

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación
IIC2233 — Programación Avanzada 2025-1

Examen

1 de Julio 2025

Instrucciones

- La evaluación consta de 31 preguntas de alternativas (30 de contenidos y 1 bonus sobre Tópicos Avanzados 2). 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 60 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.
- Para quienes justificaron su ausencia en el *midterm* con su Unidad Académica y fue aceptada su justificación, deben tener las primeras 30 preguntas correctas de esta evaluación para lograr el 7.0 equivalente sólo al apartado de alternativas del *midterm* (50 % de la nota final del *midterm*).

Selección múltiple

1. En el contexto de programación funcional, ¿cuál o cuáles de los siguientes criterios se deben cumplir para que una función sea considerada pura?

   I. Su *output* depende únicamente de sus argumentos de entrada.

   II. No debe mutar valores existentes.

   III. Si se mantienen los argumentos de entrada, el resultado debe ser siempre el mismo.

   IV. No debe retornar tipos de datos mutables, como listas o diccionarios.

   A) I y II.

   B) I y III.

   C) I, II y IV.

   D) I, II y III.

   E) I, II, III y IV.

2. ¿Cuál de las siguientes opciones representa mejor el valor almacenado en la variable `var`?

```
1 from functools import reduce
2
3 var = reduce(lambda acumulado, x: f'{acumulado}, {x}', '1234', 0)
```

   A) Una instancia de `reduce` que contiene el `string` *"0, 1, 2, 3, 4"*.

   B) Una instancia de `reduce` que contiene el `string` *"3, 4"*.

   C) El `string` *"3, 4"*.

   D) El `string` *"0, 1234"*.

   E) El `string` *"0, 1, 2, 3, 4"*.

3. ¿Cuál de las siguientes afirmaciones es **correcta** respecto a los elementos gráficos básicos de una interfaz gráfica?

   A) Todos los *widgets* que no son una ventana deben tener una referencia explícita a su *widget* padre para poder ser instanciados.

   B) La única clase de la que podemos heredar en PyQt es `QWidget`, que nos permite crear las ventanas del programa.

   C) No es posible mover un *widget* una vez posicionado en la Ventana, sino que debemos eliminarlo y reemplazarlo por un nuevo *widget* en el código.

   D) Al posicionar manualmente elementos gráficos en la ventana, debemos seguir un sistema de coordenadas donde el eje X crece hacia la derecha y el eje Y crece hacia arriba.

   E) No es posible anidar *layouts* verticales dentro de *layouts* horizontales ni viceversa, sólo es posible lograr grillas usando `QGridLayout`.

4. Respecto al siguiente bloque de código, asuma que todas las importaciones ocurren y son correctas. Además, asuma que las operaciones matemáticas son correctas y que no generan errores.

```
datos_estudiantes = [{
    ["nombre"]: f"Prof_{i}",
    ["data"]:[("Clase_A", random.randint(65, 100)),
              ("Clase_B", random.randint(65, 100))]} for i in range(5)]

def calcular_promedio_estudiante(estudiante):
    """Calcula el promedio de notas para un estudiante"""
    notas = map(lambda asignatura: asignatura[1], estudiante["data"])
    promedio = sum(notas) / len(notas)
    return {"nombre": estudiante["nombre"], "promedio": promedio}

def filtrar_estudiantes_destacados(estudiantes, umbral=60):
    """Filtra estudiantes cuyo promedio supera un umbral"""
    estudiantes_promedio = map(calcular_promedio_estudiante, estudiantes)
    return filter(lambda e: e["promedio"] >= umbral, estudiantes_promedio)

def obtener_mejor_estudiante(estudiantes):
    """Obtiene el estudiante con mayor promedio"""
    def mejor_promedio(mejor, actual):
        if actual["promedio"] > mejor["promedio"]:
            return actual
        return mejor

    estudiantes_destacados = filtrar_estudiantes_destacados(estudiantes)
    return reduce(mejor_promedio, estudiantes_destacados)

print(obtener_mejor_estudiante(datos_estudiantes))
```

Considerando todo el código, independientemente de si se llega a ejecutar esa línea o no, ¿cuáles de las siguientes afirmaciones causarán algún error en el código anterior?

