TP n°2 – Objets et graphisme

Projet de programmation (IK2)

Objets

Exercice 1 — *Date* Le but de cet exercice est d'écrire une classe Date pour manipuler facilement les dates du calendrier.

1. Écrire une classe `Date` qui représente une date comme : un entier représentant le jour du mois, un autre le mois, et un autre l'année. Ce sont des champs privés (`private`).
2. Écrire un constructeur pour cette classe qui prend comme paramètres trois entiers.
3. Écrire une méthode d'affichage d'une date, avec les noms des jours et des mois écrits en toutes lettres.
4. Écrire une méthode statique qui lit n dates entrées sous la forme de trois entiers séparés par un espace et qui stocke ces dates dans un tableau. Écrire une méthode qui affiche ce tableau.
5. Écrire une méthode de comparaison de deux dates.
6. Écrire une méthode qui permet d'ajouter à une date une période donnée, par exemple : 15 jours, 2 semaines, etc.

Exercice 2 — *Personne* Écrire une classe `Personne` contenant un nom, un prénom et une date de naissance (un objet de la classe Date).
Écrire un constructeur explicite permettant de créer un objet de la classe `Personne` (qui prend comme paramètres deux chaînes de caractères et une date) et une méthode permettant d'afficher un objet de la classe `Personne`. Cette méthode utilisera, bien sûr, la méthode d'affichage de la classe `Date` pour afficher la date de naissance. Écrire une méthode statique de test pour cette classe.

Exercice 3 — *Bibliothèque* Le but de cet exercice est d'écrire une classe gérant une bibliothèque.

1. Écrire une classe `Livre` contenant les champs **privés** suivants : `titre`, `auteur`, `dateEmprunt`, `nomEmprunteur`, `dureePret` et `enRayon`. On utilisera les classes Date et Personne pour les champs qui correspondent. On ajoutera le(s) constructeur(s) nécessaire(s), les méthodes permettant d'accéder à ces champs ainsi que la méthode `void affiche ()`.
   On implémentera aussi la méthode `public boolean enRetard(Date date)`.

2. Écrire une classe `GestionBibli` qui gère une bibliothèque contenant un tableau de `Livre`, une liste de personnes inscrites (tableau de `Personne`) ainsi que la date courante. On y ajoutera les méthodes suivantes :
   - `void ajouteLivre(Livre bouquin)`
   - `void supprimeLivre(Livre bouquin)`
   - `void nouvelleInscription(Personne nom)`
   - `void supprimeInscription(Personne nom)`
   - `void gererInscription()` interface avec l'utilisateur : cette méthode va demander à l'utilisateur si il veut faire une nouvelle inscription, ou supprimer une personne existante (à vérifier).
   - `void emprunt(Livre bouquin)`

- `void retour(Livre bouquin)`
- `void incrementeDate()`
- `void afficheListeLivres()`
- `void afficheListeInscrits()`
- `void afficheLivresEnRayon()`
- `void afficheLivresEmpruntes()`
- `void afficheEnRetard()` (livres qui devraient être rendu mais qui ne le sont pas)
- `void gererLivre()` cette méthode propose un accès aux fonctions de retour, emprunt, prolongation et d'affichage avec un menu simplifié, où l'utilisateur devra entrer des entiers.

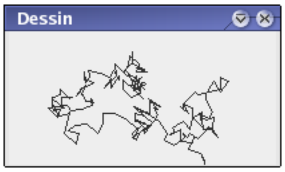
3. Écrire une classe `Bibli` contenant le `main` pour tester votre programme.

Graphisme

Exercice 4 — *Carrés concentriques.* On dira que des carrés sont concentriques s'ils ont le même centre, le centre d'un carré étant le point d'intersection de ses diagonales.

1. Ecrire une méthode qui affiche un carré et qui prend en argument son centre et la taille de ses côtés.
2. Ecrire un programme qui affiche un carré en haut à gauche de la fenêtre graphique.
3. Ecrire un programme qui affiche 10 carrés concentriques dont la taille des cotés varie de 10 à 100. Le centre des carrés sera demandé à l'utilisateur.

Exercice 5 — *Boit-sans-soif.* Boit-sans-soif est au milieu de l'écran, il se déplace selon la règle suivante : il choisit aléatoirement sa nouvelle position dans un carré de 10 pixels autour de sa position courante et s'arrête lorsqu'il a atteint le bord de la fenêtre graphique.



Écrire un programme traçant le chemin parcouru par Boit-sans-soif. Vous utiliserez la fonction `Math.random()` qui retourne une valeur de type `double` supérieure ou égale à 0 et strictement inférieure à 1.

Exercice 6 — *Mouvement.* Pour dessiner un objet qui se déplace, on l'affiche à sa position initiale, on l'efface, puis on l'affiche un peu plus loin et on recommence. Vous pourrez utiliser la méthode suivante pour ralentir l'exécution :

```
public static void sleep(){
  try {
    Thread.sleep(100);
  } catch(Exception e) {}
}
```

1. Écrire un programme qui déplace un cercle de gauche à droite.
2. Écrire un programme qui déplace un cercle du haut vers le bas.
3. Écrire un programme qui déplace un cercle le long d'une diagonale.
4. Écrire un programme qui déplace un cercle d'un point de coordonnées (x_1, y_1) jusqu'à un point de coordonnées (x_2, y_2).

Exercice 7 — *Labyrinthe.* On représente un labyrinthe par un tableau de booléens à deux dimensions. Une case a la valeur `true` s'il y a un mur et `false` sinon.
Écrire un programme qui affiche un labyrinthe.