PEP 3 Paradigmas de Programación (Junio 2022)

NOMBRE: **PROFESOR:** **RUT:**

Instrucciones: En **AMBAS PREGUNTAS** **(1)** hacer un uso adecuado del paradigma orientado a objetos,**(2)** use pseudo-java **(3)** documentar; **(4)** responder preguntas en hojas separadas indicando en cada una (preguntas y respuestas) su nombre, rut y profesor; **(5)** entregar todas las hojas (preguntas y respuestas) al final; **(6)** responda solo en base a lo solicitado.

Pregunta 1 (30 pts):

Considere la especificación de un nuevo sistema de ofimática cloud, similar a GoogleDocs, denominado CloudDocs. Este sistema está destinado a atender a usuarios alrededor del mundo. Al registrarse en la plataforma, cada usuario queda identificado con un nombre, fecha de nacimiento, correo electrónico y una contraseña, siendo estos dos últimos las credenciales para iniciar sesión. Solo estudiantes mayores de 18 años pueden registrarse en la plataforma.

Respecto de la plataforma, esta es capaz de manejar carpetas y subcarpetas, las que tienen un nombre, fecha de creación y colaboradores. Dentro de las carpetas se pueden alojar documentos, planillas de cálculo y presentaciones. Cada uno de estos tipos de archivos están identificados con un nombre único, tienen fecha de creación, colaboradores. Cada tipo de archivo se abre con software especializados. Específicamente, los documentos se abren con CloudDocument, las planillas con CloudSpreadSheet y las presentaciones con CloudSlides. Cada sistema de software ofrece servicios especializados según el tipo de archivo (ej: planillas permiten realizar operaciones aritméticas; los documentos permiten añadir hojas, las planillas permiten agregar slides, presentar).

El nuevo sistema se lanzará con estos tres tipos de documentos y software especializados, sin embargo se proyecta que en los próximos años se incorporen nuevos formatos (videos, fotos, planos, mapas, etc.), por lo que el diseño debe considerar la posibilidad de extender el sistema a futuro.

CloudDocs será empleado directamente por personas naturales y también por organizaciones. Las organizaciones pueden albergar una cantidad limitada de usuarios, los que quedan definidos por una cuota.

A partir de estos antecedentes, abordar los siguientes ítems:

a) **(20 pts)** Elabore un diagrama de clases a nivel de diseño preliminar que permita ofrecer una solución al escenario descrito anteriormente.
b) **(10 pts)** Señale directamente en el diagrama (como comentarios) todos los conceptos de POO que estarían presentes en su propuesta.