I. La función `calcular_promedio_estudiante` utiliza un generador para calcular el promedio.

II. La función `obtener_mejor_estudiante` usa `reduce` sin un valor inicial.

III. El diccionario `datos_estudiantes` se define incorrectamente.

IV. La función `filtrar_estudiantes_destacados` devuelve un iterador pero nunca se convierte a lista antes de usarlo en `obtener_mejor_estudiante`.

A) I y III.

B) II y IV.

C) I, II y III .

D) Sólo III.

E) Sólo I.

5. Se te solicita programar una calculadora usando PyQt, para hacer cálculos localmente en una computadora. ¿Cuál de estas alternativas **NO** debería considerarse al momento de programar la calculadora?

   A) El contenido de cada archivo.

   B) Lo que se programa en el *back-end* y en el *front-end*.

   C) Los datos a enviar entre el *front-end* y el *back-end*.

   D) Cómo lograr la cohesión deseada.

   E) Todas deberían considerarse.

6. ¿Cuál de estas **NO** es una ventaja de usar la cohesión y el acoplamientos deseados al diseñar *front-end* y *back-end*?

   A) Mejor modularidad.

   B) Mayor escalamiento.

   C) Mejor mantención.

   D) Menor tiempo de ejecución.

   E) Todas las anteriores son ventajas.

7. ¿Qué **NO** se necesita para mandar un mensaje de una máquina a otra máquina, y que este mensaje sea entendido por la otra?

   A) Un medio por donde transmitir los mensajes.

   B) Una forma de encontrar al destinatario.

   C) Una forma de encontrar al emisor.

   D) Un protocolo para transmitir mensajes.

   E) Un protocolo para entender el contenido de los mensajes.

8. En el contexto de programación funcional en Python, ¿qué son las funciones generadoras?

   A) Funciones que generan otras funciones automáticamente.

   B) Funciones que utilizan `yield` para retornar un generador.

   C) Funciones que sólo pueden ser llamadas desde clases.

   D) Funciones que generan una instancia de una clase como resultado.

   E) Funciones que utilizan `return` y retornan un generador.

9. ¿Cuántos *bytes* necesitas para almacenar la sigla del curso `IIC2233` en formato `ASCII`?

   A) 1

   B) 4

   C) 7

   D) 10

   E) 11

10. A continuación, se presenta un listado de arquitecturas de *networking* y un listado de requerimientos mínimos.

Arquitecturas	**Requerimientos mínimos**
1. Un servidor que se comunica con un cliente.	a. Múltiples *sockets*.
2. Un servidor que se comunica simultáneamente con múltiples clientes.	b. Un *socket*.
3. Un cliente que se comunica con múltiples servidores de forma no simultánea.	c. Múltiples *sockets* y *threads*.

Indica la alternativa que asocie correctamente cada arquitectura con sus requerimientos mínimos:

A) 1b, 2a, 3c.

B) 1a, 2c, 3b.

C) 1c, 2b, 3a.

D) 1c, 2a, 3b.

E) 1b, 2c, 3a.

11. ¿Cuál o cuáles de las siguientes afirmaciones corresponden a **beneficios** de los módulos `Pandas` y `Numpy`?

I. Los *DataFrame* de `Pandas` carga la información como una tabla, permitiendo hacer operaciones en columnas y filas.

II. Los *Arrays* de `Numpy` son más eficiente en memoria que las listas de Python.

III. Las estructuras de `Pandas` y `Numpy` sólo pueden contener datos numéricos.

A) Sólo I.

B) Sólo II.

C) Sólo III.

D) I y II.

E) I, II y III.

12. ¿Cuál de las siguientes afirmaciones es **correcta** respecto a los mecanismos de sincronización de *threads*?

A) Es posible evitar problemas de sincronización siempre que se instancie y empiecen los *threads* secundarios en el orden adecuado.

B) Al usar *Locks*, es necesario instanciar un *Lock* nuevo por cada *thread* secundario que se desee sincronizar.

