Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación
IIC2233 — Programación Avanzada 2025-2

Examen

09 de Diciembre 2025

Instrucciones

- La evaluación consta de 31 preguntas de alternativas (30 de contenidos y 1 bonus). Para obtener el 7.0 en la evaluación, se deben tener 29 preguntas correctas.
- Recibirás una hoja de respuestas que deberás rellenar con tus datos y las respuestas correspondientes a cada alternativa o pregunta de desarrollo. Sólo se corregirá la hoja de respuestas. **Ten mucha precaución de anotar correctamente tus datos**.
- Cada pregunta de selección múltiple tiene únicamente 1 alternativa correcta. Responder con 2 o más alternativas implicará dejar inválida esa pregunta y se considerará incorrecta. Cada pregunta presenta el mismo puntaje y no se descontará por respuesta incorrecta.
- Debes completar con tus datos personales en cada hoja utilizada. Luego debes entregar de vuelta **todas** las hojas de respuesta (tanto para alternativas como de desarrollo) de la prueba, aunque estas se encuentren en blanco.
- Sólo podrás retirarte de la sala una vez hayan transcurrido 30 minutos desde que inició de la evaluación.
- Durante la evaluación se realizarán 2 rondas de preguntas. Estas preguntas únicamente serán respondidas por los profesores del ramo.
- En caso de que sea necesario, podrás solicitar hojas blancas para apoyar al desarrollo de la prueba. Para esto levanta la mano y espera que un ayudante se acerque a tu puesto.

Selección múltiple

1. ¿Qué hace "`next(mi_generador)`" en el siguiente código?

```
def un_generador():
    mi_variable = 1
    while True:
        mi_variable = yield mi_variable

mi_generador = un_generador()
next(mi_generador)
```

   A) Hace que `mi_generador` avance hasta el **`yield`**.

   B) Envía el valor **`None`** para ser almacenado en `mi_variable`.

   C) Hace que el generador termine.

   D) Hace que se caiga el programa.

   E) No afecta a `mi_generador`.

2. En el contexto de la Programación Funcional ¿cuál de las siguientes alternativas es **falsa**?

   A) La "pureza" mide el grado de dependencia de la función hacia el sistema o el programa.

   B) Una función pura retorna lo mismo para el mismo *input*, y no posee efectos secundarios.

   C) Una función pura no puede crear archivos ni modificarlos.

   D) Una función pura no puede asignar valores en su interior.

   E) Si un programa recuerda el *output* para un *input* de una función pura, se pueden reemplazar los siguientes llamados de la misma función con el mismo *input* por el valor entregado en la primera llamada.

3. Dada la siguiente función, ¿qué valor entrega `caja_secreta(13)`?

```
from functools import reduce

def caja_secreta(n):
    return reduce(lambda x, y: y - x, filter(
        lambda x: x % 3 != 0, map(
            sum, enumerate(range(n - 4, n))
    )))
```

   A) `-6`

   B) `-2`

   C) `2`

   D) `6`

   E) Ninguna de las anteriores.

4. Dado un *bytearray*, haces una función generadora que entrega de uno en uno sus *bytes*, cambiando el *endianness* original del *bytearray*.

   Completa la segunda línea de la función dado que la primera es:

   ```
   def cambio_endianness(bytearray_original):
   ```

   A) `yield reversed(bytearray_original)`

   B) `yield from reversed(bytearray_original)`

   C) `yield (256 - i for i in bytearray_original)`

   D) `yield from (256 - i for i in bytearray_original)`

   E) `return (256 - i for i in bytearray_original)`

5. Según el modelo de arquitectura basado en eventos, ¿cuál de las siguientes opciones corresponde a un **evento**?

   I. El usuario hizo *click* en un botón dentro de la interfaz.

   II. El usuario apretó una tecla en el teclado del computador.

   III. El cursor del usuario se posiciona sobre una imagen.

   A) Solo I

   B) I y II

   C) I y III

   D) II y III

   E) I, II y III

6. Dentro de las siguientes alternativas, ¿cuál es la práctica que mejor ayuda a que un programa sea **poco acoplado**?

   A) Definir funciones breves que se encarguen de realizar una sola acción.

