TAD

Semestre 2

Question 1 à 4 : spécification fonctionnelle du TAD
Question 5 à 7 : spécification algorithmique du TAD (type abstrait $\rightarrow$ type concret)

Identifier les opérations
- nombre d'éléments (cardinal)
- insérer un élément à un ensemble
- supprimer un élément d'un ensemble
- Tester l'appartenance d'un élément à un ensemble
- créer un ensemble vide

Syntaxe des opérations

$$\begin{aligned} ensembleVide &: \rightarrow Ensemble[T] \\ estVide &: Ensemble[T] \rightarrow Boolen \\ appartient &: Ensemble[T] \times T \rightarrow Boolen \\ ajouter &: Ensemble[T] \times T \rightarrow Ensemble[T] \\ supprimer &: Ensemble[T] \times T \rightarrow Ensemble[T] \\ cardinal &: Ensemble[T] \rightarrow Entier \end{aligned}$$

Préconditions des opérations
- L'opération d'ajout n'est possible que si l'élément n'appartient pas déjà à l'ensemble.
- L'opération de suppression nécessite que l'élément soit présent dans l'ensemble.

Précondition Pour ens de type Ensemble[T] et e de type T.

ajouter(ens, e) est défini si et seulement si non appartient(ens, e) ;
supprimer(ens, e) est défini si et seulement si appartient(ens, e) ;

Sémantique des opérations

ensembleVide	Générateur de base
estVide	Observateur
apartient	Observateur
ajouter	Générateur de base
suprimer	Générateur secondaire
cardinal	Observateur

Pour ens de type Ensemble[T] et e de type T.

$$
\begin{aligned}
estVide(ensembleVide) &= VRAI \\
appartient(ensembleVide, e) &= FAUX \\
cardinal(ensembleVide) &= 0 \\
estVide(ajouter(ens,\ e)) &= FAUX \\
appartient(ajouter(ens,\ e),\ e') &= si\ e = e'\ alors\ VRAI\ sin \\
cardinal(ajouter(ens,\ e)) &= cardinal(ens) + 1 \\
suprimer(ensembleVide,\ e) &= NON\ VALIDE.\ A\ SUP \\
suprimer(ajouter(ens,\ e),\ e') &= si\ e = e'\ alors\ ens\ sinon
\end{aligned}
$$

Incarner le type abstrait en un type concret

```
1  procedure créerensembleVide(sortie ens <
     Ensemble[T]>);
2
3  fonction estVide(entree ens <Ensemble[T
     ]>) retourne <Booleen>;
4
5  fonction appartient(entree ens <Ensemble[
     T]>, entree e <T>)
6    retourne <Booleen>;
7
8  procedure ajouter(maj ens <Ensemble[T]>,
     entree e <T>)
9    declenche elementExistant,
       ensemblePlein;
10
```

```
11 procedure supprimer(maj ens <Ensemble[T],
     entree e <T>);
12   declenche elementInexistant,
       ensembleVide;
13
14 fonction cardinal(entree ens <Ensemble[T
   ]>)
15   retourne <Entier>;
```

Listing 1 – Entête des opératoins

```
1 -- + propriétés en commentaire
2
3 procedure créerensembleVide(sortie ens <
   Ensemble[T]>);
4
5 fonction estVide(entree ens <Ensemble[T
   ]>) retourne <Booleen>;
6
7 fonction appartient(entree ens <Ensemble[
   T]>, entree e <T>)
8   retourne <Booleen>;
9
10 procedure ajouter(maj ens <Ensemble[T]>,
    entree e <T>)
11   declenche elementExistant,
       ensemblePlein;
12
13 procedure supprimer(maj ens <Ensemble[T],
     entree e <T>);
14   declenche elementInexistant,
```

```
        ensembleVide;
15
16 fonction cardinal(entree ens <Ensemble[T
       ]>)
17    retourne <Entier>;
```

Listing 2 – Entête des opératoins