C) Siempre que un *thread* no logre adquirir un *Lock*, quedará pausado antes de la sección crítica hasta que logre adquirirlo.

D) Es posible reiniciar el estado de un *Event* después de que este ya haya sido gatillado por un *thread*.

E) La única forma de revisar si un *Event* fue gatillado es usando el método `.wait()`, que pausará el *thread* hasta que el evento se gatille.

13. Respecto al siguiente bloque de código:

```
import threading

evento_inicio = threading.Event()
evento_fin = threading.Event()

def trabajador():
    print("Trabajador: Marcando asistencia")
    evento_inicio.wait()
    print("Trabajador: Trabajando")
    evento_fin.set()

thread = threading.Thread(target=trabajador)
thread.start()

print("Main: Dando inicio")
evento_inicio.set()
evento_fin.wait()
print("Main: Trabajo completado")
```

¿Qué sucederá al ejecutarlo?

A) Se producirá un *deadlock* porque los eventos no se reinician.

B) El orden de impresión será: `Trabajador: Marcando asistencia`, `Main: Dando inicio`, `Trabajador: Trabajando`, `Main: Trabajo completado`.

C) El orden de impresión será: `Main: Dando inicio`, `Trabajador: Marcando asistencia`, `Trabajador: Trabajando`, `Main: Trabajo completado`.

D) El *thread* principal terminará antes que el *trabajador* pueda completar su tarea.

E) Se levantará una excepción de tipo `ThreadException`.

14. Respecto a los conceptos teóricos de paralelismo y concurrencia. Asocie los siguientes ... con el concepto que corresponda para que todas sean verdaderas:

I. _____ implica la ejecución simultánea de varias tareas en diferentes núcleos o procesadores.

II. _____ consiste en gestionar varias tareas de forma intercalada en un mismo núcleo o procesador.

III. En Python, _____ se puede alcanzar con *threads* mediante el módulo `threading`.

A)	Paralelismo: I y II	Concurrencia: III
B)	Paralelismo: I	Concurrencia: II y III
C)	Paralelismo: II y III	Concurrencia: I
D)	Paralelismo: I y III	Concurrencia: II
E)	Paralelismo: III	Concurrencia: I y II

15. Asuma que las importaciones del siguiente código ocurren y son exitosas.

```
lock_datos = threading.Lock()
lock_resultados = threading.Lock()

def procesar_thread1():
    with lock_datos:
        time.sleep(5)  # Simulamos procesamiento de datos

        with lock_resultados:
            time.sleep(20) # Simulamos operación de resultado

def procesar_thread2():
    with lock_resultados:
        time.sleep(0.1)  # Simulamos procesamiento de datos

        with lock_datos:
            time.sleep(10) # Simulamos operación de resultados


# Existe código arriba de este comentario
t1 = threading.Thread(target=procesar_thread1)
t2 = threading.Thread(target=procesar_thread2)
t1.daemon = True
t2.daemon = True
t1.start()
t2.start()
t1.join()
t2.join()
# Existe código abajo de este comentario
```

¿Cuáles de las siguientes opciones puede provocar un error al ejecutar el código anterior?

A) Los *threads* están definidos como funciones y no como clases.

B) El orden en que los *threads* `procesar_thread1` y `procesar_thread2` adquieren los *locks*.

C) Los *threads* `procesar_thread1` y `procesar_thread2` son `daemon`.

D) El orden en que se ejecutan los `joins` de los *threads*.

E) Los *locks* están definidos fuera del *scope* de las funciones que usan los *threads*.

16. ¿Qué situación es un caso en el que `pickle` es no seguro para el computador que ejecuta la acción?

A) Des-serializar usando `pickle` un archivo obtenido de github.

B) Serializar usando `pickle` una foto obtenida de un pendrive encontrado en la calle.

C) Enviar *bytes* que fueron serializados con `pickle` a un servidor externo.

D) Abrir un archivo local que contenga lo almacenado por `pickle`.

E) Importar `pickle` una vez hecha una conexión con Stack Overflow.

