JAVA :

Mercredi :

1. La programmation orienté objet :

* La programmation orientée objet est un paradigme de Programmation qui permet de représenter des objets du monde réel en utilisant des classes et des objets.

1. Les classes :

* Voici la syntaxe pour créer une classe en java :

public class NomDeLaClasse {

// attributs / champs

// constructeurs

// methode

}

* Les classes sont constituées de champs, de constructeurs et de méthodes. Les champs représentent les données de la classe, les constructeurs permettent de créer des objets et les méthodes permettent de manipuler les objets.
* A ces notions fondamentales vont s'ajouter trois concepts importants :
  + L’encapsulation
  + L’héritage
  + Le polymorphisme
* L'encapsulation permet de protéger les données d'une classe en les rendant privées. On va ainsi cacher certains éléments qui ne sont pas nécessaire à l'utilisateur de la classe, mais nécessaire a son bon fonctionnement. Dans le cas d'une classe Article, l'utilisateur n'a pas besoin de connaitre le prix de revient de l'article, mais il a besoin de connaitre son prix de vente. Dans le cas d'une voiture, l'utilisateur n'a pas besoin de connaitre la vitesse de rotation du moteur, mais il a besoin de connaitre la vitesse de la voiture.
* Les éléments de la classes visibles depuis l'extérieur constitue l'interface de la classe.
* L'héritage permet de créer des classes filles à partir d'une classe mère. Autrement dit on crée une nouvelle classe à partir d'une classe existante. La classe fille hérite de tous les éléments de la classe mère. On peut alors ajouter des éléments supplémentaires à la classe fille.
* Le polymorphisme permet de manipuler des objets de classes différentes avec une même interface. Ce concept est un peu difficile a comprendre au départ, mais il est très important en programmation orientée objet. Il est possibles d'utiliser plusieurs classes différentes qui ont une même interface. Deux autres concepts sont souvent associes au polymorphisme : la surcharge et la substitution.

1. Mise en pratique avec JAVA :

Une image contenant texte, Police, diagramme, ligne

Description générée automatiquement

* Déclaration d’une classe

[modifecateur] class NomDeLaClasse {

Implements NomDe

}

1. Les modificateurs de classes :

* public : la classe est visible depuis n'importe quel autre classe. Sans ce modificateur, la classe est visible uniquement depuis le package dans lequel elle est déclarée.
* abstract: la classe est abstraite. Une classe abstraite ne peut pas être instanciée.
* final: la classe ne peut pas être héritée.
* Les méthodes de classes :

[modificateur] typeDeRetour nomDeLaMethode([parametres]) {

// code source de la methode

}

1. Les modificateurs de méthodes :

* public : la méthode est visible depuis n'importe quel autre classe. Sans ce modificateur, la méthode est visible uniquement depuis le package dans lequel elle est déclarée.
* protected : la méthode est visible depuis la classe, le package et les classes filles.
* private : la méthode est visible uniquement depuis la classe.
* static : la méthode est une méthode de classe. Elle peut être utilisée sans instancier la classe.
* abstract : la méthode est abstraite. Elle doit être redéfinie dans les classes filles.
* final : la méthode ne peut pas être redéfinie dans les classes filles.
* native : la méthode est implémentée dans un langage autre que Java.
* synchronized : la méthode ne peut etre utilisée que par un seul thread a la fois.

1. Création d'accesseurs pour les champs prives (getter et setter):

public typeDeRetour getNomDuChamp() {

return nomDuChamp;

}

public void setNomDuChamp(typeDuChamp nomDuChamp) {

this.nomDuChamp = nomDuChamp;

}

* Les constructeurs d'une classe :

Il s'agit d'une méthode particulière qui permet de créer des objets à partir d'une classe. Le constructeur porte le même nom que la classe et n'a pas de type de retour. Il est possible de créer plusieurs constructeurs dans une classe. On parle de surcharge de constructeurs. Si aucun constructeur n'est déclaré dans une classe, un constructeur par défaut est créé par le compilateur. Ce constructeur par défaut est un constructeur vide qui ne fait rien.

public NomDeLaClasse() {

// code source du constructeur

}

1. Les packages

* Un package est un ensemble de classes regroupées dans un même répertoire.
* Les packages permettent d'organiser les classes en catégories.
* Les packages permettent de protéger les classes en les rendant privées.
* Les packages permettent de gérer les conflits de noms de classes.
* Les packages permettent de contrôler l'accès aux classes.
* Les packages permettent de créer des bibliothèques de classes.
* Les packages permettent de créer des classes qui ne sont pas visibles depuis l'extérieur.
* Les packages permettent de créer des classes qui ne sont visibles que depuis le package.

1. Sujet examen pratique

Créez un programme console java qui permet de simuler l'interaction avec un ATM (Distributeur Automatique de Billets). Pour cela, il est conseillé de créer un diagramme de classe UML avant de commencer à coder. Il devra figurer dans le dépôt git. Il devra présenter le nom des classes, les attributs et les méthodes. Ainsi que les éventuels liens entre les classes.

L'ATM devra permettre de gérer les comptes bancaires des clients. Il devra permettre de créer un compte bancaire, de consulter le solde, de déposer de l'argent, de retirer de l'argent, de consulter l'historique des opérations.

L'ATM devra permettre de gérer son compte client. Il devra permettre de modifier les infos du client (nom, prénom, adresse, code postal, ville).

Afin d'éviter que votre programme ne s'arrête pensez à configurer un menu qui permettra de choisir les actions à

effectuer. On maintient ainsi le programme en vie tant que l'utilisateur ne décide pas de le quitter. Utilisez une boucle while pour cela.