BPOO - Support TD 5 - Eléments de solution

Dut/Info-S2/M2103

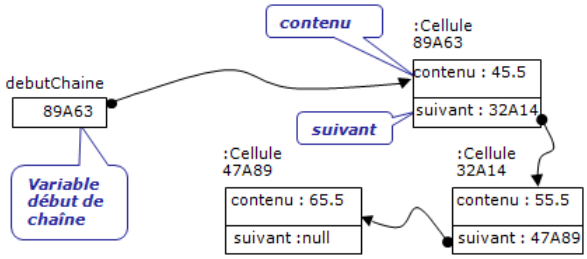
Table des matières

Version corrigée

Cette version comporte des indications pour les réponses aux exercices.

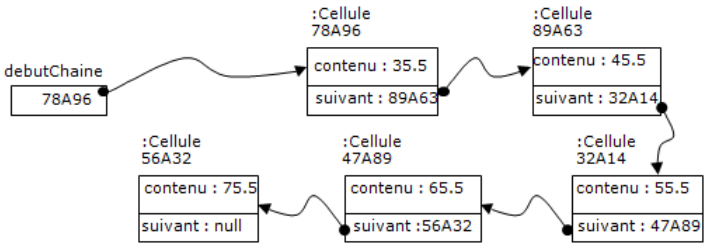
1. Eléments de solution du TD (voir sujet en parallèle)

1.1. Exemple de liste chaînée

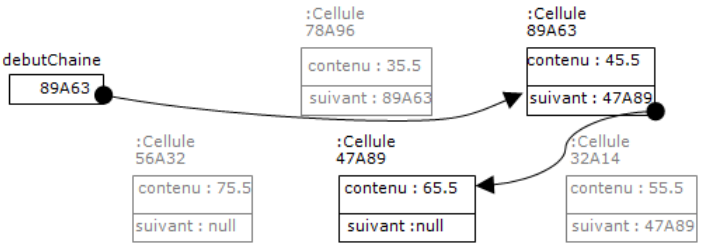


Exemple :

1. Un élément 35.5 de plus au début.
2. Un élément 75.5 de plus en fin.



3. L'élément 35.5 en moins.
4. L'élément 55.5 en moins.
5. L'élément 75.5 en moins.



6. Comment représenter une liste chaînée ne comportant aucune valeur ?

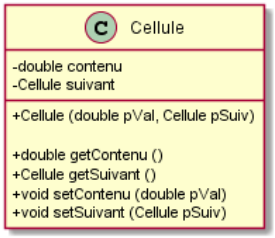
```
debutChaine vaut null; // debutChaine ne référence aucune Cellule
```

1.2. Utiliser "manuellement" une liste chaînée

1.2.1. La classe Cellule

Afin de pouvoir gérer les cellules d'une liste chaînée, on propose la classe ci-dessous :