   B) Usar nombres de variables y funciones que sean explicativos de su contenido.

   C) Usar señales para conectar las funcionalidades del programa.

   D) Separar el programa en múltiples archivos.

   E) Hacer clases personalizadas que hereden de `QWidget`.

7. Si manejas un `Lock` manualmente con `acquire()`, ¿qué debes hacer después para **liberarlo**?

   A) `lock.end()`

   B) `lock.release()`

   C) `lock.wait()`

   D) `lock.delete()`

   E) `lock.close()`

8. ¿Cuál de las siguientes alternativas es **verdadera** sobre señales en `PyQt5`?

   A) El atributo que contiene a la señal solo puede tener caracteres ASCII en su nombre.

   B) Las señales deben declararse como atributos de instancia para funcionar correctamente.

   C) Es posible que una misma señal que recibe un solo argumento envíe distintos tipos de datos a su objetivo.

   D) La señal solo puede ser emitida por instancias de la Clase donde se definió.

   E) Solo es posible obtener detalles del emisor de la señal si estos se envían como argumento al emitirla.

9. A continuación, se presentan **tres definiciones** relacionadas con ***Threading***. Seleccione la alternativa que relacione correctamente la definición con su respectivo **concepto**.

   I. Ejecución de procesos diferentes de manera intercalada.

   II. Cuando dos o más *threads* esperan a que se libere un recurso que otro *thread* tiene bloqueado, y por ende nunca se liberan.

   III. Parte del código que solo un *thread* puede ejecutar a la vez para completar una operación sin ser interrumpida.

A)	I - Paralelismo	II - Sección Critica	III - Deadlock
B)	I - Concurrencia	II - Paralelismo	III - Deadlock
C)	I - Paralelismo	II - Deadlock	III - Sección Critica
D)	I - Concurrencia	II - Sección Critica	III - Paralelismo
E)	I - Concurrencia	II - Deadlock	III - Sección Critica

10. Suponga que quieres usar archivos JSON para guardar el contenido de una cola. Indica la alternativa que define la función y clase necesarias para almacenar y cargar la información correctamente:

    A)
    ```
    def cola_hook(obj):
        return {'cola': list(obj)}

    class ColaDecoder(json.JSONDecoder):
        def default(self, obj):
            return deque(obj['cola'])
    ```

    B)
    ```
    def cola_hook(obj):
        return {'cola': obj}

    class ColaDecoder(json.JSONDecoder):
        def default(self, obj):
            return obj['cola']
    ```

    C)
    ```
    class ColaEncoder(json.JSONEncoder):
        def default(self, obj):
            return {'cola': obj}

    def cola_hook(obj):
        return obj['cola']
    ```

    D)
    ```
    class ColaEncoder(json.JSONEncoder):
        def default(self, obj):
            return {'cola': list(obj)}

    def cola_hook(obj):
        return deque(obj['cola'])
    ```

    E) Es posible cargar y guardar la información de una cola sin tener que definir clases o funciones adicionales.

11. ¿Cuál de las alternativas es **correcta**, con respecto al código mostrado a continuación?

```
from threading import Thread, Lock
import time

class MiThread(Thread):
    contador = 0

    def __init__(self, id, lock):
        super().__init__()
        self.daemon = True
        self.id = id
        self.lock = lock

    def run(self):
        with self.lock:
            time.sleep(5) # Esperar 5 segundos
            MiThread.contador += self.id
            time.sleep(5) # Esperar 5 segundos

for x in range(1, 5):
    lock = Lock()
    subthread = MiThread(x, lock)
    subthread.start()
print(MiThread.contador)
```

A) El *output* del código será 0.

B) El *output* del código será 10.

C) El *output* del código no siempre será el mismo para diferentes ejecuciones.

D) El código no tiene *output* ya que se queda estancado antes del *print*.

E) El código se demora al menos 10 segundos en terminar de ejecutarse.

12. ¿Cuál de las siguientes alternativas es **verdadera** sobre las clases abstractas?

