



Tiempo de ejecución: 4 horas  
Planteamiento de la sesión

## Ejercitaciones

- Escribir una clase en python llamada **círculo** que contenga un radio, con un método que devuelva el área y otro que devuelva el perímetro del círculo.
- Escribir una clase en python llamada **rectángulo** que contenga una base y una altura, y que contenga un método que devuelva el área del rectángulo.
- Escribir una clase en python con 2 métodos: **get\_string** y **print\_string**. **get\_string** acepta una cadena ingresada por el usuario y **print\_string** imprime la cadena en mayúsculas.
- Escribir una clase en python que convierta un número entero a número romano
- Escribir una clase en python que convierta un número romano en un número entero

## Sistema de Gestión de Estudiantes y Notas

**Nota:** Utiliza principios de encapsulamiento y herencia según sea necesario para organizar y estructurar tu código de manera eficiente. La asignación es permitir a los usuarios agregar, eliminar, modificar y consultar información sobre estudiantes, asignaturas y sus respectivas notas.

### Funcionalidades a Implementar:

- **Gestión de Estudiantes:**
  - Permite agregar nuevos estudiantes al sistema con su nombre, apellido y edad.
  - Permite eliminar estudiantes del sistema.
  - Permite modificar la información de estudiantes existentes,



Incluyendo su nombre, apellido y edad.

- **Gestión de Asignaturas:**

- Permite crear nuevas asignaturas en el sistema con su nombre.
- Permite eliminar asignaturas del sistema.
- Permite modificar el nombre de las asignaturas existentes.

- **Gestión de Notas:**

- Permite agregar notas para un estudiante en una asignatura específica.
- Permite consultar las notas de un estudiante en todas sus asignaturas.
- Permite consultar el promedio de notas de un estudiante en todas sus asignaturas.
- Permite consultar el promedio de notas de una asignatura específica.
- Permite consultar las notas individuales de todos los estudiantes en una asignatura específica.

Utiliza clases para representar estudiantes, asignaturas y notas, con atributos y métodos adecuados para cada una.

Documenta tu código de manera adecuada, incluyendo comentarios descriptivos para explicar la funcionalidad de cada clase y método.

## Solución

```
class Estudiante:
```

```
    def __init__(self, nombre, apellido, edad):
```

```
        self.nombre = nombre
```

```
        self.apellido = apellido
```

```
        self.edad = edad
```

```
        self.notas = {}
```

```
    def agregar_nota(self, asignatura, nota):
```

```
        self.notas[asignatura] = nota
```

```
    def promedio_general(self):
```

```
        if not self.notas:
```

```
            return 0
```

```
        return sum(self.notas.values()) / len(self.notas)
```

```
    def __str__(self):
```

```
        return f'{self.nombre} {self.apellido} (Edad: {self.edad})'
```



```
class Asignatura:
    def __init__(self, nombre):
        self.nombre = nombre
        self.estudiantes = {}
        def agregar_estudiante(self, estudiante, nota):
            self.estudiantes[estudiante] = nota

    def promedio_asignatura(self):
        if not self.estudiantes:
            return 0
        return sum(self.estudiantes.values()) / len(self.estudiantes)

    def __str__(self):
        return self.nombre

class SistemaGestionEstudiantes:
    def __init__(self):
        self.estudiantes = {}
        self.asignaturas = {}

    def agregar_estudiante(self, estudiante):
        self.estudiantes[estudiante] = True

    def eliminar_estudiante(self, estudiante):
        if estudiante in self.estudiantes:
            del self.estudiantes[estudiante]

    def agregar_asignatura(self, asignatura):
        self.asignaturas[asignatura] = True

    def eliminar_asignatura(self, asignatura):
        if asignatura in self.asignaturas:
            del self.asignaturas[asignatura]

    def asignar_nota(self, estudiante, asignatura, nota):
        if estudiante in self.estudiantes and asignatura in self.asignaturas:
            asignatura.agregar_estudiante(estudiante, nota)
            estudiante.agregar_nota(asignatura, nota)
```



```
def mostrar_notas_estudiante(self, estudiante):
    if estudiante in self.estudiantes:
        print(f"Notas de {estudiante}:")
        for asignatura, nota in estudiante.notas.items():
            print(f"{asignatura}: {nota}")

def promedio_general_estudiante(self, estudiante):
    if estudiante in self.estudiantes:
        print(f"Promediode {estudiante}: {estudiante.promedio_general()}")

def promedio_asignatura(self, asignatura):
    if asignatura in self.asignaturas:
        print(f"Promediode{asignatura}: {asignatura.promedio_asignatura()}")
```

Ejemplo de uso:

```
sistema = SistemaGestionEstudiantes()

# Crear estudiantes
estudiante1 = Estudiante("Juan","Pérez", 20)
estudiante2 = Estudiante("María","Gómez", 22)

# Crear asignaturas
asignatura1 = Asignatura("Matemáticas")
asignatura2 = Asignatura("Ciencias")

# Agregar estudiantes y asignaturas al sistema
sistema.agregar_estudiante(estudiante1)
sistema.agregar_estudiante(estudiante2)
sistema.agregar_asignatura(asignatura1)
sistema.agregar_asignatura(asignatura2)
```



### # Asignar notas

```
sistema.asignar_nota(estudiante1, asignatura1, 90)  
sistema.asignar_nota(estudiante1, asignatura2, 85)  
sistema.asignar_nota(estudiante2, asignatura1, 88)  
sistema.asignar_nota(estudiante2, asignatura2, 92)
```

### # Mostrar notas y promedios

```
sistema.mostrar_notas_estudiante(estudiante1)  
sistema.mostrar_notas_estudiante(estudiante2)  
sistema.promedio_general_estudiante(estudiante1)  
sistema.promedio_general_estudiante(estudiante2)  
sistema.promedio_asignatura(asignatura1)  
sistema.promedio_asignatura(asignatura2)
```