TD 5 : étude d'un applet simple, classe `Vector`

1 Étude syntaxique d'un applet

Il s'agit dans cet exercice d'expliquer ce qu'est *syntaxiquement* chaque élément d'un petit programme applet. On ne demande pas de décrire ce que fait chaque classe ou chaque méthode. Pour chaque ligne, il y a, en général, plusieurs éléments à expliquer. Par exemple :

```
int a=3;
```

donne *déclaration d'un entier (type primitif, pas un objet) et initialisation à 3.*

```
import javax.swing.JApplet;
import java.awt.Graphics;
import java.awt.Color;
import java.awt.event.*;

public class ACommenter extends JApplet implements MouseListener
{
    int lx,ly;
    public void paint(Graphics g)
    {
        double r=Math.random();
        g.setColor(Color.green);
        g.setColor(new Color(.1f,.5f,.9f));
        g.drawLine(0,0,lx,ly);
    }
    public void init()
    {
        addMouseListener(this);
    }
    public void mouseEntered (MouseEvent e) {;}
    public void mouseExited  (MouseEvent e) {;}
    public void mousePressed (MouseEvent e) {;}
    public void mouseReleased(MouseEvent e) {;}
    public void mouseClicked (MouseEvent e)
    {
        System.out.println("PositionA:"+
                           e.getX()   +","+
                           e.getY());
        lx=e.getX();
        ly=e.getY();
        repaint();
    }
}
```

2 Étude sémantique d'un applet

Il s'agit dans cet exercice d'expliquer ce que fait (ou à quoi sert) chaque élément du petit programme applet vu précédemment. Notez bien qu'il s'agit d'un travail très différent de celui effectué précédemment. On demande des commentaires pour presque toutes les lignes.

3 Classe `Vector`

L'API java fournit une classe `Vector` qui permet de stoker des éléments. Nous allons, dans cet exercice, reimplémenter une classe `Vector` semblable. Voici un exemple de l'utilisation que l'on souhaiterai pouvoir faire de cette classe :

```
import java.awt.Point;
class Lapin
{
    String nom;
    public String toString(){return nom;}
    Lapin(String _nom){nom=_nom;}
};
class VectorTest
{
    public static void main(String args[])
    {
        Vector v=new Vector();
        v.add(new Point(1,2));
        v.add(new Lapin("roger"));
        v.add(1,new Lapin("titi"));
        for(int i=0;i<v.size();i++)
        {
            // affichage, utilise toString de chaque Object dans v
            System.out.println(v.get(i));
        }
        // enleve l'element à l'indice 0
        v.remove(0);
        // affichage, utilise toString de Vector
        System.out.println(v);
        v.clear();
    }
}
```

Notez bien que la méthode `add` permet d'ajouter un élément à la fin du `Vector`, et ce autant de fois qu'on veut. Nous utiliserons un tableau pour stoker les données du `Vector`.