A) Una clase abstracta solo puede heredar de una única superclase a la vez, que es la superclase `ABC`.

B) Si una subclase hereda de una clase abstracta y no sobreescribe todos sus métodos abstractos, también es considerada como una clase abstracta.

C) Si una clase abstracta define un `__init__` con atributos, estos deben ser sobreescritos en las subclases o el código generará una excepción.

D) Todos los métodos de una clase abstracta deben tener el decorador `@abstractmethod` o el código generará una excepción.

E) No es posible que una subclase herede de dos clases abstractas a la vez, solo puede heredar de una.

13. Dado el siguiente código en Python, ¿cuál es el orden de salida resultante de los llamados a `print`?

```python
import threading

def f_thread_1(lock):
    lock.acquire()
    print("1", end=" ")

def f_thread_2(evento, lock):
    evento.wait()
    lock.acquire()
    print("2", end=" ")

def f_thread_3(evento):
    evento.set()
    print("3", end=" ")

evento = threading.Event()
lock = threading.Lock()

thread_1 = threading.Thread(target=f_thread_1, args=(lock,))
thread_2 = threading.Thread(target=f_thread_2, args=(evento, lock))
thread_3 = threading.Thread(target=f_thread_3, args=(evento,))

thread_2.start()
thread_1.start()
thread_1.join()
thread_3.start()
```

A) `1 3`

B) `2 3`

C) `1 3 2`

D) `2 3 1`

E) No se imprime nada

14. ¿Cuál de los siguientes elementos **siempre** se encuentra presente tanto en una *request* como una *response*?

A) *Body*

B) *Headers*

C) Parámetros

D) Argumentos

E) *Status code*

15. Se desea implementar una **API** para administrar la información de una plataforma de películas. De acuerdo con los estándares sobre diseño de **APIs** revisados en clases, determine cuál afirmación describe correctamente los métodos `HTTP` y las rutas que deberían utilizarse a partir de los siguientes requerimientos.

    I. Obtener la información completa de una película.

    II. Modificar solo el año de estreno de una película existente.

    III. Agregar una nueva película al catálogo.

    IV. Eliminar una película específica.

    Suponga que las rutas propuestas en cada alternativa corresponden exactamente al mismo orden de los requerimientos.

A)
```
GET    /peliculas/<id>
PATCH  /peliculas/<id>
POST   /peliculas/<id>/anio
DELETE /peliculas/<id>
```

B)
```
GET    /peliculas/<id>
POST   /peliculas/<id>/anio
PATCH  /peliculas
DELETE /peliculas/<id>
```

C)
```
GET    /peliculas/<id>
POST   /peliculas/<id>/anio
GET    /peliculas/agregar-pelicula
DELETE /peliculas/<id>/eliminar
```

D)
```
GET    /peliculas/<id>
PATCH  /peliculas/<id>
POST   /peliculas
DELETE /peliculas/<id>
```

E)
```
GET    /peliculas
DELETE /peliculas/<id>/anio
GET    /peliculas/nueva
PATCH  /peliculas/anio/<id>
```

16. Suponiendo que cada método recibe los parámetros necesarios para su correcta ejecución, indica los métodos de un `socket` y el orden en que se deben ejecutar para asegurar que un servidor quede disponible y que reciba el mensaje de un cliente.

    I. `socket.accept()`

    II. `socket.bind()`

    III. `socket.listen()`

    IV. `socket.recv()`

    A) II y IV.

    B) III, II y IV.

    C) II, III y IV.

    D) II, III, I y IV.

    E) III, II, I y IV.

17. Responde la pregunta a partir del siguiente contexto:

> *Se desea programar un servidor que permita compartir archivos de música mediante sockets.*
>
> *Se espera que el servidor pueda manejar múltiples clientes de forma simultánea y que no falle ante la desconexión repentina de un cliente.*
>
> *Cuando un cliente solicita un archivo, le envía al servidor el nombre del mismo y el servidor le responde con su contenido.*

Además del contenido de *sockets*, ¿cuáles de los siguientes contenidos son **fundamentales** para programar el servidor descrito?

