Nombre y Apellido: Legajo:

Examen Recuperatorio

1er Parcial

Tiempo mínimo para el examen: 1 hora reloj.
Tiempo máximo para el examen: 2 horas reloj.

El examen se compone los siguientes ejercicios. Se solicita documentar bien su código (por ej., cuando deba definir funciones/métodos indique claramente los tipos de los parámetros, el tipo del valor de retorno y explique brevemente qué hace la función/método).

1. Implemente una función recursiva llamada `invertir_numero_rec` que recibe como parámetro un entero mayor o igual a cero y devuelve otro entero compuesto por los mismos digitos pero en orden inverso. También implemente una función iterativa `invertir_numero_it` que realice la misma tarea. No puede usar ninguna función ya existente para realizar la tarea de inversión, debe implementar Ud. el paso a paso. Sí puede ayudarse definiendo una función auxiliar, y que la recursividad/iteración solicitada esté en esa función auxiliar. Preste atención a los ejemplos con el tema de los ceros en los extremos.

   Ejemplo de Uso:

   - `invertir_numero_rec(12345)` debería devolver `54321`.
   - `invertir_numero_rec(30500)` debería devolver `503`.
   - `invertir_numero_it(120)` debería devolver `21`.
   - `invertir_numero_it(76349)` debería devolver `94367`.

2. Considere la siguiente función:

```
def f(l: list) -> int:
    sol = 0
    n = len(l)
    for i in range(n):
        for j in range(n - 1, -1, -1):
                sol += ( l[i] * l[j] )
    return sol
```

   y responda las siguientes preguntas:

   1. ¿Cuántas operaciones de multiplicación se realizan en total en la función f?
   2. ¿Cuál es el orden de complejidad temporal de la función?

3. **Sistema de Gestión de Reservas de Habitaciones de un Hotel**

   Implemente un sistema de gestión de reservas de hotel utilizando programación orientada a objetos. Para ello, implemente las clases que se especifican a continuación:

   Implementación de una clase `Habitacion`:

   Implemente una clase `Habitacion` que tenga los siguientes atributos:

   - `numero_habitacion (str)`: el número de identificación único de la habitación.
   - `tipo (str)`: el tipo de habitación (simple, doble, suite).
   - `precio_noche (float)`: el precio por noche.

La clase debe incluir un método `__init__` (con los argumentos que considere necesarios) y `__str__` que devuelva una representación en cadena de la habitación.

Implementación de la clase `Cliente`:

Implemente una clase `Cliente` que tenga los siguientes atributos:

- `dni_cliente (str)`: el DNI del cliente.
- `mail_cliente (str)`: el mail del cliente.
- `nombre_cliente (str)`: el nombre del cliente.

La clase debe incluir un método `__init__` (con los argumentos que considere necesarios) y `__str__` que devuelva una representación en cadena del cliente.

Implementación de la clase `ReservaHabitacion`:

Implemente una clase `ReservaHabitacion` que tenga los siguientes atributos:

- `nro_reserva (str)`: el número de identificación de la reserva.
- `dni_cliente (str)`: el dni del cliente asociado a la reserva.
- `numero_habitacion (str)`: el número de habitación asociada a la reserva.
- `fecha_inicio (str)`: fecha de inicio de la reserva en formatto yy-mm-dd
- `cant_noches (int)`: cantidad de noches asociada a la reserva.
- `monto_total (float)`: monto total abonado por la reserva.

La clase debe incluir un método `__init__` y `__str__` que devuelva una representación en cadena de la información de la reserva. No se preocupe por el formato de la fecha, asumimos que los controles se realizan previamente y que la fecha tiene el formato especificado (no debe chequear el formato).

Implementación de la clase SistemaHotel:

Implemente una clase `SistemaHotel` que gestione múltiples habitaciones y clientes y que tenga los siguientes atributos:

- `habitaciones (dict[str, Habitacion])`: un diccionario con todas las habitaciones gestionadas en el sistema. La clave será el número de la habitación.
- `reservas (dict[str, dict[str, ReservaHabitacion]])`: un diccionario cuya clave es el número de habitación y cuyo valor es otro diccionario con clave número de reserva y valor asociado la reserva con todos sus datos.

Esta clase debe tener los siguientes métodos:

- `__init__()`: Crea un sistema de hotel vacío (sin habitaciones ni reservas).
- `agregar_habitacion`: Recibe como parámetro un objeto de tipo Habitacion y lo agrega al diccionario habitaciones.
- `eliminar_habitacion`: Recibe como parámetro el número de una habitación y elimina la habitación correspondiente del sistema.
- `mostrar_habitaciones`: Muestra por pantalla todas las habitaciones disponibles en el sistema.
- `buscar_reservas_cliente`: Muestra por pantalla todas las reservas de un cliente. Elija el/los parámetro(s) que

considere adecuados para este método.

- `realizar_reserva`: Recibe como parámetros el DNI del cliente, el número de habitación, fecha de inicio (asumimos en formato válido) y cantidad de noches. Permite realizar una reserva si la habitación está disponible. Para ello, asumimos que la clase `SistemaHotel` tiene un método `chequear_fechas` que recibe el número de habitación, la fecha de inicio de una posible reserva y la cantidad de noches deseadas y devuelve un booleano, indicando si la habitación está (`True`) o no (`False`) disponible en ese período. No debe implementar el método. `chequear_fechas`, lo asumimos ya existente. El monto de la reserva se calcula multiplicando el precio de la habitación por la cantidad de noches. En número de reserva se genera concatenando el dni del cliente, el número de la habitacion y la fecha de inicio de la reserva, separados por puntos.

- `cancelar_reserva`: Permite cancelar una reserva hecha por un cliente. Elija los parámetros que considere adecuados para este método.

```
# Ejemplo de uso
sistema = SistemaHotel()

# Creando habitaciones
habitacion1 = Habitacion("101", "Doble", 100.0)
habitacion2 = Habitacion("102", "Suite", 200.0)
sistema.agregar_habitacion(habitacion1)
sistema.agregar_habitacion(habitacion2)

# Mostrando habitaciones
sistema.mostrar_habitaciones()

# Realizando reservas
sistema.realizar_reserva("12345678", "101", "2023-10-20", 3)
sistema.realizar_reserva("87654321", "102", "2023-10-21", 2)

# Buscando reservas
sistema.buscar_reservas_cliente("12345678")

# Cancelando una reserva
sistema.cancelar_reserva("12345678.101.2023-10-20")  # Cancela la primera reserva
```