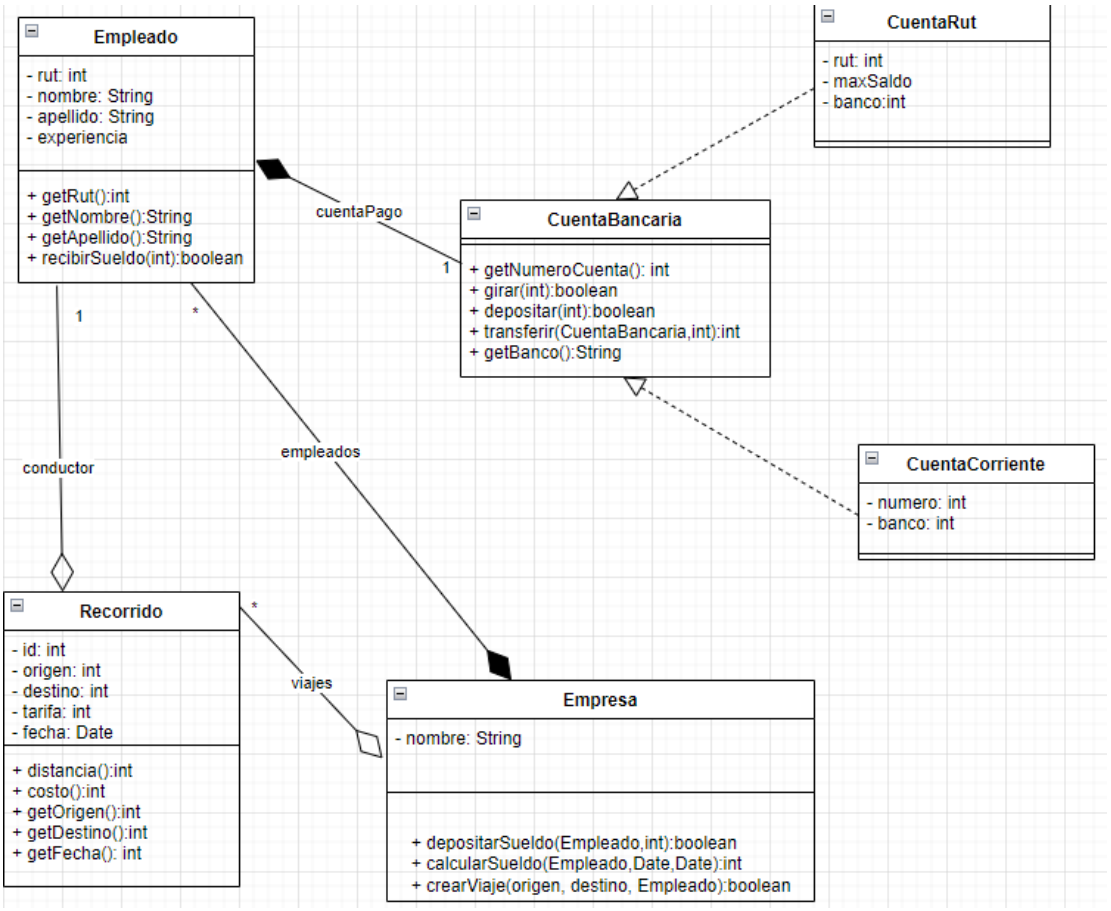
PEP 3 Paradigmas de Programación (Junio 2022)

NOMBRE: **PROFESOR:** **RUT:**

Instrucciones: En **AMBAS PREGUNTAS** **(1)** hacer un uso adecuado del paradigma orientado a objetos,**(2)** use pseudo-java **(3)** documentar; **(4)** responder preguntas en hojas separadas indicando en cada una (preguntas y respuestas) su nombre, rut y profesor; **(5)** entregar todas las hojas (preguntas y respuestas) al final; **(6)** responda solo en base a lo solicitado.

Pregunta 2 (30 pts):