I. Manejo de Excepciones.

II. *Threading*.

III. Serialización.

IV. Interfaz Gráfica.

A) I, II, III y IV.

B) I, II y III.

C) II y IV.

D) I y III.

E) I y II.

18. ¿Cuál de las siguientes alternativas es **verdadera** sobre la integración de interfaces gráficas con *networking*?

A) El servidor debe mandar información a la interfaz del cliente mediante señales.

B) Al integrar *networking* a una interfaz, el servidor reemplaza al *backend* del cliente.

C) No es necesario que el servidor tenga su propia interfaz, pero es posible.

D) El servidor no puede enviar imágenes, el cliente debe contenerla.

E) Al trabajar con interfaces, el servidor debe estar en el mismo computador que el cliente.

19. A partir del siguiente contexto, indica cuál es la estructura más adecuada para modelar *Blocks World*:

> *Blocks world* indica el estado de un sistema donde se apilan bloques, el cual tiene una "garra" que puede tomar de a un bloque para luego reubicarlo.
>
> *Blocks world* se deberá modelar como una estructura inmutable, que contenga dos atributos: primero es `garra`, que posee lo que contiene la garra; y segundo es `bloques`, que posee información respecto a los bloques que no están en la garra.

A) `list`

B) `tuple`

C) `namedtuple`

D) `dict`

E) `set`

20. ¿Cuál o cuáles de las siguientes afirmaciones corresponden a **beneficios** de utilizar *sockets* sobre API para una comunicación cliente-servidor?

    I. Permite que el cliente pueda recibir información del servidor sin la necesidad de realizar una *request*.

    II. Permite que el servidor no esté obligado a responder una solicitud.

    III. Permite manejar múltiples clientes de forma simultánea.

    A) Solo II.

    B) I y II.

    C) I y III.

    D) II y III.

    E) I, II y III.

21. ¿Cuál de las siguientes funcionalidades **puede realizarse** tanto con la clase `Thread` como con la clase `QThread`?

    I. Cambiar constantemente la posición de un `QLabel` en la interfaz.

    II. Llevar una cuenta regresiva sin que se congele la interfaz.

    III. Instanciar señales de `PyQt5` como atributos de clase.

    A) Solo II.

    B) I y II.

    C) I y III.

    D) II y III.

    E) I, II y III.

22. Se tiene un programa de Python con una interfaz gráfica programada en `PyQt5`, que busca conectarse a una API programada en otro lenguaje. ¿Cuál de las siguientes alternativas es **correcta**?

    A) Para realizar la conexión, es necesario implementar un servidor como intermediario que se conecte a la interfaz mediante *sockets* y haga solicitudes a la API.

    B) Al ser un servidor tipo API y no uno con *sockets*, las *requests* pueden hacerse en el *thread* principal de la interfaz sin generar incovenientes.

    C) A pesar de ser un servidor tipo API y no uno con *sockets*, sí es posible iniciar una *request* sin que el usuario interactúe con algún elemento de la interfaz primero.

    D) Al estar la API en otro lenguaje, solo se pueden implementar señales que no envíen argumentos, o envien argumentos de tipo `str` e `int`.

    E) Al estar interactuando con una API en otro lenguaje, deberemos usar *Threads* en vez de *QThreads* ya que `PyQt5` es único a Python.

23. Debes implementar una aplicación usando `PyQt5`, en el cual se te solicita que cumpla los siguientes objetivos:

    1. **Recepción asíncrona de mensajes:** La aplicación debe recibir continuamente mensajes de un servidor, sin que la interfaz deje de procesar *inputs* por intentar recibir cada mensaje.
    2. **Actualización segura de la interfaz gráfica:** Los mensajes recibidos deben mostrarse inmediatamente en un campo de texto de la ventana principal, sin congelar la aplicación.

    Según lo que hemos visto en clases, ¿cuál **conjunto de herramientas** es el **más adecuado** para implementar los **dos objetivos anteriores**?

    A) Solo un temporizador para revisar mensajes periódicamente.

