

## Génie Logiciel TD n° 1, 2

### Jeux de tests

#### Objectifs du TD

Ce(s) TD, premier(s) de l'UE, est une sorte de révision sur Java. Ses deux objectifs sont :

- Réviser Java.
- Comprendre le but d'un jeu de tests.

En annexe figure le programme Java `Chimie` permettant de créer, associer, dissocier et détruire atomes et molécules. Le but des questions suivantes est de **tester** et **corriger** ce programme.

#### Question 1

Connaissez vous le langage Java ?

Si c'est le cas vous pouvez passer directement à la question 2.

Sinon, si vous connaissez C++, il est conseillé de faire le sujet analogue à celui-ci sur les jeux de tests mais utilisant C++ comme support. Ce sera plus facile.

Cependant, si vous connaissez C++ et pas Java, C++ et Java se ressemblant beaucoup, vous pouvez malgré tout passer à la question 2 en vous disant que, en plus de tester et corriger le programme, vous allez commencer votre apprentissage de Java.

#### Question 2

- Quelles sont les méthodes redéfinies ?
- Quelle est la différence entre les méthodes `toNom()` et `toString()` ?
- A quoi sert la classe `VectorE` ?

#### Question 3

Java ne peut compiler `menuPrincipal()`. Pourquoi ? Rajouter le mot-clé `static` là où cela est nécessaire.

#### Jeux de tests

Le programme `Chimie` contient encore des erreurs et chaque question 4 à 8 va permettre de déceler une erreur.

Pour chaque question suivante, on donne un jeu de test effectué par un utilisateur, et on demande :

- a) le but du test,
- b) la sortie *erronée* du programme,
- c) l'explication de l'erreur,
- d) les lignes corrigeant l'erreur dans le source,
- e) la sortie *correcte* du programme.

Un jeu de test est composé d'une suite de caractères tapés au clavier par l'utilisateur du logiciel Chimie. Un espace ou un saut de ligne de la suite de caractères correspond à la touche "entrée" du clavier.

Le but du test (a) et l'explication de l'erreur (c) seront donnés en quelques mots.

Dans les sorties du programme (b, e), on supprimera l'affichage des menus Atome, Molecule et Chimie.

L'erreur (b) et sa correction (e) seront mises en évidence clairement en soulignant la partie de la sortie qui est erronée ou corrigée.

L'explication de l'erreur (c) contiendra éventuellement des diagrammes d'instances UML.

On indiquera clairement le lieu des lignes Java corrigeant l'erreur (e).

#### Question 4

```
A C java A P Q
```

#### Question 5

```
M C genie P A C java P M J genie java A P A A D java A P M A P Q
```

#### Question 6

```
M C genie P A C java P M J genie java A P
A A P M D genie A P A A P Q
```

#### Question 7

```
M C genie C logiciel P A C java C uml P
M J genie java J logiciel java A P A A P Q
```

#### Question 8

```
M C genie C logiciel P A C java C uml P
M J genie java J logiciel uml A R genie uml A P A A P Q
```

## Annexe

```

import java.util.*;

class Chimie {

    public static void main(String argv[]) {
        System.out.println( "Chimie, bonjour,");
        Chimie ch = new Chimie();
        while (ch.menuPrincipal()!='Q');
        System.out.println( "Chimie, au revoir.");
    }

    public char menuPrincipal() {
        System.out.println( "Chimie :");
        System.out.println( "\t M. Molecule");
        System.out.println( "\t A. Atome");
        System.out.println( "\t Q. Quitter");
        char r = Keyboard.getChar();
        switch(r) {
            case 'M': Molecule.boucle(); break;
            case 'A': Atome.boucle(); break;
            case 'Q': ; break;
        }
        return r;
    }
}

class Entite {

    public String nom;

    public Entite(String n) { nom = n; }

    public String toString() { return ( "\n" + toNom()); }

    public String toNom() { return ( "nom " + nom + "\n"); }
}

class VectorE extends Vector {

    public String toNom() {
        String s = "[\n";
        for (int x=0; x< size(); x++) {
            Entite e = (Entite) elementAt(x);
            if (e==null) return s+"]\n";
            s = s + e.toNom() + "\n,\n";
        }
        return s+"]\n";
    }

    public Entite getInstance(String n) {
        for (int x=0; x<size(); x++) {
            Entite i = (Entite) elementAt(x);
            if (i==null) return null;
            if (i.nom.equals(n)) return i;
        }
        return null;
    }
}

