Licence Informatique 2ème année

Programmation Objet 1

TP n°1 – *Modularité, Héritage*

1- Premier programme Java

1.1 Placez le code suivant dans un fichier Personne.java, compilez et exécutez le.

```
class Personne{
  String nom;
  int age;
  Personne(String nom, int age){
    this.nom = nom;
    this.age = age;
  }
  String getNom(){ return this.nom; }
  int getAge(){ return this.age; }
  void setAge(int a){ this.age = a; }
}
```

NB : normalement, on crée un fichier par classe (modularité). C'est même obligatoire si les classes sont publiques (cette notion sera vue plus tard dans la partie encapsulation du cours).

1.2 Pour qu'il se passe quelque chose, ajoutez une méthode principale qui crée 2 personnes, affiche leurs noms et âges, ajoute 1 à leurs âges, et réaffiche leurs âges.

2- Liste et cryptage

2.1 Écrire une classe Liste qui représente une liste chaînée d'objets. Une liste chainée contient une valeur (de type Object ici) et une "queue", qui est le reste de la liste. Cette classe offrira au moins des méthodes permettant de :

- créer une liste vide
- ajouter un objet à la fin de la liste
- ajouter un objet en tête de liste
- connaitre la position d'un objet donné dans la liste
- récupérer l'objet situé à une position donnée

2.2 Dans une méthode principale, créer une liste de chaines de caractères aléatoires, la parcourir et afficher les chaines. Voici une fonction qui génère une chaine de caractères alphabétiques minuscules aléatoires :

```
static String randomString(int length){
  char[] t = new char[length];
  for(int i=0;i<length;i++) t[i] = (char) ('a'+(int) (Math.random()*26));
  return new String(t);
}
```

2.3 Créer une liste de listes de chaines et afficher les chaines.

2.4 Créer une classe Crypteur dont chaque instance peut crypter un texte selon le principe du code de César, ou décalage circulaire : à chaque lettre correspond la lettre située n places plus loin dans l'alphabet. Un objet de type Crypteur doit opérer sur deux listes de caractères, la première contient les lettres de a à z, la seconde les lettres décalées. Exemple de listes de cryptages pour un décalage de 3 :

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c

2.5 Dans une méthode principale, créer deux crypteurs avec des décalages différents et leur faire crypter la même chaine de caractères.

3- <u>Simulation de foule</u>

On veut simuler le comportement d'une foule. 3 classes sont déjà écrites :

- une classe Displayable dont les instances peuvent s'afficher sur une interface graphique
- une classe CrowdFrame qui affiche des Displayable dans une fenêtre
- une classe CrowdGame, incomplète, qui fait tourner la simulation

Toutes les personnes qu'on veut afficher, en plus d'être des Displayable doivent avoir une direction et une vitesse. De plus, les personnes s'affichent sous forme d'un rond complété par un petit trait qui indique la direction vers où la personne se déplace. Toutes les personnes ont également une méthode void move() qui déplace la personne en fonction de sa direction et de sa vitesse.

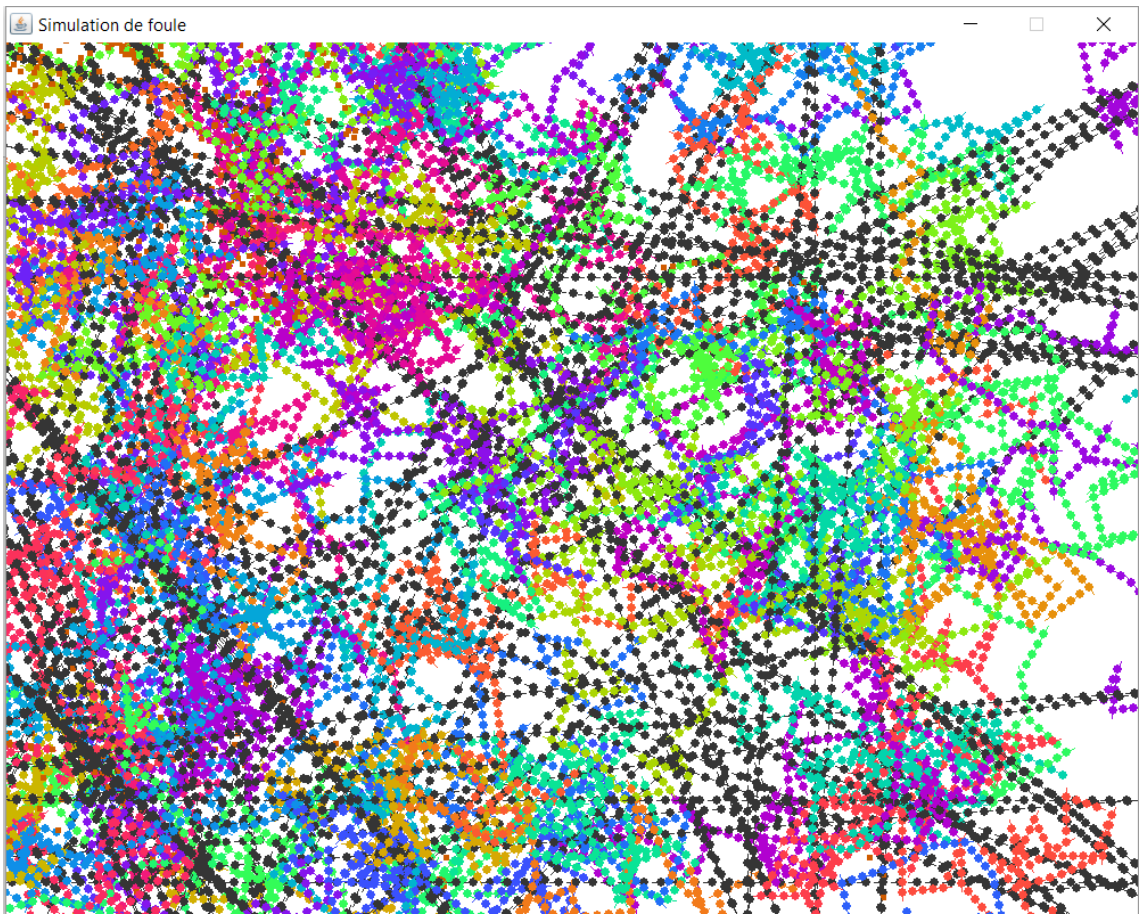
3.1 Sans rien changer aux classes Displayable et CrowdFrame, écrire une classe qui représente les personnes.