17. Asuma que un cliente con capacidad de conectarse a un servidor tiene una interfaz gráfica, dentro de la cual usa un `QThread`. ¿Cuál es el propósito de emitir una señal `pyqtSignal` desde un `QThread` en una aplicación PyQt?

    A) Enviar datos directamente entre *sockets* del *backend* del servidor.

    B) Hacer que el *thread* principal del servidor espere la respuesta del cliente.

    C) Evitar que se ejecute el *thread* principal en el cliente.

    D) Ejecutar una función del *backend* del servidor desde la interfaz gráfica del cliente.

    E) Actualizar elementos de la interfaz gráfica del cliente desde un *thread* secundario sin que se congele la interfaz.

18. ¿Cuál o cuáles de estos tipos de datos de Python **requieren** de **personalizar** la serialización para ser serializados con `json`?

    I. `tuple` (tuplas)

    II. Clases personalizadas que hereden únicamente de **`Exception`**

    III. `deque` (cola)

    A) Sólo I.

    B) I y III.

    C) II y III.

    D) I, II y III.

    E) Ninguna es verdadera.

19. ¿Cuál o cuáles de las siguientes afirmaciones son correctas respecto a los parámetros (*params*) y el cuerpo (*body*) de una *request*?

    I. Los parámetros de una consulta forman parte del URL de una *request*.

    II. El *body* de una consulta forma parte del URL de una *request*.

    III. Las *requests* pueden tener parámetros o *body*, pero no ambos.

    A) Sólo I.

    B) I y II.

    C) I y III.

    D) II y III.

    E) I, II y III.

20. Se dispone de un servidor con una API de 3 *endpoints*, donde cada *endpoint* fue programado correctamente, pero no fue conectado a la solicitud correspondiente:

```python
from flask import Flask, request

app = Flask(__name__)
DATABASE = [{"nombre": "Los diarios de la boticaria (2a temp)", "caps": "-"},
            {"nombre": "Violet Evergarden", "caps": 13}]

@app.route("/anime/<int:index>", methods=["XXX"])
def funcion_a(index: int):
    body_data = request.get_json(force=True)
    caps = body_data["caps"]
    DATABASE[index]["caps"] = caps
    return DATABASE[index]

@app.route("/anime", methods=["XXX"])
def funcion_b():
    body_data = request.get_json(force=True)
    caps = body_data["caps"]
    nombre = body_data["nombre"]
    DATABASE.append({"nombre": nombre, "caps": caps})
    return DATABASE[-1]

@app.route("/anime/buscar/<str:nombre>", methods=["XXX"])
def funcion_c(nombre: str):
    return list(filter(lambda d: d["nombre"] == nombre, DATABASE))

if __name__ == "__main__":
    app.run(host="localhost", port=2233)
```

Indique la alternativa que, al reemplazar los `"XXX"`, conecta **correctamente** y respetando el protocolo HTTP, cada uno de los *endpoint* con la solicitud correspondiente.

A) `funcion_a`: `PATCH` `funcion_b`: `POST` `funcion_c`: `GET`

B) `funcion_a`: `GET` `funcion_b`: `POST` `funcion_c`: `PATCH`

C) `funcion_a`: `PATCH` `funcion_b`: `GET` `funcion_c`: `POST`

D) `funcion_a`: `POST` `funcion_b`: `PATCH` `funcion_c`: `GET`

E) `funcion_a`: `POST` `funcion_b`: `GET` `funcion_c`: `PATCH`

21. ¿Cuál de los siguientes tipos de datos **no** presenta un iterador?

A) *Strings*

B) Conjuntos (*Sets*)

C) Tuplas

D) Diccionarios

E) Todos los tipos de datos anteriores presentan un iterador.

22. ¿Qué error tiene el siguiente código al intentar conectar una señal?

```
class Ventana(QWidget):
    senal_enviar = pyqtSignal(str)
    def __init__(self):
        super().__init__()
        self.boton = QPushButton("Enviar")
        self.boton.clicked.connect(self.emitir_senal)

    def emitir_senal(self):
        self.senal_enviar("Hola")
```

A) Falta definir la señal antes del `__init__()`.

