Nombre y Apellido: ..... N° Legajo: .....

Segundo Parcial de Programación Orientada a Objetos (72.33)

08/11/2018

Ejercicio 1	*Ejercicio 2*	*Ejercicio 3*	*Nota*	*Firma Docente*
/3	/3	/4		

- **Condición mínima de aprobación: SUMAR 5 PUNTOS.**
- **Las soluciones que no se ajusten al paradigma OO, no serán aceptadas.**
- **Las soluciones que no se ajusten estrictamente al enunciado, no serán aceptadas.**
- **Puede entregarse en lápiz.**
- **No es necesario escribir las sentencias `import`.**
- **Además de las clases solicitadas se pueden agregar las que consideren necesarias.**
- **Escribir en cada hoja Nombre, Apellido, Legajo, Número de Hoja y Total Hojas entregadas.**

Ejercicio 1

La clase `MapIterator` modela un **iterador de mapas**. Permite iterar sobre una instancia de `Map` (que recibe en su constructor), accediendo a cada una de las claves del mismo con el método `next()` y al valor asociado a esa clave en el mapa mediante el método `getValue()`.

Implementar todo lo necesario para que, con el siguiente programa

```
import java.util.HashMap;
import java.util.Map;

public class MapIteratorTester {

    public static void main(String[] args) {
        Map<Integer, String> map = new HashMap<>();
        map.put(1, "One");
        map.put(2, "Two");
        map.put(3, "Three");
        MapIterator<Integer, String> mapIterator = new MapIterator<>(map);
        while (mapIterator.hasNext()) {
            System.out.println("Key: " + mapIterator.next());
            System.out.println("Value: " + mapIterator.getValue());
        }
        mapIterator = new MapIterator<>(map);
        try {
            mapIterator.getValue();
        } catch (Exception ex) {
            System.out.println(ex.getClass());
        }
        System.out.println(mapIterator.next());
        System.out.println(mapIterator.getValue());
        System.out.println(mapIterator.getValue());
        while (mapIterator.hasNext()) {
            mapIterator.next();
        }
        mapIterator.next();
    }

}
```

se obtenga la siguiente salida:

```
Key: 1
Value: One
Key: 2
Value: Two
Key: 3
Value: Three
```

```
class java.util.NoSuchElementException
1
One
One
Exception in thread "main" java.util.NoSuchElementException
...
```

Ejercicio 2

Se cuenta con la interfaz `DoubleKeyMap` que modela un mapa con **claves compuestas**.

La clave compuesta, que es única, está dada por dos valores, no necesariamente del mismo tipo. Cada clave compuesta tiene sólo un valor asociado.

```
public interface DoubleKeyMap<K1,K2,V> {

   int size();

   boolean isEmpty();

   boolean containsKey(K1 firstKey, K2 secondKey);

   boolean containsValue(V value);

   V get(K1 firstKey, K2 secondKey);

   void put(K1 firstKey, K2 secondKey, V value);

}
```

<u>Implementar todo lo necesario</u> para que, con el siguiente programa se obtenga la salida indicada en los comentarios:

```
public class DoubleKeyMapTester {

    public static void main(String[] args) {
        DoubleKeyMap<String, String, Integer> doubleKeyMap = new DoubleKeyHashMap<>();
        doubleKeyMap.put("Juan", "Perez", 49);
        System.out.println(doubleKeyMap.size()); // 1
        doubleKeyMap.put("Lucas", "Gomez", 37);
        doubleKeyMap.put("Lucas", "Lopez", 26);
        doubleKeyMap.put("Juan", "Lopez", 55);
        System.out.println(doubleKeyMap.size()); // 4
        System.out.println(doubleKeyMap.isEmpty()); // false
        System.out.println(doubleKeyMap.containsKey("Juan", "Ramirez")); // false
        System.out.println(doubleKeyMap.containsKey("Juan", "Gomez")); // false
        System.out.println(doubleKeyMap.containsKey("Lucas", "Gomez")); // true
        System.out.println(doubleKeyMap.get("Lucas", "Gomez")); // 37
        System.out.println(doubleKeyMap.get("Lucas", "Lopez")); // 26
        System.out.println(doubleKeyMap.containsValue(26)); // true
        doubleKeyMap.put("Lucas", "Lopez", 27);
        System.out.println(doubleKeyMap.size()); // 4
        System.out.println(doubleKeyMap.containsValue(26)); // false
        System.out.println(doubleKeyMap.get("Lucas", "Lopez")); // 27
        System.out.println(doubleKeyMap.containsKey("Gomez", "Lucas")); // false
        System.out.println(doubleKeyMap.containsValue(10)); // false
    }

}
```

Ejercicio 3

Se debe implementar un conjunto de clases para utilizar una "**canasta de compras**". Para ello se deben modelar:

- los **artículos**, donde cada artículo cuenta con una descripción y, dependiendo del tipo de artículo, se cuenta además con:
  - la cantidad de unidades (un número entero mayor a cero) si el artículo se vende por unidad.
  - la cantidad (un número real mayor a cero) si el artículo se vende por peso.
  - nada, si es un servicio.
- la **canasta de compras**, donde cada canasta cuenta con el nombre del cliente y una serie de artículos.

Implementar todo lo necesario y completar los ..... para que, con el siguiente programa de prueba:

```
public class BasketTester {

   public static void main(String args[]) {

       // Creamos un artículo de venta por unidad: un libro que cuesta $ 235.00

       ..................... book = new .....................("El Aleph - Borges", 235.0);

       // Creamos un artículo de venta por peso: los porotos cuestan $80 el kilo

       ..................... bean = new .....................("Porotos Blancos", 80.0);

       // Creamos un servicio: el envío express cuesta $145.50

       ..................... express = new .....................("Envío Express", 145.50);

       // Juana Saldivar compra:
       // dos copias del libro de Borges
       // 750 gramos de porotos
       // y quiere el envío express
       Basket basket = new Basket("Juana Saldivar")
                       .add(book, 2).add(bean, 0.75).add(express);

       // Aumentaron los porotos
       bean.setPrice(90.0);

       // Listamos el detalle de la compra
       System.out.print(basket);

       System.out.println("----------");

       // Consultamos el total de la venta
       System.out.print(basket.getCost());
   }

}
```

se obtenga la siguiente salida:

```
Juana Saldivar
El Aleph - Borges: 2 * $235.00 = $470.00
Porotos Blancos: 0.75 * $90.00 = $67.50
Envío Express: $145.50
----------
683.0
```

siendo `683.0 = 2 * 235 + 0.75 * 90 + 145.50`