Explicación Práctica

- Heap

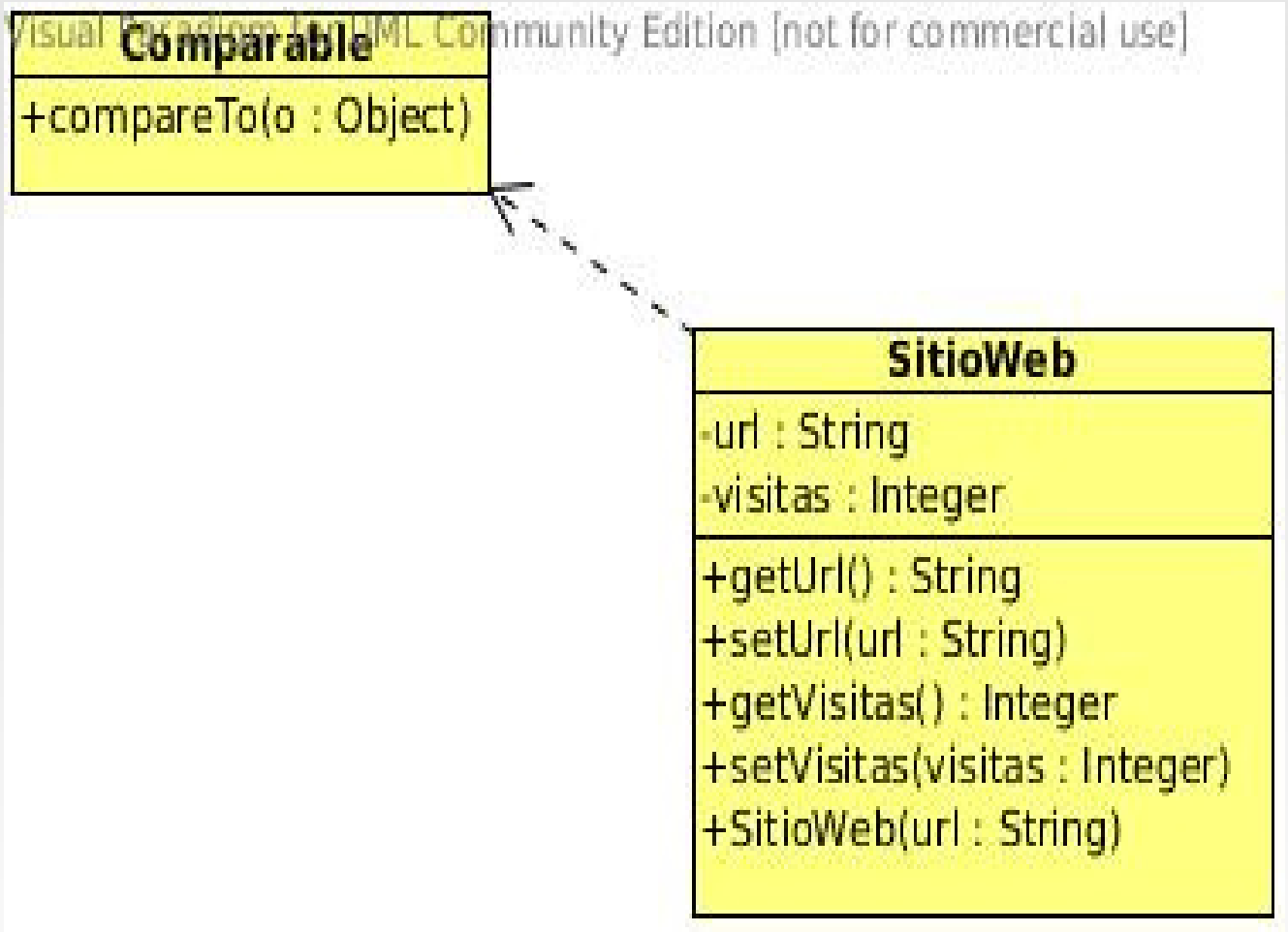
Aplicación

- Dada una secuencia de valores que representan la cantidad de visitas a distintos sitios web, se necesita buscar el sitio web rankeado en la posición k. La secuencia de elementos está almacenada en una lista, y para encontrar el k-ésimo elemento sólo se pueden almacenar en memoria k elementos simultáneamente. Resolver el ejercicio de forma tal que la cantidad de comparaciones entre elementos sea mínima.

Descripción

- Del total de elementos disponibles en la lista, nos interesa manejar de manera eficiente el acceso a los k-máximos.
- **K < N**

Clase SitioWeb



Estructura a utilizar

- ¿Qué estructura se debería utilizar para almacenar y administrar de manera eficiente los *k* sitios más visitados ? Y poder recuperar el sitio que se encuentra en la posición k con un algoritmo de O (1) ?

Propuesta de Solución

- Se propone manejar el ranking a través de una MinHeap de SitiosWeb con capacidad para *k* sitios, y devolver el tope de la misma que sería el sitio k-ésimo más visitado.

Implementación

```
public class AdministradorDeSitiosWeb{

        private Heap<SitioWeb> ranking;


        public <SitioWeb> obtenerRanking(ListaGenerica<SitioWeb> sitios, int k){

                this.crearHeapDeKElementos(k, sitios);
                this.agregarRestantes(sitios);
                return this.ranking.minimo();
        }

        private void crearHeapDeKElementos(int k, ListaGenerica<SitioWeb>   sitios){
                ...
        }

        private void agregarRestantes(ListaGenerica<SitioWeb> sitios){
                ...
        }
}
```

Implementación

```
private void crearHeapDeKElementos(int k, ListaGenerica<SitioWeb> sitios)  {

        SitioWeb[] arrayKSitios = new SitioWeb[k];
         sitios.comenzar();
        for (int i = 0; i < k ; i++){
                arrayKSitios[i] = sitios.elemento();
                sitios.proximo();
        }
        this.ranking = new Heap(arrayKSitios);

}
```

Implementación

```
private void agregarRestantes(ListaGenerica<SitioWeb> sitios){

        while(!sitios.fin()){
                SitioWeb sitio = sitios.elemento();
                SitioWeb min = this.ranking.minimo();
                if(sitio.compareTo(min) > 0){
                        this.ranking.eliminar();
                        this.ranking.agregar (sitio);
                }
                sitios.proximo();
        }
}
```