Diagramme UML de la classe Cellule



1.2.2. Utiliser la classe Cellule

```
public class TestCelluleManuel {
  public static void main (String[] argv) {

        // PARTIE 1 - PARTIE 1 - PARTIE 1 - PARTIE 1

        // déclarer debutChaine et des variables Cellule : pCel1, pCel2, pCel3, temp, nouvCel

        Cellule debutChaine;
        Cellule pCel1, pCel2, pCel3, temp, nouvCel, temp2;

        // créer la cellule contenant 45.5 en utilisant la variable pCel1

        pCel1 = new Cellule(45.5, null);

        // créer la cellule contenant 55.5 en utilisant la variable pCel2

        pCel2 = new Cellule (55.5, null);

        // relier les cellules contenant 45.5 et 55.5

        pCel1.setSuivant(pCel2);

        // créer la cellule contenant 65.5 en utilisant la variable pCel3

        pCel3 = new Cellule (65.5, null);

        // relier les cellules contenant 55.5 et 65.5

        pCel2.setSuivant(pCel3);

        // initialiser correctement le champ suiv de la Cellule contenant 65.5
        // pour que la structure  chaînée soit correcte (c'est la dernière cellule de la structure )

        pCel3.setSuivant(null); // Mais fait au constructeur


        // initialiser la variable debutChaine pour qu'elle contienne la référence de la première
        // cellule de la chaîne que vous venez de créer

        debutChaine = pCel1;

        // afficher les différentes valeurs contenues dans la structure  chaînée dans l'ordre
        // des Cellules en utilisant les valeurs de pCel1, pCel2 et pCel3,


        System.out.println();
        System.out.println(pCel1.getContenu());
        System.out.println(pCel2.getContenu());
        System.out.println(pCel3.getContenu());

        // afficher les différentes valeurs contenues dans la structure  chaînée dans l'ordre des Cellules
        // en utilisant debutChaine et éventuellement un autre pointeur de Cellule et en utilisant une boucle
        // NE PAS se servir des valeurs de pCel1, pCel2 et pCel3

        System.out.println();
        temp = debutChaine ;
        while (temp != null) {
                System.out.println(temp.getContenu());
                temp = temp.getSuivant();
        }

        // PARTIE 2 - PARTIE 2 - PARTIE 2 - PARTIE 2

        pCel1 = null;
        pCel2 = null;
        pCel3 = null;

        // en utilisant les variables nécessaires : ajouter l'élément 35.5 avant la cellule contenant 45.5 sans utiliser

        temp = new Cellule (35.5, null);
        temp.setSuivant(debutChaine);
        debutChaine = temp;

        // en utilisant les variables nécessaires : ajouter l'élément 75.5 à la fin de la structure
        // SANS utiliser la valeur de pCel2 (mise à null entre temps)

        nouvCel = new Cellule (75.5, null);
        temp = debutChaine ;
        while (temp.getSuivant() != null) {
```

```
        temp = temp.getSuivant();
}
temp.setSuivant(nouvCel);

// afficher les différentes valeurs contenues dans la structure chaînée

System.out.println();
temp = debutChaine ;
while (temp != null) {
        System.out.println(temp.getContenu());
        temp = temp.getSuivant();
}

// PARTIE 3 - PARTIE 3 - PARTIE 3 - PARTIE 3

// supprimer la cellule contenant 55.5.
// on supposera ne pas "connaître" sa position. Il faut donc chercher la cellule.
// on sait que ce n'est pas la première et on sait qu'elle existe !!

temp = debutChaine ;
while (temp.getSuivant().getContenu() != 55.5) {
        temp = temp.getSuivant();
}
temp2 = temp.getSuivant(); // Cellule contenant 55.5
temp2 = temp2.getSuivant(); // Cellule suivante de la cellule contenant 55.5
temp.setSuivant(temp2);

// supprimer la cellule contenant 75.5.
// on supposera ne pas "connaître" sa position. Il faut donc chercher la cellule.
// on sait que ce n'est pas la première et on sait qu'elle existe !!
// est ce si différent que ci-dessus ?

temp = debutChaine ;
while (temp.getSuivant().getContenu() != 75.5) {
        temp = temp.getSuivant();
}
temp.setSuivant(temp.getSuivant().getSuivant());

// supprimer la cellule contenant 35.5 (une ligne)

debutChaine = debutChaine.getSuivant();

// afficher les différentes valeurs contenues dans la structure chaînée

System.out.println();
temp = debutChaine ;
while (temp != null) {
        System.out.println(temp.getContenu());
        temp = temp.getSuivant();
}

//Enfin, chercher un algorithme permettant la suppression d'un élément (ou valeur, appelons-la valASup) n'import

float valASup;
Cellule pred;
boolean trouve;

// SOLUTION 1 :

valASup = // A Définir ... ;

temp = debutChaine;
pred = null;
trouve = false;

while (temp != null && ! trouve) {
        if (temp.getContenu() == valASup) {
                trouve = true;
        } else {
                pred = temp;
                temp = temp.getSuivant();
        }
}
if (trouve) {
        if (pred == null) {
                debutChaine = debutChaine.getSuivant();
        } else {
                pred.setSuivant(temp.getSuivant());
        }
}

// SOLUTION 2 :

valASup = // A Définir ... ;

if (debutChaine != null) {
        if (debutChaine.getContenu() == valASup) {
                debutChaine = debutChaine.getSuivant();
        } else {
                trouve = false;
                temp = debutChaine;
                while (temp.getSuivant() != null && ! trouve) {
                        if (temp.getSuivant().getContenu() == valASup) {
                                trouve = true;
                        } else {
```

```
                                        temp = temp.getSuivant();
                                }
                        }
                        if (trouve) {
                                temp.setSuivant(temp.getSuivant().getSuivant());
                        }
                }

        }

        // Affichage

        System.out.println();
        temp = debutChaine ;
        while (temp != null) {
                System.out.println(temp.getContenu());
                temp = temp.getSuivant();
        }
}
```

Dernière mise à jour 2016-03-18 09:13:04 CET