II.1102 / II.1202 – Algorithmique et Programmation

INFORMATIONS GENERALES

Titre du module : Algorithmique et Programmation
Identifiant du module : II.1102/II.1202
Responsable du module : Guillaume Lachaud
ECTS : 5
Quantité de travail moyenne par élève : 150h, dont 52h en présentiel
Travail en équipe : oui (projet en groupe)
Mots clés : Algorithmique, Java, Interface Graphique, Jeu

PRESENTATION

Ce module introduit les notions de bases d'algorithmique ainsi que les principes fondamentaux de la programmation orientée objet à l'aide du langage Java. Il permet de se familiariser avec les concepts de programmation d'un langage compilé, ainsi qu'avec l'utilisation de bibliothèques externes, notamment pour les interfaces graphiques. Il demande aussi de développer une capacité de modélisation de programmes via des diagrammes UML, et l'analyse de la complexité algorithmique d'un programme pour produire du code efficace.

OBJECTIFS PEDAGOGIQUES

Lien avec le référentiel de compétences ISEP

Compétences spécialisées

- Résoudre des problèmes scientifiques et techniques pluridisciplinaires sous contraintes dans le domaine des TIC
  - Analyse du problème et prise en compte des contraintes
  - Modélisation du problème et traitement formel
  - Évaluation des solutions
- Concevoir un objet technologique logiciel ou matériel à fonctionnement sûr et normalisé
  - Maîtriser les phases de conception
  - Maîtriser la gestion du processus de réalisation ou de développement
- Agir en mode projet
  - Savoir agir en maître d'ouvrage
  - Savoir agir en maître d'œuvre

Compétences transverses

- Agir en acteur dynamique dans un groupe
- Agir en bon communicant dans un environnement scientifique et technique ouvert à l'international

Plus spécifiquement…

- Concevoir une application testée et documentée en Java, partant de sa modélisation via un diagramme UML, jusqu'à sa livraison (code source ou fichier .jar)
- Être capable de décomposer et modéliser des problèmes liés à l'algorithmique, puis d'analyser les solutions (via la complexité) et de produire une solution optimisée.

Prérequis

Aucun

Contenu/programme

Concepts

- Programmation orientée objet
- Langage compilé (Java)
- Programmation récursive
- Complexité algorithmique
- Design patterns
- Manipulation de fichiers
- Gestion et automatisation de production (outils de build)
- Programmation événementielle et Interface graphique (JavaFX)
- Tests unitaires
- Documentation en programmation
- Diagrammes UML
- Système de contrôle de version (git)
- Structures de données (tableaux, listes, maps, etc.)

Savoir-faire

- Utilisation d'un IDE.
- Modélisation d'un problème sous forme de diagramme UML.
- Utilisation d'un gestionnaire de build (comme Maven).
- Réalisation de test unitaires.
- Analyser la complexité algorithmique d'un problème.
- Utilisation d'un système de contrôle de version (git).
- Savoir lire une documentation.
- Réaliser une interface graphique (JavaFX ou Swing)

Outils utilisés

- IDE recommandé : Intellij IDEA. Alternatives: VS Code, Eclipse, Vim, …
- Gestionnaire de build: maven ou gradle
- Interface graphique: JavaFX. Swing autorisé
- Git pour le versionnage de fichiers.

Mobilisations ultérieures à l'ISEP

- Le langage Java est utilisé dans de nombreux cours d'A2 et d'A3 (technologies web, algorithmique et programmation avancées, génie logiciel, base de données et big data, applications mobiles, …)
- Java s'utilise pour le développement mobile (Android), les bases de données (Oracle), le déploiement d'applications (Quarkus), pour des applications multi-plateformes (JavaFX).
- Plusieurs langages, tels Kotlin, Scala et Clojure, reposent sur Java et la JVM.

MODALITES PEDAGOGIQUES

Méthodes d'apprentissage

Les séances se décomposent en deux types : « cours/td » et « projet ».

- Les séances « cours/td » se divisent en deux : dans la première partie, les notions essentielles sont introduites avec des slides ainsi que des exemples. Dans la deuxième partie, les élèves mettent en pratique ces notions dans la résolution d'exercices.
- Les séances « projet » permettent aux élèves de travailler sur leurs projet (individuel et en groupe). Ces séances peuvent faire l'objet d'une présentation de concepts importants pour la réalisation d'une application, ou bien servir à aider chaque élève et chaque groupe à surmonter les difficultés qu'ils rencontrent.

Modalités d'évaluation

- Le cours ainsi que la capacité à produire du code Java est évalué à mi-semestre avec une évaluation papier.
- Les compétences de programmation sont évalués via un TP noté (par exemple sur Moodle).

- Chaque élève doit réaliser un projet individuel durant les deux premiers tiers du semestre (par exemple un jeu de rôle jouable en console et en interface graphique).
- En parallèle, un projet de groupe (jeu de plateau) est à réaliser par les élèves et fait l'objet d'une soutenance lors de la dernière séance de l'année.
- Un partiel final contenant des questions de cours, de code, d'algorithmique et de modélisation UML complète l'ensemble des notes.

Langue de travail

- Français
- Le code doit être écrit en anglais, afin de se conformer aux exigences de l'industrie. De plus, la plupart des documentations en ligne sont en anglais.

BIBLIOGRAPHIE, WEBOGRAPHIE, AUTRES SOURCES

- Pour les bases de l'algorithmique, « Algorithms » de Sedgewick and Wayne
- Pour les design patterns (programmation orientée objet), "Design Patterns: Elements of Reusable Object-Oriented Software" de Gamma et al.
- Les documentations officielles, telles que celle de Java ou celle de JavaFX.
- Le livre "Learn JavaFX 17" de Sharan et Späth propose un traitement assez exhaustif de la programmation d'interface graphique avec JavaFX.
- "Effective Java" de Joshua Bloch fournit une liste de bonnes pratiques à adopter pour produire du code Java de qualité.