B) La señal debe recibir como argumento `int` en lugar de `str`.

C) Los botones en PyQt no pueden recibir como argumento funciones, sólo señales `pyqtSignal`.

D) La señal no está enviando correctamente el mensaje.

E) La señal debe ser definida dentro del método `__init__()` para poder usarse.

23. ¿Qué imprimirá el siguiente código?

```
class CajaSecreta:
    def __init__(self, valor):
        self.valor_secreto = valor
        self.multiplicador_secreto = 2

    @property
    def multiplicador(self):
        return self.multiplicador_secreto * 3

    @property
    def valor(self):
        return self.valor_secreto * self.multiplicador

    @valor.setter
    def valor(self, otro):
        self.multiplicador_secreto = otro

caja = CajaSecreta(1)
caja.valor_secreto *= 2
caja.valor += caja.multiplicador
print(caja.valor)
```

A) 12

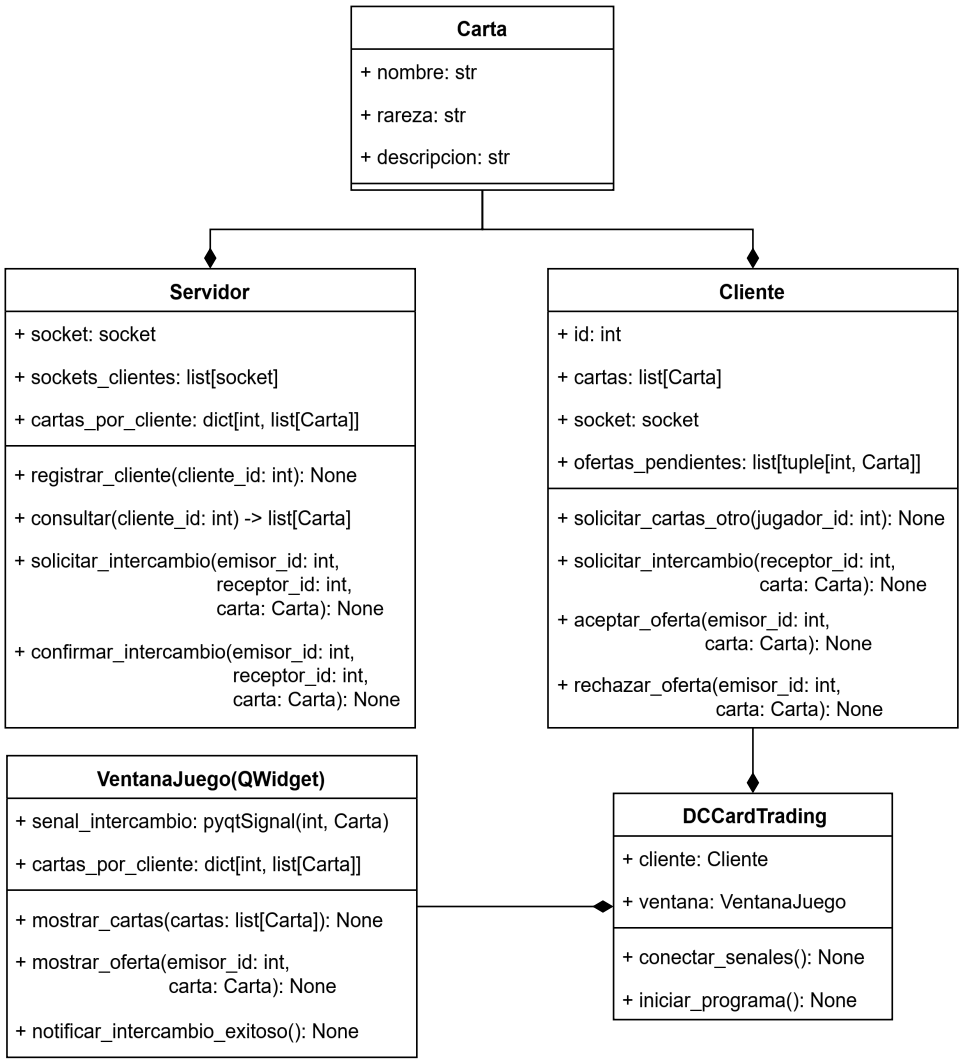
B) 24

C) 54

D) 108

E) 216

Utiliza el siguiente diagrama de clases para responder las preguntas 24 y 25. El diagrama muestra la modelación mínima de una aplicación cliente-servidor llamada DCCardTrading. El objetivo de esta aplicación es intercambiar cartas entre jugadores, como un sistema de colección digital.



