HLIN406 - Exercices sur :
les assertions et les exceptions en Java
Les Piles

Une interface `IPile` représentant le type abstrait *Pile*, une classe `Pile` l'implémentant avec une `ArrayList` pour stocker les éléments et une classe exception `PileVideException` sont présentées ci-dessous.

```
public class PileVideException extends Exception {
     public PileVideException() { }
     public PileVideException(String message) { super(message); }
}

public interface IPile<T>
{
     void initialiser();
     void empiler(T t) throws PileVideException;
     T depiler()throws PileVideException;;
     T sommet() throws PileVideException;
     boolean estVide();
     int taille();
}

public class Pile<T> implements IPile<T>{
     // structure de stockage interne des éléments
     private ArrayList<T> elements;

     // Mise en oeuvre des opérations
     public Pile(){initialiser();}

     public T depiler() throws PileVideException{
          if (this.estVide())
               throw new PileVideException("en dépilant");
          T sommet = elements.get(elements.size()-1);
          elements.remove(sommet);
          return sommet;
}

     public void empiler(T t) throws PileVideException {
          elements.add(t);
          assert this.sommet()==t : "dernier empile ="+this.sommet();
     }

     public boolean estVide() {return elements.isEmpty();}

     public void initialiser() {elements = new ArrayList<T>();}

     public T sommet() throws PileVideException{
          if (this.estVide())
               throw new PileVideException("en dépilant");
          return elements.get(elements.size()-1);
     }

     public int taille(){return elements.size();}
     public String toString(){return "Pile = "+ elements;}
}
```

Question 1 *Ajouter des assertions pour contrôler :*

- *Après avoir construit la pile, celle-ci est vide.*
- *A la fin de l'opération de dépilement, le nombre d'éléments de la pile a diminué de 1.*
- *A la fin de l'opération d'empilement, le nombre d'éléments de la pile a augmenté de 1.*

Question 2 *On veut créer un nouveau type de pile, représentant les piles bornées, c'est-à-dire dont le nombre d'éléments doit rester inférieur à une certaine limite (taille maximale).*

- *Décrivez le type abstrait (en indiquant en quoi il diffère du type pile)*
- *Ecrivez les classes d'exceptions représentant les erreurs qui peuvent se produire sur les piles bornées : la pile est pleine et on ne peut plus ajouter d'éléments ; la taille maximale ne peut pas être négative*
- *Proposez une interface pour représenter le type* `Pile bornée`
- *Proposez une classe pour l'implémenter. Les méthodes comporteront des assertions et des signalements d'exceptions aux endroits nécessaires. Par défaut, la taille maximale est de 10 éléments.*
- *Ecrivez un programme utilisant cette classe. Est-ce qu'une pile bornée peut remplacer une pile : lors de la compilation (subtituabilité syntaxique) ? lors de l'exécution (subtituabilité comportementale) ?*