UCLouvain

linfo2132

2022

Languages and translators

6.00 crédits	30.0 h + 30.0 h	Q2

Enseignants	Sadre Ramin ;
Langue d'enseignement	Anglais > Facilités pour suivre le cours en français
Lieu du cours	Louvain-la-Neuve
Thèmes abordés	• Méthodes d'analyse de langages "context-free", méthodes ascendantes et méthodes descendantes. Applications • Générateurs d'analyseurs lexicaux et syntaxiques • Sémantique statique et grammaires attribuées • Méthode de définition de traduction du code source en code cible et génération du code cible • Machine virtuelle et byte-code (JVM) • Garbage Collection et gestion mémoire • Domain Specific Languages (DSL)
Acquis d'apprentissage	**A la fin de cette unité d'enseignement, l'étudiant est capable de :** Eu égard au référentiel AA du programme « Master ingénieur civil en informatique », ce cours contribue au développement, à l'acquisition et à l'évaluation des acquis d'apprentissage suivants : • INFO1.1-3 • INFO2.2-4 • INFO5.2, INFO5.4, INFO5.5 • INFO6.1, INFO6.4 Eu égard au référentiel AA du programme « Master [120] en sciences informatiques », ce cours contribue au développement, à l'acquisition et à l'évaluation des acquis d'apprentissage suivants : • SINF1.M2 • SINF2.2-4 • SINF5.2, SINF5.4, SINF5.5 • SINF6.1, SINF6.4 Eu égard au référentiel AA du programme « Master [60] en sciences informatiques », ce cours contribue au développement, à l'acquisition et à l'évaluation des acquis d'apprentissage suivants : • 1SINF1.M2 • 1SINF2.2-4 • 1SINF5.2, 1SINF5.4, 1SINF5.5 • 1SINF6.1, 1SINF6.4 Les étudiants ayant suivi avec fruit ce cours seront capables de • expliquer de façon pratique la structure des compilateurs pour des langages algorithmiques • concevoir et implémenter un compilateur pour un langage pratique qui résout un problème à intérêt intrinsèque • montrer l'intérêt des techniques de compilation dans la résolution de problèmes dans d'autres domaines Les étudiants auront développé des compétences méthodologiques et opérationnelles. En particulier, ils ont développé leur capacité à • traiter avec rigueur une problématique en justifiant et validant chaque étape d'un projet pour pouvoir s'appuyer sur celle-ci pour mettre en oeuvre la suivante • expliquer de façon pratique comment un code-source (Java) est finalement traduit en byte-code. • expliquer les mécanismes d'exécution du byte-code par une JVM • expliquer la gestion mémoire lors de l'exécution d'un programme • expliquer le fonctionnement des mécanismes de garbage collection

1

Modes d'évaluation des acquis des étudiants	Session de juin : L'évaluation se compose de deux composantes: Le projet (réalisé en groupe) compte pour 60% de la note finale du cours. Un examen écrit compte pour 40%. Session d'août : Si l'étudiant.e n'a pas réussi le cours lors de la première session (c'est-à-dire qu'il/elle n'a pas obtenu au moins 10/20 pour la note finale), il/elle est autorisée à refaire les composantes (projet ou examen ou les deux) de l'évaluation pour lesquelles il/elle n'a pas obtenu au moins 50% des points respectifs. Il/elle conservera les points de la composante qu'il/elle a réussie (le cas échéant). Les mêmes pondérations que lors de la session de juin sont appliquées pour le calcul de la note finale. Les deux sessions : Le professeur peut demander à un.e étudiant.e de passer un examen oral supplémentaire en complément de l'examen et/ou des activités du projet, dans des cas incluant, mais non limités à, des problèmes techniques, ou des suspicions d'irrégularités.
Méthodes d'enseignement	Le cours consiste d'une série de présentations pré-enregistrées, ainsi que de sessions de consolidation/cours hebdomadaire ou bimensuelles. Il y aura également des séances de travaux pratique afin de préparer les étudiants au projet.Au cours du quadrimestre, les étudiants devront réaliser le projet, qui consiste à étendre un interpreteur/ compilateur pour un language de programmation basique avec de nouveaux paradigmes.
Contenu	Le cours présente la théorie et la pratique de l'implémentation des langages de programmation, et de l'architecture des compilateurs. Nous balayerons les composants standards d'un compilateur, du front-end (analyse lexicale et syntactique) au back-end (émission de code machine, ou interpréteur) en passant par l'analyse statique et les systèmes de types. A terme, les étudiants seront capables de comprendre les tenants et les aboutissants des différentes méthodes d'implémentation de langage en usage aujourd'hui. Au cours du quadrimestre, les étudiants seront amenés à implémenter leur propre langage de programmation.
Ressources en ligne	Teams et/ou Moodle
Bibliographie	Ouvrage(s) recommandé(s) : • Crafting Interpreters, Bob Nystrom (https://craftinginterpreters.com/) • How To Create Pragmatic Lightweight Languages, Federico Tomassetti • Introduction to Compiler Construction in a Java World, Bill Campbell, Swamilyer, Bahar Akbal-Deliba • Modern Compiler Implementation in Jaav, Andrew W. Appel
Autres infos	Préalables : • LINGI1122 : Méthodes rigoureuses de conception de programmes • LSINF1121 : langage de programmation de haut niveau, algorithmique et structures de données • LINGI1101 : Logique et structures discrètes
Faculté ou entité en charge:	INFO

Programmes / formations proposant cette unité d'enseignement (UE)				
Intitulé du programme	Sigle	Crédits	Prérequis	Acquis d'apprentissage
Master [120] : ingénieur civil en informatique	INFO2M	6		🔍
Master [120] en sciences informatiques	SINF2M	6		🔍
Master [60] en sciences informatiques	SINF2M1	6		🔍