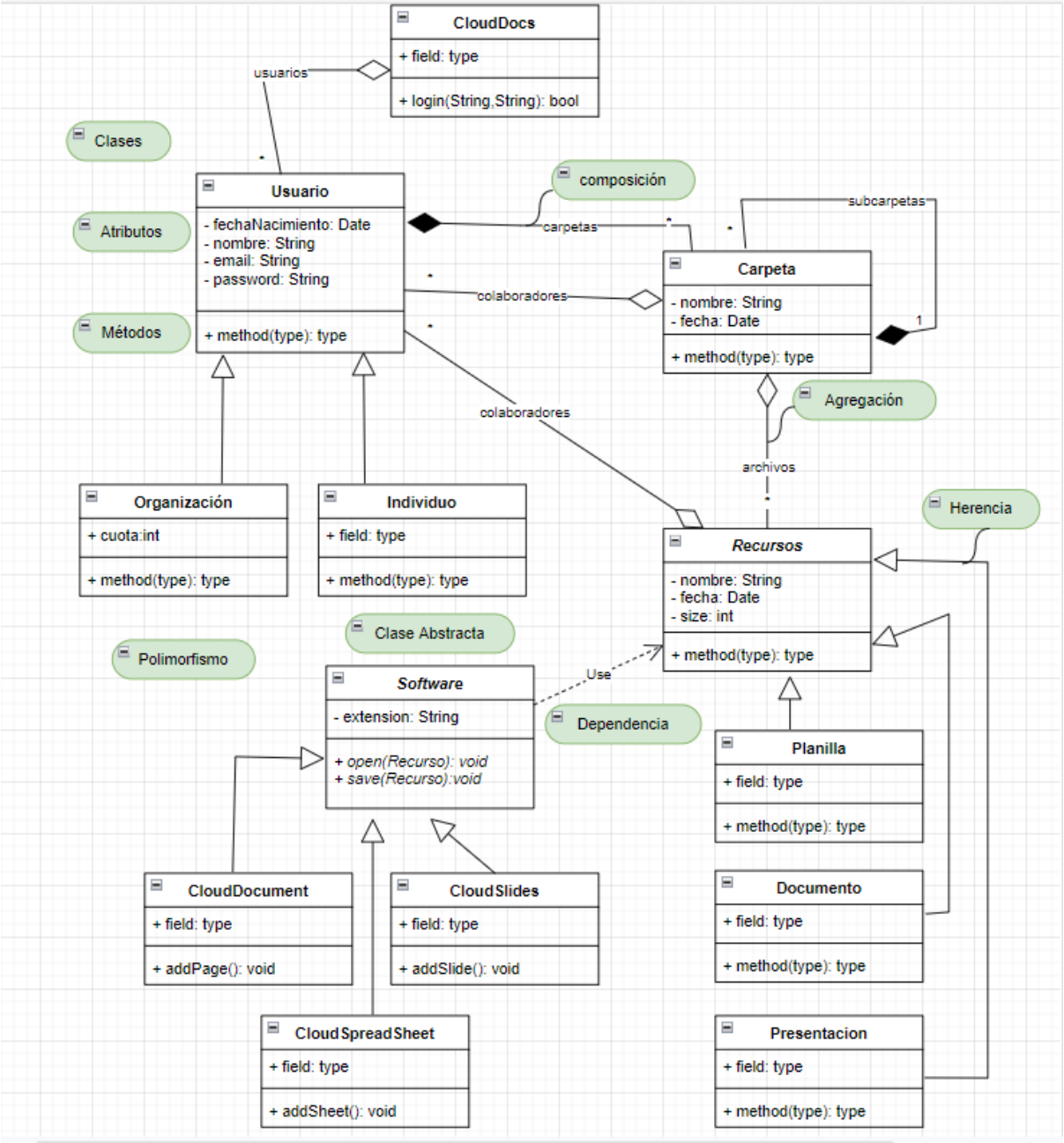
Se planea construir un sistema para un servicio de transporte urbano (como Uber o Lyft) A partir del siguiente diagrama de clases:



a) **(5 pts)** En el diagrama añada comentarios para señalar los tipos de relaciones que se expresan en el modelo.
b) **(8 pts)** Relate a través de una narrativa breve (máximo 8 líneas) la situación, problema o solución que se expresa a través del diagrama de clases.
c) **(12 pts)** Realizar una implementación en pseudo-java a partir del diagrama de clases. Procure destacar con comentarios donde queden expresadas las relaciones. Solo implemente la estructura de las clases, atributos y constructores si corresponde. No implementar el cuerpo de los métodos.
d) **(5 pts)** Implemente el método calcularSueldo, que permite calcular el sueldo de un empleado para un periodo comprendido entre dos fechas (Date). Asuma que los métodos de las otras clases ya están implementados.

