    print("archivo")
    stream.close()
except IOError as error:
    if error.errno == errno.EACCES:
        print("permiso")
    else:
        print("desconocido")
```

Resumen

desconocido

4.3.1.18 RESUMEN DE SECCIÓN.

Ejercicio 1
¿Qué se espera del método `readlines()` cuando el stream está asociado con un archivo vacío?

```
for line in open("file", "rt"):
    print(line, end="")
```

Una lista vacía (una lista de longitud cero).

Ejercicio 2
¿Qué se pretende hacer con el siguiente código?

```
for line in open("file", "rt"):
    for char in line:
        if char in "aeiouy":
            print(char, end="")
```

Copia el contenido del archivo file hacia la consola, ignorando las vocales.

Ejercicio 3
Vas a procesar un mapa de bits almacenado en un archivo llamado `Image.png` y quieres leer su contenido como un todo en una variable `bytearray` llamada `Image`. Agrega una línea al siguiente código para lograr este objetivo.

```
try:
    stream = open("Image.png", "rb")
    # Inserta una linea aqui.
    stream.read()
except IOError:
    print("Fallido")
else:
    print("Exito")
```

```
Image = bytearray(stream.read())
```

4.4.1.9 RESUMEN DE SECCIÓN.

Ejercicio 1
¿Cuál es el resultado del siguiente fragmento si se ejecuta en Unix?

```
import os
print(os.name)
```

Nota: `posix`

Ejercicio 2
¿Cuál es el resultado del siguiente fragmento de código?

```
import os
os.listdir("Hello")
```

Nota: `['Hello']`

4.5.1.23 RESUMEN DE SECCIÓN.

The screenshot shows a web browser window with multiple tabs open. The main content area displays a user profile for 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES'. On the left, there's a sidebar with links for 'Área personal', 'Perfil', and 'Calificaciones'. Below this, a code editor shows Python code related to date and time manipulation. To the right, there are two exercise sections labeled 'Ejercicio 1' and 'Ejercicio 2', each containing a code snippet and a text input field for the user to enter their answer.

4.6.1.14 RESUMEN DE SECCIÓN

This screenshot is similar to the one above, showing a user profile for 'ALAN FRANCISCO EMMANUEL AGUILAR FUENTES'. The sidebar and code editor are identical. The exercise sections on the right also show the same two exercises: 'Ejercicio 1' and 'Ejercicio 2', both involving date and time calculations.