24. Consideremos el caso en el que un jugador quiere revisar las cartas que tiene otro jugador antes de ofrecerle una carta. ¿Cuál de las siguientes secuencias de interacción entre clases describe **correctamente** la interacción esperada entre los componentes del sistema?

    A) `VentanaJuego` usa su método `mostrar_cartas` para consultar las cartas del otro jugador directamente al `Servidor`.

    B) El `Cliente` emite una señal que es capturada por `VentanaJuego`, quien se comunica con el `Servidor` para obtener la información del otro jugador.

    C) El `Servidor` envía automáticamente las cartas de todos los jugadores a un `Cliente` al momento de conectarse, y las guarda en su atributo `cartas`.

    D) Un `Cliente` se comunica directamente con el atributo `cartas_por_cliente` del `Servidor` para obtener las cartas del otro `Cliente`.

    E) El usuario interactúa con `VentanaJuego`, lo que activa una señal que es recibida por el `Cliente`, quien envía una solicitud usando `socket`. El `Servidor` procesa la solicitud y responde.

25. Respecto al diagrama de clases indicado, considere el siguiente flujo:

    1. El usuario presiona un botón en la interfaz para seleccionar una carta y solicitar intercambiarla.
    2. Se ejecuta la lógica que comunique esa intención al servidor.

    ¿Cuál de las siguientes líneas de código, que implican conexiones de señales y métodos, debe usarse y además es más coherente con la arquitectura?

    A) `Cliente.senal_intercambio.connect(VentanaJuego.solicitar_intercambio)`

    B) `VentanaJuego.senal_intercambio.connect(Cliente.solicitar_intercambio)`

    C) `VentanaJuego.senal_intercambio.connect(Servidor.confirmar_intercambio)`

    D) `VentanaJuego.senal_intercambio.connect(Carta.mostrar)`

    E) `Cliente.senal_intercambio.connect(Servidor.solicitar_intercambio)`

26. Según lo visto en el curso sobre algoritmos de búsqueda en grafos, ¿qué estructura es la **más adecuada** para mantener el registro de nodos que debe visitar en cada algoritmo?

    A) BFS: *stack* DFS: *set*

    B) BFS: *set* DFS: cola

    C) BFS: cola DFS: *set*

    D) BFS: *stack* DFS: cola

    E) BFS: cola DFS: *stack*

27. En expresiones regulares usamos: `.` para representar cualquier carácter, `()` para agrupar, `+` para indicar que el carácter o grupo está 1 o más veces y `{n}` para indicar que el carácter o grupo está n veces.

    Dado lo anterior, ¿qué alternativa presenta una expresión regular que hace *fullmatch* con el texto `'llama'`?

    A) `'(.{2}a)+'`

    B) `'(.+a){2}'`

    C) `'(.a){2}'`

    D) `'(.{2}a){2}'`

    E) `'(.a)+'`

28. ¿Cuál de las siguientes afirmaciones es **verdadera** sobre iteradores e iterables?

    A) En algunos casos, es posible recorrer un mismo iterador múltiples veces.

    B) Un iterable es un tipo de iterador, pero un iterador no es un tipo de iterable.

    C) El estado que almacena el siguiente valor a retornar de un iterador es independiente del iterable.

    D) Un iterador debe siempre retornar todos los elementos del iterable al menos una vez.

    E) En Python, no es posible iterar sobre ningún tipo clases personalizadas, ya que lanzarán una excepción.

