1CC2000 - Algorithmique & Complexité

Responsables : **Fabrice POPINEAU**

Département de rattachement : **DÉPARTEMENT INFORMATIQUE**

Langues d'enseignement : **ANGLAIS , FRANCAIS**

Type de cours : **Cours commun**

Campus où le cours est proposé : **CAMPUS DE PARIS - SACLAY , CAMPUS DE METZ , CAMPUS DE RENNES**

Nombre d'heures d'études élèves (HEE) : **60**

Nombre d'heures présentielles d'enseignement (HPE) : **31**

Année académique : **2024-2025**

Niveau avancé : **non**

Présentation, objectifs généraux du cours :

Ce cours a pour objectif de présenter les méthodes informatiques de résolution de problèmes d'ingénierie. Il se base d'une part sur la représentation de différentes familles de problèmes à l'aide de modèles théoriques, et d'autre part sur leur résolution par des algorithmes séquentiels ou distribués. Nous nous attacherons à déterminer l'existence d'une solution et sa qualité. Nous nous intéresserons à la complexité des problèmes étudiés ainsi que celle des algorithmes développés.

Période(s) du cours (n° de séquence ou hors séquence) :

ST2

Prérequis :

SG1 – Systèmes d'information et programmation (SIP)

Plan détaillé du cours (contenu) :

Cours 1 : Introduction : problème de décision et d'optimisation, solution, algorithme, complexité des algorithmes, représentation de graphe, parcours de graphe non pondéré
Cours 2 : Parcours des graphes acycliques (DAG) et ordonnancement, parcours de graphe pondéré
Cours 3 : Arbre couvrant minimum, algorithmes de Prim, Kruskal
Cours 4 : Flot max, Ford-Fulkerson, applications
Cours 5 : Programmation dynamique
Cours 6 : Complexité des problèmes et réduction polynomiale, NP-complétude
Cours 7 : Méthodes exactes et approchées pour la résolution de problèmes NP : backtracking ; problème du voyageur de commerce (TSP)

Déroulement, organisation du cours :

-7x1h30 de cours
-12x1h30 de TD dont 3 TP et 2 TD-machine
3h d'examen

Organisation de l'évaluation :

Examen écrit

-examen écrit final: 80%

-contrôle continu (rendu du TP en présentiel): 20%

Moyens :

- Equipe enseignante (noms des enseignants des cours magistraux) :
  - Fabrice POPINEAU
  - Arpad RIMMEL
  - Nicolas SABOURET
  - Joanna TOMASIK
  - Benoit VALIRON
  - Marc-Antoine WEISSER
  - Idir AIT SADOUNE
  - Lina YE
- Taille des TD (par défaut 35 élèves) : 30 élèves maximum
- Outils logiciels et nombre de licence nécessaire : environnement de développement Python vu dans le cours SG1 – systèmes d'information et programmation (VSCode, etc)

Acquis d'apprentissage visés dans le cours :

À l'issue de ce module, les élèves seront capables :
- De raisonner en termes algorithmiques pour résoudre des problèmes de la vie réelle (pensée computationnelle ou « computational thinking ») ;
- De connaître des techniques génériques de conception d'algorithmes (force brute, diviser pour régner, etc.) et les appliquer pour résoudre un problème ;
- D'utiliser des méthodes exactes (programmation dynamique, backtracking, etc.) et des méthodes heuristiques (glouton, A*, etc.) pour obtenir des solutions approchées à un problème d'optimisation ;
- D'analyser des algorithmes et d'estimer leur complexité temporelle et spatiales ;
- D'étudier la classe de complexité d'un problème pour choisir les bonnes méthodes de résolution.

Description des compétences acquises à l'issue du cours :

By the end of this course, students will be able to:
- Reason in algorithmic terms to solve real-life problems (computational thinking). Given a real-life problem, students will propose a model, determine the complexity class of the associated computational problem, propose an algorithmic solution to the problem and evaluate the quality of the solution.

In relation to skill C6.1 (Solve a problem numerically)
- Know generic techniques for designing algorithms (brute force, divide and conquer, etc.) and apply them to solve a problem;
- Use exact methods (dynamic programming, etc.) and heuristic methods (greedy, A*, etc.) to obtain approximate solutions to an optimization problem;
- Analyze algorithms and estimate their temporal and spatial complexity;
- Determine the complexity class of a problem to choose the right solutions.

In relation to skill C6.2 (Designing software)
- 3 labs and 2 practical exercises will allow students to implement the algorithms they studied through a Python program.

Bibliographie :

Les transparents seront accessibles en ligne sur la plate-forme de gestion de contenus de l'école.

Les élèves intéressés peuvent consulter les ouvrages de référence suivant :

- Introduction to Algorithms, Third Edition. Par Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein. MIT Press, 2009.
  L'édition précédente (1996) est disponible en français chez Dunod sous le titre « Introduction à l'algorithmique ».
- Algorithm Design. Par Jon Kleinberg et Éva Tardos. Pearson Ed. (Addison-Wesley), 2006. Il n'y a pas d'édition en français mais le livre est disponible en PDF sur Internet.
- Programmation Efficace : Les 128 Algorithmes Qu'Il Faut Avoir Compris et Codés en Python au Cours de sa Vie. Par Christophe Dürr et Jill-Jênn Vie. Ellipse, 2016.