```

```
class Molecule extends Entite {

    public static VectorE mesInstances = new VectorE();

    public VectorE mesAtomes;

    public Molecule(String n) {
        super(n);
        mesAtomes = new VectorE();
        mesInstances.addElement(this);
    }

    public void detruire() {
        mesInstances.removeElement(this);
    }

    public String toString() {
        return ("\n"+toNom()+"\nMes atomes: "+mesAtomes.toNom()+"\n");
    }

    public void boucle() {
        while (menu()!='P');
    }

    public static char menu() {

        System.out.println( "Molecule :");
        System.out.println( "\t C. Creation");
        System.out.println( "\t J. aJouter un atome");
        System.out.println( "\t R. Retirer un atome");
        System.out.println( "\t D. Destruction");
        System.out.println( "\t A. Affichage");
        System.out.println( "\t P. Precedent");

        char r = Keyboard.getChar();

        switch(r) {
            case 'C': creation(); break;
            case 'J': ajoutAtome(); break;
            case 'R': retraitAtome(); break;
            case 'D': destruction(); break;
            case 'A': System.out.println(mesInstances); break;
        }

        return r;
    }

    public static void creation() {
        System.out.println("\nCreation d'une molecule: ");
        System.out.print("\nNom de la molecule ? ");
        String s = Keyboard.getString();
        Molecule m = new Molecule(s);
        System.out.println("\nMolecule '" + s + "' creee.");
    }
}
```

```

public static void destruction() {
    System.out.print("\nDestruction d'une molecule: \n");
    System.out.print("\nNom de la molecule ? ");
    String s = Keyboard.getString();
    Molecule m = (Molecule) mesInstances.getInstance(s);
    if (m!=null) {
        m.detruire();
        System.out.println("\nMolecule '" + s + "' detruite.");
    }
    else System.out.println("\nPas de molecule avec le nom '" + s + "'.");
}

```

```

public static void ajoutAtome() {
    System.out.print("\nAjout d'un atome a une une molecule: \n");
    System.out.print("\nNom de la molecule ? ");
    String sm = Keyboard.getString();
    Molecule m = (Molecule) mesInstances.getInstance(sm);
    if (m==null) {
        System.out.println("\nPas de molecule avec le nom '" + sm + "'.");
        return;
    }
    System.out.print("\nNom de l'atome ? ");
    String sa = Keyboard.getString();
    Atome a = (Atome) Atome.mesInstances.getInstance(sa);
    if (a==null) {
        System.out.println("\nPas d'atome avec le nom '" + sa + "'.\n");
        return;
    }
    m.mesAtomes.addElement(a);
    a.maMolecule = m;
    System.out.println("\nAtome '" + sa
        + "' ajoute a la molecule '" + sm + "'.");
}

```

```

public static void retraitAtome() {
    System.out.print("\nRetrait d'un atome d'une une molecule: \n");
    System.out.print("\nNom de la molecule ? ");
    String sm = Keyboard.getString();
    Molecule m = (Molecule) mesInstances.getInstance(sm);
    if (m==null) {
        System.out.println("\nPas de molecule avec le nom '" + sm + "'.");
        return;
    }
    System.out.print("\nNom de l'atome ? ");
    String sa = Keyboard.getString();
    Atome a = (Atome) Atome.mesInstances.getInstance(sa);
    if (a==null) {
        System.out.println("\nPas d'atome avec le nom '" + sa + "'.");
        return;
    }
    m.mesAtomes.removeElement(a);
    a.maMolecule = null;
    System.out.println("\nAtome '" + sa
        + "' retire de la molecule '" + sm + "'.");
}
}

```

```

class Atome extends Entite {

    public static VectorE mesInstances = new VectorE();
    public Molecule maMolecule;

    public Atome(String n) {
        super(n);
        maMolecule = null;
        mesInstances.addElement(this);
    }

    public void detruire() {
        mesInstances.removeElement(this);
    }

    public String toString() {
        return ( "\n" + toNom() + toMolecule());
    }

    public String toMolecule() {
        return ("ma molecule: \n" + maMolecule.toNom() + "\n");
    }

    public void boucle() {
        while (menu()!='P');
    }

    public static char menu() {
        System.out.println( "Atome :");
        System.out.println( "\t C. Creation");
        System.out.println( "\t D. Destruction");
        System.out.println( "\t A. Affichage");
        System.out.println( "\t P. Precedent");
        char r = Keyboard.getChar();
        switch(r) {
            case 'C': creation(); break;
            case 'D': destruction(); break;
            case 'A': System.out.println(mesInstances); break;
        }
        return r;
    }

    public static void creation() {
        System.out.print("\nCreation d'un atome: \n");
        System.out.print("\nNom de l'atome ? ");
        String s = Keyboard.getString();
        Atome a = new Atome(s);
        System.out.println("\nAtome '" + s + "' cree.");
    }

    public static void destruction() {
        System.out.println("\nDestruction d'un atome: ");
        System.out.print("\nNom de l'atome ? ");
        String s = Keyboard.getString();
        Atome a = (Atome) mesInstances.getInstance(s);
        if (a!=null) {
            a.detruire();
            System.out.println("\nAtome '" + s + "' detruit.");
        }
        else System.out.println("\nPas d'atome portant le nom '" + s + "'.");
    }
}

```