Chapitre 7 : Travaux Pratique

02/05/2022

1 La classe `Caractere`

On va manipuler un caractère quantitatif à valeurs entières, en donnant que les valeurs et les effectifs (entiers).
Ecrire la classe `Caractere` contenant les méthodes suivantes :

- `private Caractere(int[] xi, int[] ni)`
  un constructeur privé qui construit en recopiant le tableau des valeurs et celui des effectifs. On doit préserver l'encapsulage, donc garantir l'intégrité des données contenues dans l'objet.
- `public static Caractere fabrique(int[] xi, int[] ni)`
  permet de construire un objet `Caractere` valide :
  - il faut des valeurs,
  - le nombre de valeurs doit être égal au nombre d'effectifs,
  - les valeurs doivent être strictement croissantes,
  - les effectifs doivent être strictement positifs.
- `public int effectif(int valeur)`
  retourne l'effectif de `valeur`
- `public int effectifTotal()`
  retourne l'effectif total.
- `public int frequence(int valeur)`
  retourne la fréquence de `valeur`
- `public int frequenceCumulee(int valeur)`
  retourne la fréquence cumulée de `valeur`, c'est-à-dire la proportion de la population dont le caractère est inférieur ou égal à `valeur`.
- `public int mode()`
  retourne le mode.
- `public double moyenne()`
  calcule la moyenne.
- `public double moment2()`
  calcule le moment d'ordre 2, c'est-à-dire : $\frac{1}{n}\sum_i n_i \times x_i^2$.
- `public double variance()`
  calcule la variance.
- `public double ecartType()`
  calcule l'écart-type.

— `public double frequenceIntervalle(double t)`
calcule la proportion de la population dont le caractère est compris entre $m - t \times \sigma$ et $m + t \times \sigma$Directeur des études de 1ère année jusqu'au 31 août 2022 Département Informatique IUT Le Havre
— `public int getNbValeur()`
donne le nombre de valeurs.
— `public int getValeur(int i)`
donne la valeur d'indice `i`.
— `public int getEffectif(int i)`
donne l'effectif d'indice `i`.
— `public String toString()`
donne une chaine de caractères où on a le tableau des valeurs, des effectifs, des fréquences, des fréquences cumulées, ainsi des informations comme l'effectif total, la moyenne, l'écart-type, le mode et la proportion de la population dont le caractère est compris entre $m - 2\sigma$ et $m + 2\sigma$

2 Les classes `Fenetre` et `Plateau`

Il faut étudier le code de ces deux classes. Ici on ne respecte pas le modèle MVC, mais vous avez le droit de le respecter. On a fait un aspect très primitif, vous avez aussi le droit d'améliorer ceci.
La classe `Fenetre` permet de créer un `Plateau` et de l'afficher dans une fenêtre qu'on peut fermer.

```
public class Fenetre
{
   public static void voir(JComponent component, String nom)
   {
      JFrame frame = new JFrame(nom);
      WindowAdapter wa = new WindowAdapter()
      {
         public void windowClosing(WindowEvent e) {System.exit(0);}
      };
      frame.addWindowListener(wa);
      frame.getContentPane().add(component);
      frame.pack();
      frame.setVisible(true);
   }

   public static void main(String[] arg)
   {
      Plateau plateau = new Plateau();
      plateau.setBackground(Color.WHITE);
      plateau.setPreferredSize(new Dimension(400,200));
      voir(plateau,"test");
   }
}
```

La classe `Plateau` permet d'afficher des rectangles, des lignes et des arcs pleins, de plusieurs couleurs.

```
public class Plateau extends JPanel
{
   public void paint(Graphics g)
   {
      super.paint(g);
      Color c = g.getColor();

      g.setColor(Color.RED);
      g.fillRect(10,10,80,80);

      g.setColor(Color.BLUE);
      g.fillOval(150,50,80,80);

      g.setColor(Color.PINK);
      g.drawOval(200, 150, 50, 50);

      g.setColor(Color.CYAN);
      g.drawRect(20, 100, 50, 50);

      g.setColor(Color.BLACK);
      g.fillArc(250, 100, 50, 50, 0, 45);
      g.setColor(Color.GREEN);
      g.fillArc(250, 100, 50, 50, 45,30);
      g.setColor(Color.PINK);
      g.fillArc(250, 100, 50, 50, 75,105);

      g.setColor(Color.BLACK);
      g.drawLine(320,10,320,160);
      for(int i=0;i<=15;i++)
      {
         g.drawLine(310,10+i*10,330,10+i*10);
         g.drawString(""+i,335,10+i*10);
      }

      g.setColor(c);
   }
}
```

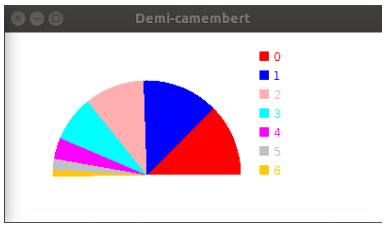
Etudiez la javadoc, pour comprendre les paramètres de ces méthodes.

3 Différentes représentations graphiques

Ecrire les classes suivantes. On leur fournit un objet `Caractere`, et elle dessine le diagramme correspondant. On a été basique, mais vous avez le droit de faire mieux.

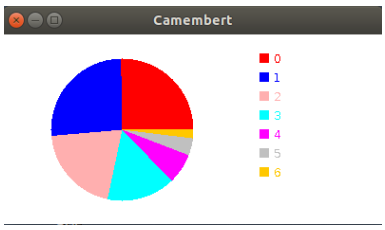
3.1 La classe `DemiCamembert`

Cette classe permet d'afficher un demi-camenbert d'un caractère.



3.2 La classe `Camembert`

Cette classe permet d'afficher un camenbert d'un caractère.



3.3 La classe `DiagrammeABarres`

Dans un diagramme en bâtons vertical, on représente pour chaque modalité d'un caractère discret un rectangle dont la hauteur représente l'effectif (ou la fréquence) et dont la largeur n'a pas d'interprétation statistique.