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

Segundo Parcial de Estructuras de Datos y Algoritmos

26 de Noviembre de 2009

Ejercicio 1	*Ejercicio 2*	*Ejercicio 3*	*Nota*

Condición Mínima de Aprobación: Tener por lo menos dos ejercicios con B-

Consideraciones a tener en cuenta. MUY IMPORTANTE

- **El ejercicio que no respete estrictamente el enunciado será anulado.**
- **Se puede entregar el examen escrito en lápiz**
- **Se tendrán en cuenta la eficiencia y el estilo de programación.**
- **Los teléfonos celulares deben estar apagados.**

Ejercicio 1

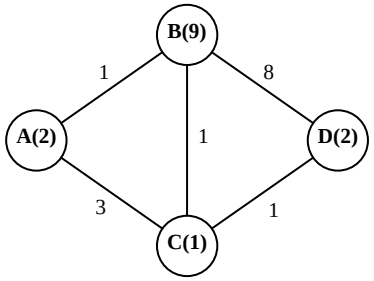
Se cuenta con la siguiente implementación parcial de un grafo que almacena pesos enteros positivos en las aristas y en los nodos:

```
public class Graph<V> {

	private HashMap<V, Node> nodes = new HashMap<V, Node>();
	private List<Node> nodeList = new ArrayList<Node>();

	public void addVertex(V vertex, int weight) {
		if (!nodes.containsKey(vertex)) {
			Node node = new Node(vertex, weight);
			nodes.put(vertex, node);
			nodeList.add(node);
		}
	}

	public void addArc(V v, V w, int weight) {
		Node origin = nodes.get(v);
		Node dest = nodes.get(w);
		if (origin != null && dest != null && !origin.equals(dest)) {
			for (Arc arc : origin.adj) {
				if (arc.neighbor.info.equals(w)) {
					return;
				}
			}
			origin.adj.add(new Arc(weight, dest));
			dest.adj.add(new Arc(weight, origin));
		}
	}

	private class Node {
		V info;
		int weight;
		boolean visited = false;
		int tag = 0;
		List<Arc> adj = new ArrayList<Arc>();

		public Node(V info, int weight) {
			this.info = info;
			this.weight = weight;
		}
		public int hashCode() {
			return info.hashCode();
		}
```

```
		public boolean equals(Object obj) {
			if (obj == null || !(obj.getClass() != getClass())) {
				return false;
			}
			return info.equals(((Node)obj).info);
		}
	}
	private class Arc {
		int weight;
		Node neighbor;

		public Arc(int weight, Node neighbor) {
			this.weight = weight;
			this.neighbor = neighbor;
		}
	}
}
```

Agregarle a esta implementación el método **minWeight**, que dado un nodo origen y un nodo destino retorna el peso del camino mínimo entre ambos, en donde el peso de un camino se calcula como la sumatoria de los pesos de las aristas y de los nodos que lo conforman.

En el siguiente grafo, el camino mínimo entre los nodos A y D está formado por los nodos {A, C, D} y tiene peso 9.



Ejercicio 2

Agregarle a la implementación de grafos del ejercicio anterior el método **countCycles**, que dado un nodo cuenta la cantidad de ciclos simples a los que éste pertenece.

En el grafo de ejemplo del ejercicio anterior, los nodos A y D pertenecen a 2 ciclos simples, mientras que los nodos B y C pertenecen a 3.

Ejercicio 3

Se denomina cuadrado mágico de orden N a la disposición de N^2 números enteros diferentes en una matriz de N x N, tal que la suma de cada fila, de cada columna y de las diagonales es constante. A continuación se muestra un ejemplo de una posible solución para N=3. En este caso todas las filas, columnas y diagonales suman 15:

2	7	6
9	5	1
4	3	8

Implementar un programa que dado un valor de N imprima en la salida **todos** los cuadrados mágicos que se pueden formar con los números entre 1 y N^2.