29. Analiza el siguiente código, ¿qué ocurre al ejecutarlo?

```
class A:
    def imprimir(self):
        print('Buenos días')

class B:
    def imprimir(self):
        print('Buenas tardes')

class C(A, B):
    def imprimir(self):
        pass

c = C()
c.imprimir()
```

A) Arrojará un error de conflicto de nombres de método.

B) El código del método de la clase `A` toma precedencia.

C) El código del método de la clase `B` toma precedencia.

D) Ambos códigos de los métodos `imprimir` de A y B se ejecutan en orden, primero el de A y luego el de B.

E) No se imprime ningún texto.

30. Se presenta la siguiente secuencia de comandos de GIT sobre un repositorio local:

```
git status
git add gato.txt
git commit -m "gato añadido"
git commit -m "perro añadido"
git add perro.txt
git commit -m "ahora si que si perro añadido perdón"
git add carpeta/capibara.py
git commit -m "carpeta añadida"
git status
git commit -m "todo lo que falta añadido"
git push
```

Asumiendo que todos los archivos y carpetas mencionadas poseen cambios respecto al repositorio remoto, ¿cuántos *commits* nuevos se habrán creado en el repositorio tras correr todos los comandos?

A) 1

B) 2

C) 3

D) 4

E) 5

Preguntas bonus - Tópicos Avanzados 2

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.

31 ¿Cuál es el tema que responderás como pregunta 32?

A) Tema 1: Archivos de Audio e Imagen.

B) Tema 2: Simulación.

C) Tema 3: Aplicaciones Web a lo largo de la historia.

D) Tema 4: Seguridad computacional y *Scraping*.

E) Tema 5: *Machine Learning* y *Deep Learning* en la actualidad.

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**:

Tema 1. Dado lo visto en la experiencia con archivos de audio (`.wav`), ¿cuál de estos casos requiere de modificar únicamente el *header* del archivo?

A) Hacer que el archivo de audio se reproduzca al revés.

B) Agregarle eco al audio.

C) Cambiar el nombre del archivo.

D) Invertir los canales de audio, es decir, en qué oreja sale el sonido.

E) Hacer que el archivo de audio se reproduzca mas rápido.

Tema 2. Te encuentras modelando un problema, donde te dan el siguiente contexto:

> Se desea modelar un juego de mundo abierto donde 1 día en el juego dura 1 hora en la vida real. Durante un día, existen tanto eventos que depende de la hora (inicio del amanecer, día, anochecer y noche) como evento que no depende de la hora (inicio y termino de distintos tipos de clima).

¿Qué alternativa indica el paradigma de programación **más adecuado** para implementar lo pedido, junto a la razón que **fundamenta** esta decisión?

A) Simulación Síncrona, debido a que 1 día de juego dura 1 hora real.

B) Simulación Síncrona, debido a que hay eventos que dependen de la hora.

C) Simulación por Eventos Discretos, debido a que hay eventos que depende de la hora.

D) Simulación por Eventos Discretos, debido a que hay eventos que NO depende de la hora.

E) Ambos paradigmas son equivalentes, ya que el objetivo es simular el flujo del tiempo.

Tema 3. ¿Qué evento ocurrió en la fecha que es actualmente denominada como "el cumpleaños del internet"?

A) La creación de la red ARPANET en Estados Unidos.

B) La adopción de TCP/IP como protocolo por defecto.

C) La publicación de la primera página web.

D) La creación del primer navegador web.

E) La introducción de JS y CSS a las páginas web.

Tema 4. ¿Cuál de las siguientes opciones NO se considera como un tipo de *Malware*?

A) Gusanos

B) Ransomware

C) Cryware

D) Spyware

E) Adware

Tema 5. Una de las siguientes parejas de concepto y definición **no es correcta**. Identifica cuál és.

A) Características (*features*): etapa donde se limpian y transforman los datos antes de entrenar el modelo.

B) Red convolucional: tipo de red neuronal profunda especializada en procesamiento de imágenes mediante filtros.

C) Clasificación: proceso de asignar una etiqueta o categoría a los datos.

D) Modelo no supervisado: encuentra patrones en datos sin etiquetas.

E) Set de *test*: subconjunto de datos usado para evaluar el rendimiento del modelo.