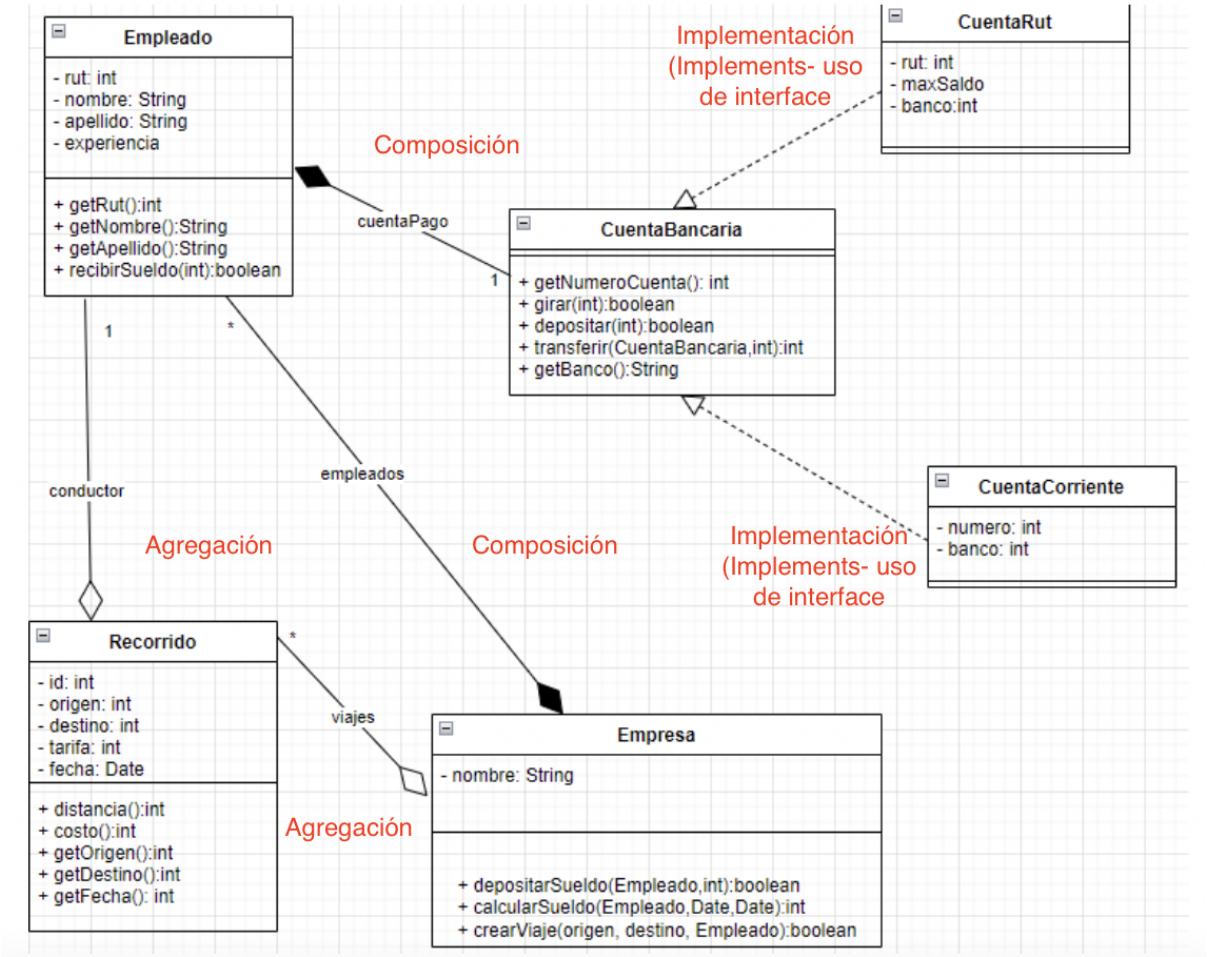
PAUTA PEP 3 Paradigmas de Programación (Junio 2022)

PREGUNTA 1



PREGUNTA 2

a) **(5 pts)** En el diagrama añada comentarios para señalar los tipos de relaciones que se expresan en el modelo.



- Recordar que:
  - Rombo vacío: agregación
  - Rombo pintado: composición
  - Línea punteada flecha triangular: implementación, uso de interfaces
  - Linea solida flecha triangular: herencia (extends)

Ponderación	**Explicación**
0	No responde o respuesta totalmente incorrecta
0,25	Solo define unas pocas relaciones correcta
0,5	Define la mitad de las asociaciones de forma correcta
0,75	Solo contiene 1 o 2 errores referentes a la asociaciones
1	Respuesta correcta sin fallos ni observaciones

b) **(8 pts)** Relate a través de una narrativa breve (máximo 8 líneas) la situación, problema o solución que se expresa a través del diagrama de clases.

En esta pregunta debe narrar de forma sucinta lo que usted observa del diagrama, puede describir la situación, el problema o la solución que se le presentó.

En esencia debe describir que existe una empresa la cual tiene empleados y realiza recorridos. Cada empleado tiene una cuenta bancaria que puede ser Cuenta Rut o Cuenta Corriente. Los empleados de esta empresa realizan viajes que consisten en un origen, destino y una fecha. Luego, la empresa calcula el sueldo a través del costo del viaje y asimismo deposita el sueldo a la cuenta bancaria del empleado.