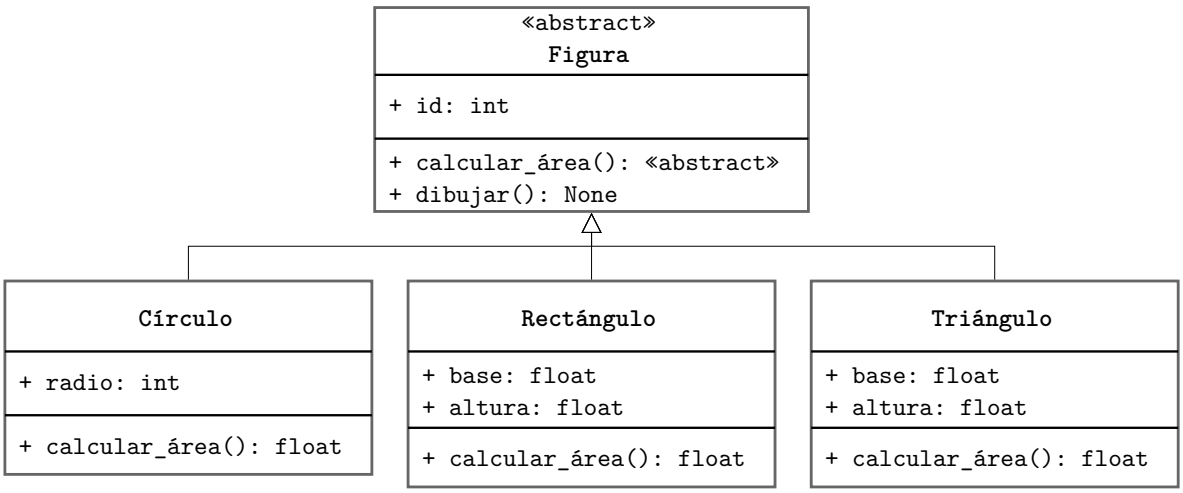
    B) Usar *sockets* para recibir mensajes y *locks* para proteger el acceso al campo de texto.

    C) Usar *threads* (QThreads) para recibir los mensajes y un manejo de señales para actualizar la interfaz.

    D) Usar *try/except* para manejar la recepción de mensajes y modificar la ventana directamente desde la excepción.

    E) Implementar un servidor dentro de la aplicación cliente para recibir mensajes y estar actualizando el campo de texto periódicamente.

24. A partir del diagrama siguiente, ¿cuál afirmación es **verdadera**?



    A) La clase `Figura` estaría incorrectamente modelada, ya que al ser una clase abstracta todos sus métodos deberían ser abstractos.

    B) La clase `Figura` hereda los atributos de sus subclases, como el atributo `radio` de `Círculo`.

    C) El método `calcular_área()` solo puede ser usado por la clase `Figura`, ya que las subclases no pueden acceder a métodos abstractos.

    D) El atributo `id` de `Figura` desaparece en las subclases, porque los atributos de la clase madre no se heredan.

    E) Las clases `Círculo`, `Rectángulo` y `Triángulo` tienen el método `dibujar()` porque lo heredan de la clase Figura.

25. ¿Cuál de las siguientes afirmaciones sobre los ***arrays* de NumPy** es **falsa**?

    A) Son rápidos para hacer cálculos matemáticos.

    B) Usan poca memoria.

    C) Necesitan bucles (ciclos) para sumar todos sus elementos.

    D) Guardan los datos de forma ordenada en memoria.

    E) Permiten aplicar operaciones a todos los elementos a la vez.

26. ¿En cuál de los siguientes **casos** sería **más conveniente** usar **expresiones regulares (regex)**?

    A) Cambiar el color de un botón en la interfaz gráfica según la acción del usuario.

    B) Ordenar alfabéticamente los nombres de los usuarios registrados.

    C) Contar el número de usuarios registrados en la base de datos.

    D) Encriptar la contraseña del usuario antes de almacenarla.

    E) Verificar si un correo electrónico ingresado tiene el formato correcto.

