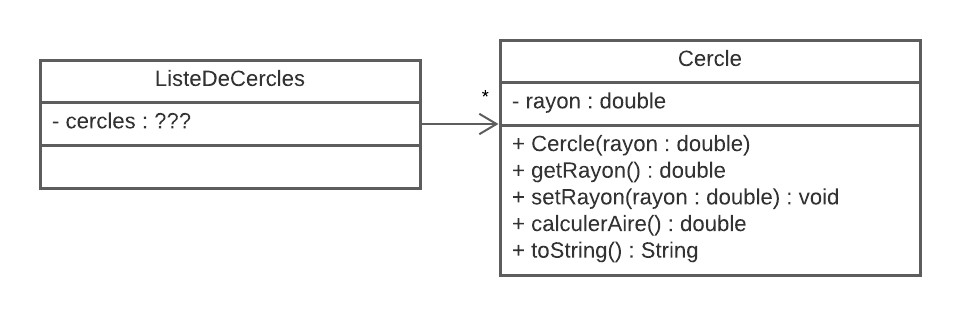
**Association multiple, Arraylist, égalité des objets**

**Association multiple**

Une association multiple est lorsque on utilise un attribut qui permet de garder plusieurs de ce type (**Arraylist< >).**

**En uml elle est caractérisé par une fléche partant de la classe liste a la classe objet :**

…Plusieurs cercles…



\*

**Foreach : permet de faire une boucle dans les éléments de la liste (tableau)**

for (Cercle c : cercles){

texte += "\n" + c.toString();

}

return texte;

}

**Représentation en mémoire : si on enleve un élement de la liste, on doit faire bouger tout les éléments vers l’avant, si on supprime un éléments qui n’est pas dans la liste car déjà un éléments de sa même valeur l’est alors cela va supprimé l’éléments avec la même valeur dans la liste**

liste

:

ListeDeCercles

cercles

=

:

ArrayList

>

Cercle

<

size = 3

**…**

c1

:

Cercle

rayon = 6.0

c2

:

Cercle

rayon = 3.0

c3

:

Cercle

rayon = 6.0

**Pour comparé 2 variables qui pourrait avoir la même valeur sans pour autant avoir la même place en mémoire on utilise equals**

**Pour comparé les string on utilise toujours le equals**

Il faut définir 2 méthodes :

Equals

hashCode

public class Cercle{l

private double rayon;

**public boolean equals(Object o)** {

//s’ils ont la même adresse mémoire, ils sont

//égaux.

if (this == o) return true;

//si l’objet en paramètre est null, ils ne sont pas égaux.

if (o == null) return false;

//Si les objets ne sont pas de la même classe, ils ne sont pas égaux.

if (this.getClass() != o.getClass()) return false; //On “transforme” le paramètre en un objet de type Cercle et on compare les rayons.

Cercle cercle = (Cercle) o;

return this.rayon == cercle.rayon; }

**Si on utilise la methode equals on doit utilisé hashCode**

public int hashCode() {

return Objects.hash(rayon);

}