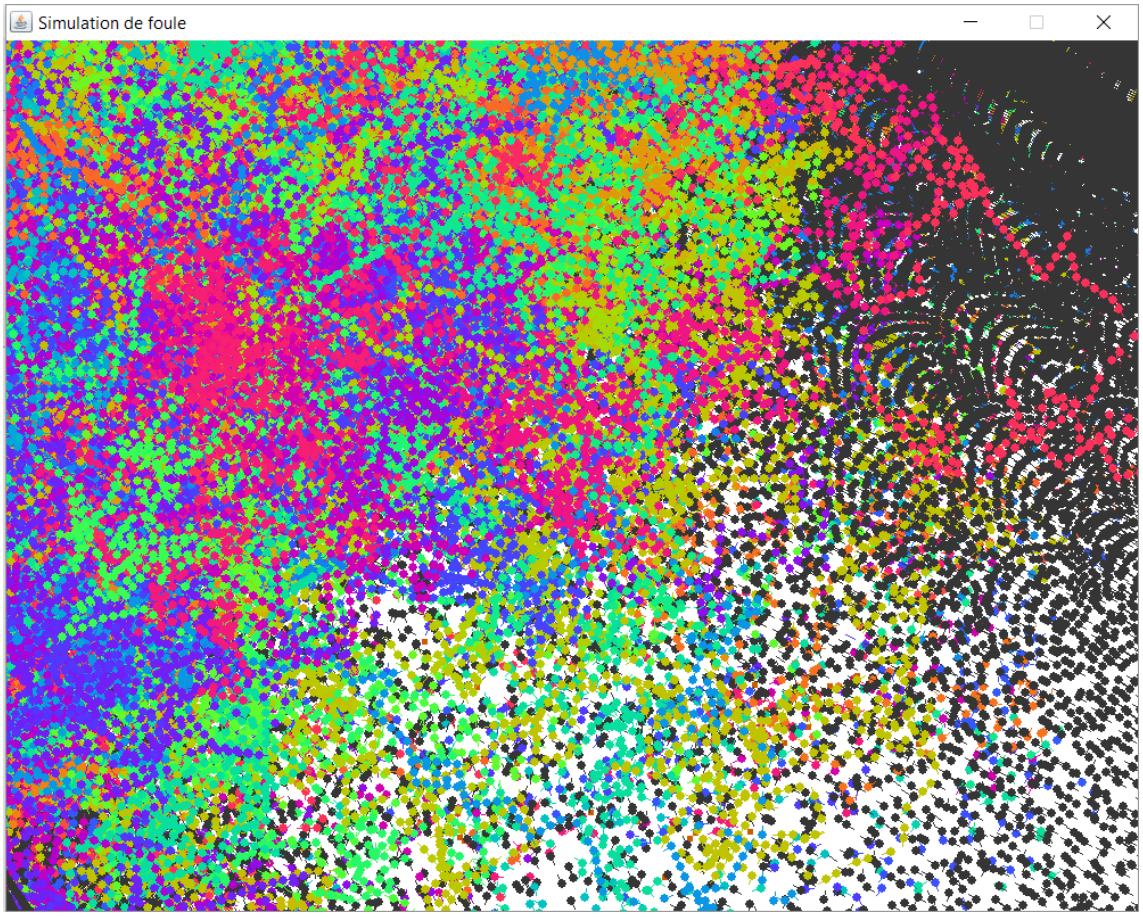
Il y a au moins deux types de personnes :

- les *travailleurs*, qui ont une couleur grise et marchent toujours dans la même direction à une vitesse plus élevée que les autres personnes.
- les *touristes*, qui changent de direction aléatoirement à chaque appel de la méthode move, ou avec une certaine périodicité et sont de toutes les couleurs.

3.2 Écrire les classes représentant les travailleurs et les touristes

3.3 Compléter la classe CrowdGame pour générer des personnes des deux types et faire tourner la simulation (mettre la ligne 22 de CrowdFrame.java en commentaire pour afficher les positions successives).


Simulation de foule


Simulation de foule