27. A partir del siguiente código:

```
1  def aplicar_operacion(x, y):
2      try:
3          print("I", end=" ")
4          x(y)
5      except (ValueError, TypeError, NameError):
6          print("II", end=" ")
7      else:
8          print("III", end="")
```

    ¿Qué alternativa imprime "`I II III`" en consola?

    A) `aplicar_operacion(sum, range(3))`

    B) `aplicar_operacion(sum, [1, '2'])`

    C) `aplicar_operacion(range(3), sum)`

    D) `aplicar_operacion(list, list)`

    E) Ninguna de las anteriores.

28. ¿Cuál de las siguientes acciones se puede controlar tanto con bloques de código **`try/except`** como con bloques **`if/else`**?

    A) Dividir dos números entregados por el usuario, donde el divisor podría ser 0.

    B) Leer un archivo que puede no existir en la ruta estipulada.

    C) Obtener un elemento de un diccionario con una llave que puede no existir.

    D) Multiplicar dos elementos, sin saber con certeza el tipo de dato de cada uno.

    E) Todas las anteriores son correctas.

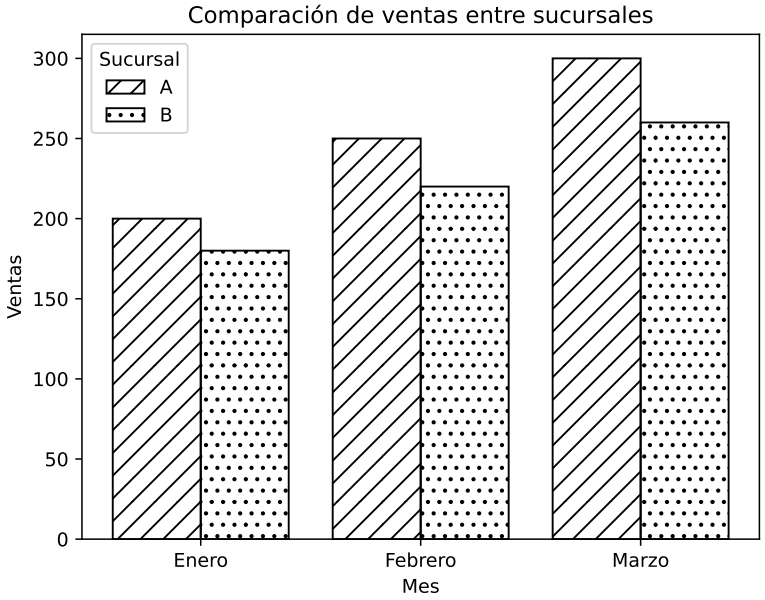
29. Se tienen los siguientes datos de ventas por mes de dos sucursales, a partir de los cuales se obtuvo la siguiente gráfica:

Datos de la sucursal A

Mes	Ventas	Sucursal
Febrero	250	A
Enero	200	A
Marzo	300	A

Datos de la sucursal B

Mes	Ventas	Sucursal
Marzo	260	B
Enero	180	B
Febrero	220	B



Para crear la gráfica de barras 'Comparación de ventas entre sucursales', es necesario realizar una serie de acciones sobre los datos. ¿Cuál o cuáles de las siguientes operaciones se utilizan para obtener la gráfica?

I. Concatenar verticalmente las tablas (`pd.concat()`), colocando las filas de la sucursal B debajo de las filas de la sucursal A.

II. Agrupar los datos por sucursal usando `groupby` para obtener la cantidad de ventas.

III. Fusionar horizontalmente las tablas (`pd.merge()`) por la columna `Mes`, sin aplicar ninguna otra transformación adicional.

A) Solo I

B) Solo II

C) Solo III

D) I y II

E) II y III

30. ¿Qué tipo de información se puede mandar en el *body* de una *request* o *response*?