Ponderación	**Explicación**
0	No responde o respuesta totalmente incorrecta
0,25	En su narración no logra identificar el contexto ni las clases, agrega otro contexto
0,5	En su narración identifica las clases y el contexto, pero se desvia de lo pedido, agrega clases que no son necesarias
0,75	En su narración identifica clases, relaciones y el contexto, solo tiene pequeñas observaciones
1	Respuesta correcta sin fallos ni observaciones

c) **(12 pts)** Realizar una implementación en pseudo-java a partir del diagrama de clases. Procure destacar con comentarios donde queden expresadas las relaciones. Solo implemente la estructura de las clases, atributos y constructores si corresponde. No implementar el cuerpo de los métodos.

- Las clases son públicas y se expresan como: Public Class NombreClase
- Los modificadores de acceso son: - (private), + (public), # (protected)

```
Public Class Empleado {

   private int rut;

   private String nombre;

   private String apellido;
```

```
    private int experiencia;
    private CuentaBancaria cuentaPago = new CuentaRut();  //solo para ilustrar composición

    public int getRut() { }
    public String getNombre() { }
    public String getApellido() { }
    public boolean recibirSueldo(int sueldo) { }
}

Public Class Recorrido {
    private int id;
    private int origen;
    private int destino;
    private int tarifa;
    private Date fecha;
    private Empleado conductor;

    public int distancia() { }
    public int costo() { }
    public int getOrigen() { }
    public int getDestino() { }
    public int getFecha() { }
}

Public Class Empresa {
    private String nombre;
    private List<Empleado> empleados = new ArrayList<Empleado>();
```

```
    private List<Recorrido> viajes= new ArrayList<Recorrido>();
    public Empresa()
 {
//Solo para ilustrar composición
       this.empleados.add(new Empleado());
       this.empleados.add(new Empleado());
   }
    public boolean depositarSueldo(Empleado empleado, int sueldo) { }
    public int calcularSueldo(Empleado empleado, Date fechaInicio, Date fechaFinal) { }
    public boolean crearViaje(Recorrido origen, Recorrido destino, Empleado empleado) { }
}

Public Interface CuentaBancaria {
    public int getNumeroCuenta();
    public boolean girar(int cantidad);
    public boolean depositar(int cantidad);
    public int transferir(CuentaBancaria cuentaATransferir, int cantidad);
    public String getBanco();
}

Public Class CuentaRut implements CuentaBancaria {
    private int rut;
    private int maxSaldo;
    private int banco;
    //Implementación de métodos de interface CuentaBancaria }

Public Class CuentaCorriente implements CuentaBancaria {
```

```
    private int numero;

    private int banco;

    //Implementación de métodos de interface CuentaBancaria

}
```

Ponderación	Explicación
0	No responde o respuesta totalmente incorrecta
0,25	Identifica uso de clases, clase abstractas, interface, herencia, etc. Define Empleado con su constructor, getters and setters, uso pseudo java
0,5	Define todas las clases del diagrama, uso correcto de constructor, getters and setters y métodos de clase, no obstante no implementa uso de relaciones
0,75	Implementa todo lo anterior junto con el uso correcto de las relaciones, pero solo tiene observaciones menores
1	Respuesta correcta sin fallos ni observaciones

d) **(5 pts)** Implemente el método calcularSueldo, que permite calcular el sueldo de un empleado para un periodo comprendido entre dos fechas (Date). Asuma que los métodos de las otras clases ya están implementados.

La clase Empresa tiene los siguientes atributos:

```
    private String nombre;

    private List<Empleado> empleados;

    private List<Recorrido> viajes;
```

Del cual

```
Public int calcularSueldo(Empleado empleado, Date fechaInicio, Date fechaTermino) {

    // Asuma que el tipo Date se puede comparar con <, >, =

    int sueldo = 0;

    for (Recorrido viaje : viajes) {

        if (viaje.getFecha() >= fechaInicio && viaje.getFecha <= fechaTermino) {
```

```
            sueldo += recorrido.getCosto();

        }

    }

    return sueldo;

}
```