UNIVERSIDAD POPULAR DEL CESAR

GUÍA DE ACTIVIDADES PARA LA CONSTRUCCIÓN DE CONOCIMIENTO

NOMBRE DE LA ASIGNATURA	Programación de Computadores II							
NOMBRE DE LA ACTIVIDAD	Parcial 2							
TIPO DE ACTIVIDAD	Sincrónica		Asincrónica	x	Individual	x	Grupal	

TEMÁTICA REQUERIDA PARA LA ACTIVIDAD	**OBJETIVOS**
Unidad 3. Herencia, polimorfismo e interfaces	Evaluar la capacidad de utilizar los mecanismos de herencia, polimorfismo, clases abstractas e interfaces, colecciones de objetos en la construcción de programas.

COMPETENCIAS	**INSUMOS PARA EL DESARROLLO DE LA ACTIVIDAD / REFERENCIAS BIBLIOGRÁFICAS**
• Identificación de clases, clases abstractas, atributos, métodos concretos y métodos abstractos. • Representación de herencia entre clases con UML • Implementación de herencia entre clases • Implementación de sobreescritura de métodos • Implementación de polimorfismo • Creación e implementación de interfaces • Uso de colecciones del tipo Arraylist	• Material educativo y material complementario de la asignatura "**Unidad 3.**" • Talleres y código elaborado en el desarrollo de la asignatura.

CONOCIMIENTOS PREVIOS REQUERIDOS

Conceptos fundamentales de POO - Estructura básica de clases – Relaciones entre clases y UML

ESPECIFICACIONES DE LA ACTIVIDAD

Requerimientos de la evaluación:

- Diseño de diagrama de clases
- Implementación en código Java

Problema planteado:

Se requiere implementar un aplicativo para un restaurante el cual mantiene varias sedes en el País. El aplicativo debe permitir la gestión de los platos y pedidos.

El restaurante maneja dos tipos de menú: vegano y a base de carnes. Todos los platos veganos tienen una tarifa básica (10000), más un incremento según la variedad de frutas, vegetales y legumbres que contenga asi: para menos de 4 variedades tiene un costo de 5000, entre 4 y 7 variedades un incremento de 8000, más de 7 variedades un adicional de 12000.

Por su parte, los platos a la carta, por ser platos especiales, su costo se calcula teniendo en cuenta su proteina principal, del cual se debe conocer su nombre, cantidad en gramos, y el valor por gramos.

Al momento de registrar cualquier plato se requiere el código del plato y su nombre.

Por políticas del restaurante, los platos veganos solo se despachan para consumir en el local, mientras que los platos a base de carnes, adicionalmente pueden ser despachados a domicilio. El servicio de domicilio tiene una tarifa del 5%, mientras que el servicio de los platos que se sirven en la mesa de local, se cobra una propina equivalente al 10% del pedido en el caso de las carnes y 5% en el caso de los platos veganos.

Se requiere un diagrama de clases adecuado para la solución, con las clases, interfaces y relaciones que usted considere necesarias.

Se debe incluir una clase Restaurante, en la cual se debe implementar los siguientes métodos:

- **agregarPlato:** El cual debe permitir adicionar un nuevo plato a la lista de platos del restaurante, usted debe identificar que parámetro o valores de retorno requiere el método.
- **buscarPlato:** buscara y retornara un plato de la lista de platos del restaurante a partir de su código
- **servicioDomicilio:** para calcular y retornar el valor total a cancelar por el pedido de un plato del restaurante mediante domicilio. El método debe validar que el plato permita servicio a domicilio.
- **servicioDeMesa:** para calcular y retornar el valor total a cancelar por el pedido de un plato del restaurante en el local. El método debe validar que el plato permita despacho en el local.

El uso de los mecanismos de herencia, polimorfismo, interfaces, clases abstractas y colecciones influirá directamente en la evaluación.

Un ejemplo de la clase principal seria:

```
        Restaurante restaurante= new Restaurante();

        restaurante.agregarPlato(new Carne(123, "Baby beaf", "Lomito fino", 250, 100));
        restaurante.agregarPlato(new Vegano(234, "Frutas silvestre",6));

        Plato servicio1 = restaurante.buscarPlato(123);
        Plato servicio2 = restaurante.buscarPlato(234);
        Plato servicio3 = restaurante.buscarPlato(123);
        Plato servicio4 = restaurante.buscarPlato(234);

        System.out.println("Costo servicio 1: "+ restaurante.servicioDeMesa(servicio1));
        System.out.println("Costo servicio 2: "+ restaurante.servicioDomicilio(servicio2));
        System.out.println("Costo servicio 3: "+ restaurante.servicioDomicilio(servicio3));
        System.out.println("Costo servicio 4: "+ restaurante.servicioDeMesa(servicio4));
    }
```

Salida

```
run:
Costo servicio 1: 27500.0
Costo servicio 2: 0.0
Costo servicio 3: 26250.0
Costo servicio 4: 18900.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

- **Modo de entrega**

Montar en aula web archivo con extensión .rar con nombre del estudiante, que incluya el código y diagrama elaborado.

RECOMENDACIONES / OBSERVACIONES	Sin observaciones