Programación II
Conjunto paramétrico<T>

Introducción

Extenderemos la clase Conjunto para que sea paramétrica.

Sintaxis

También agregaré la sintaxis para exigir que los elementos del Conjunto sean comparables.
De esta forma, los métodos como "máximo()" podrán funcionar sin problemas.

Conjunto definido sobre el tipo T

Definición de la clase y operaciones básicas:

```
public class Conjunto<T extends Comparable<T>>{
private ArrayList<T> conjunto;

public Conjunto(){
      conjunto = new ArrayList<T>();
}

public void agregar(T elem){
      if (!conjunto.contains(elem)){
            conjunto.add(elem);
      }
}

public T iesimo(int i){
      return conjunto.get(i);
}

public int tamano(){
      return conjunto.size();
}

public void eliminar(int i){
      T elem = conjunto.remove(i);

}
```

A continuación agregaremos operaciones basadas en las operaciones básicas.
En particular implementaremos "máximo()", donde se hace obligatorio que los elementos de Conjunto sean comparables entre si.

```
public void union(Conjunto c){ //conjunto UNION c
	for (int i=0;i<c.tamano();i++){
		agregar((T)c.iesimo(i));
	}
}


public boolean pertenece(T elem){
	return conjunto.contains(elem);
}

public void interseccion(Conjunto c){
	for (int i=0;i<tamano();i++){
		if (!c.pertenece(iesimo(i))){
			eliminar(i);
		}
	}
}

public T maximo(){
	T max = null;

	if (tamano()>0){
		max = conjunto.get(0);
	}

	for (int i=0;i<tamano();i++){

		if (max.compareTo(conjunto.get(i)) < 0){
                                                //max < conjunto.get(i)
			max=conjunto.get(i);
		}
	}
	return max;
}
}
```

CompareTo() devuelve tres valores posibles:

a.compareTo(b)	Valor
a > b	1
a = b	0
a < b	-1

Por eso si "`max.compareTo(conjunto.get(i)) < 0`", significa que:
`Max < conjunto.get(i)`