

GROUPE	40 BOC, Alix	boc.alix@uqam.ca	(514) 987-3000 4808	PK-4660
Jeudi, de 17h30 à 20h30 (cours) – Jeudi, 20h30 à 22h30 (ateliers)				

DESCRIPTION Être initié aux concepts fondamentaux de la programmation orientée objet, avec le même langage que le cours INF7212. Comprendre les possibilités et les limites du langage choisi. Orientation objet (OO) comme technique d'emballage de composants réutilisables. Mécanismes d'abstraction et de paramétrisation en OO (dissimulation de l'information, surcharge, généricité, polymorphisme). La librairie de base. Lignes directrices de conception et de programmation.

Ce cours intègre la théorie et la pratique sur des postes informatiques. Il comporte une séance supplémentaire obligatoire de laboratoire.

Préalables: INF7212 Introduction aux systèmes informatiques ; INF7213 Algorithmes et structures discrètes

- OBJECTIFS** Ce cours vise à :
- Présenter la terminologie et les concepts de l'approche orientée objet ;
 - Amorcer une réflexion sur la conception orientée objet ;
 - Utiliser concrètement un langage de programmation orienté objet pour la réalisation de systèmes simples
 - en concevant des classes et
 - en réutilisant des classes existantes ;
 - Apprécier les avantages de l'approche.

Les compétences développées dans le cadre de ce cours rendront l'étudiant, l'étudiante capable de :

- Comprendre une explication mettant en cause la description d'un système orienté objet.
- Expliquer les différentes sections d'un diagramme statique de classe présenté en UML.
- Concevoir un diagramme en UML représentant une hiérarchie d'agrégation/composition et/ou de spécialisation/généralisation de classes dans le processus de résolution d'un problème avec l'approche orientée objet.
- Discriminer les contextes suivants de réutilisation de code : agrégation/composition, spécialisation ou simple dépendance
- Effectuer la trace des instructions impliquées dans un programme Java utilisant des classes.
- Acquérir une certaine "culture générale" des bibliothèques de code reliées au domaine d'application.
- Rechercher et intégrer des classes dans un projet de développement logiciel.
- Élaborer une stratégie de tests. Effectuer la codification en Java.
- Appliquer la stratégie de tests

ÉVALUATION	Description sommaire	Date	Pondération
	TP1	Cours 6	15%
	Examen intra	Cours 8	25%
	TP2	Cours 10	15%
	TP3	Cours 14	20%
	Examen final	Cours 15	25%

CONTENU L'ordre indiqué pour les concepts est séquentiel, mais la mise en pratique de ces concepts pourra se retrouver disséminée à travers les divers travaux pratiques demandés.

Les concepts présentés sont propres à Java et les exemples seront autant que possible inspirés ou tirés de problèmes de biologie moléculaire. Les représentations schématiques se feront en UML (autant que possible).

BLOC I – Conception de classes I

- Généralités sur Java
- Philosophie de la POO
- Conversions de types (transtypages)
- Design et représentation UML
- Les membres : attributs et catégories de méthodes : constructeur simple, accesseur, mutateur

- Variables d'instance finales
 - Organisation en paquetages
 - Utilisation : construction d'un objet, accès à ses données et mutations
 - Tests
 - Encapsulation : les modificateurs de visibilité : public et private
- **BLOC II – Conception de classes II**
- La relation d'agrégation (ou de composition) en tant que réutilisation
 - La référence *this*
 - Constructeur par défaut
 - Surdéfinition de méthodes et de constructeurs
 - Ordre d'initialisation des champs
 - Interface d'une classe
 - Objets constants et classe comme énumération
 - Exemple : structures chaînées itératives et récursives
- **BLOC III – Conceptions de hiérarchies d'héritage**
- La relation d'héritage en tant que réutilisation
 - Les modificateurs de visibilité : *paquetage* et *protected*
 - Redéfinition de méthodes
 - Appels aux constructeurs de base
 - Transtypages
 - Liaison dynamique des méthodes
 - Polymorphisme
 - Généricité
 - La classe *Object* comme superclasse par défaut et ses méthodes
 - Gestion des exceptions
 - Classes et méthodes abstraites
 - Utilité dans le polymorphisme
 - Interfaces
 - Classes et méthodes finales
- **BLOC IV – Étude de bibliothèques de code et autres sujets**
- Classes conteneurs d'objects (Collection et Map)
 - Sérialisation et E/S
 - Classes internes
 - Classes de l'API Swing pour les interfaces graphiques
 - Classes de l'API BioJava pour le traitement de données biologiques

CALENDRIER

Période	Contenu	Lecture et laboratoire
1, 2 et 3	Conception de classes I	
4, 5 et 6	Conception de classes II	
7	Hiérarchies d'héritage I	
-	Semaine de relâche	
8	Examen	
9 et 10	Hiérarchies d'héritage II	
11	Étude de bibliothèques de code et autre sujets	
12, 13 et 14	Étude de bibliothèques de code et autres sujets	
15	Examen	

RÉFÉRENCES

- VO *Notes de cours distribuées par l'enseignant.*
Principalement par le biais du site du cours.
- UO <http://www.bioinfo.uqam.ca/inf7214>
Site du cours, actualisé chaque semaine.
- VR *Un livre de référence portant sur le langage de programmation Java.*
- VR Delannoy – *Programmer en Java* – Eyrolles.
- VR Deitel & Deitel – *Java How to Program* – Prentice Hall.
- VR Eckel – *Thinking in Java, (3rd edition)*, – Éd. Prentice Hall. – <http://www.mindview.net/Books/TIJ/>
Livre gratuit disponible Internet .
- VR Horstmann & Cornell – *Au coeur de Java2, Notions fondamentales, Volume 1* – Campus Press France.
- VR Levenick, James R. – *Java simplifié* – Thomson groupe Modulo, 2006.
- VC Booch, Rumbaugh, Jacobson – *The Unified Modeling Language User Guide (UML)* – Addison Wesley.
- VC Fournier, J.-P. – *Passeport pour l'algorithmique objet* – Thompson Publishing France.
- LC *NetBeans 6 - Environnement de développement intégré* – java.sun.com
- LC *J2SE SDK 1.6 - Compilateur et machine virtuelle Java* – java.sun.com
- LC *BioJava - Bibliothèque de code pour applications bioinformatiques* – www.biojava.org

A : article – C : comptes rendus – L : logiciel – N : notes – R : revue –
S : standard – U : uri – V : volume

C : complémentaire – O : obligatoire – R : recommandé