A) Texto plano

B) Bytes

C) JSON

D) HTML

E) Todas las anteriores

Pregunta bonus

A continuación te encontrarás con 6 preguntas, la primera corresponde a la **pregunta 31** que no tiene puntaje asociado, pero nos permitirá identificar el tema que deseas abordar como pregunta *bonus*. Puedes escoger el tema que estimes conveniente y es obligación responder únicamente 1 alternativa. En caso de no seguir las instrucciones o no responder esta pregunta, el *bonus* será anulado.

Cada pregunta bonus está asociada a un profesor de alguna sección, pero no es necesario que respondas la pregunta asociada a tu profesor/a, puedes responder cualquiera de tu elección.

31. Selecciona la pregunta bonus que responderás:

    A) Tema 1. Profesora Daniela Concha (Sección 5)

    B) Tema 2. Profesora Tamara Vidal (Sección 4)

    C) Tema 3. Profesora Francisca Ibarra (Sección 3)

    D) Tema 4. Profesor Pablo Araneda (Sección 2)

    E) Tema 5. Profesor Cristian Ruz (Sección 1)

A continuación se presentan las 5 preguntas, una por tema, pero **sólo debes responder la pregunta asociada al tema elegido previamente**. La respuesta la debes escribir en la hoja de respuesta como la **pregunta 32**.

En cada una de estas preguntas, el profesor asociado ha descrito diferentes conceptos asociados a su contenido favorito. Sin embargo, se ha filtrado un impostor. Debes identificar correctamente cuál de los grupos de conceptos tiene una palabra o concepto incorrecto, es decir, que no está directamente relacionado con el contenido favorito del profesor/a:

Tema 1. Las Interfaces Gráficas son el contenido preferido de la profesora Daniela Concha. ¿Cuál de los siguientes grupos tiene un concepto impostor, es decir, una palabra/concepto que **no está directamente** relacionado con Interfaces Gráficas?

A) *Framework*, PyQt, Señales.

B) Eventos, *Frontend*, *Backend*.

C) Botones, Listas por compresión, *Layout*.

D) Ventanas de escritorio, POO, *Widget*.

E) *Click*, *Label*, Barra de progreso.

Tema 2. *Threading* es el contenido preferido de la profesora Tamara Vidal. ¿Cuál de los siguientes grupos tiene un concepto impostor, es decir, una palabra/concepto que **no está directamente** relacionado con *Threading*?

A) *Locks*, Eventos, `join()`

B) Operación atómica, Polimorfismo, Concurrencia

C) Paralelismo, Sección crítica, Condición de carrera

D) *Daemon*, Hilo, Proceso

E) *Global Interpreter Lock*, *multiprocess*, *Timer*

Tema 3. Estructuras de datos es el contenido preferido de la profesora Francisca Ibarra. ¿Cuál de los siguientes grupos tiene un concepto impostor, es decir, una palabra/concepto que **no está directamente** relacionado con Estructuras de datos?

A) *Stacks*, Colas, *deques*

B) Listas, Diccionarios, *namedtuples*

C) Listas ligadas, *sets*, desempaquetado

D) Árboles binarios, función pura, *kwargs*

E) Grafos, *queues*, Estructuras secuenciales

Tema 4. Codificación y serialización es el contenido preferido del profesor Pablo Araneda. ¿Cuál de los siguientes grupos tiene un concepto impostor, es decir, una palabra/concepto que **no está directamente** relacionado con Codificación y serialización?

A) Serialización, Estándar PEP8, *JSONEncoder*

B) `UTF-8`, *Data validation*, *dump*

C) `json.load`, *bytes*, *Object hook*

D) Decodificar, Conversión a estructuras de datos, *json*

E) `UTF-16`, *JSONDecoder*, Codificación

Tema 5. Excepciones es el contenido preferido del profesor Cristian Ruz. ¿Cuál de los siguientes grupos tiene un concepto impostor, es decir, una palabra/concepto que **no está directamente** relacionado con Excepciones?

A) Condición anómala, *raise*, **`SyntaxError`**

B) **`AttributeError`**, *finally*, Capturar

C) **`KeyError`**, Detención del programa, *except*

D) Excepción, **`SyntaxError`**, Manejo de excepciones

E) Flujo del programa, `f-string`, **`ZeroDivisionError`**