en espace. Aucune formalisation d'un modèle de calcul n'est exigée.

Épreuves de l'option D

1re Épreuve : Mathématiques

Le programme de cette épreuve est constitué des titres I à XI ci-dessus. Les candidats se verront proposer deux sujets au choix, l'un d'algèbre et géométrie, l'autre d'analyse et probabilités.

2e Épreuve : Informatique Fondamentale

Le programme de cette épreuve est constitué des titres Info-1 à Info-5 ci-après.

3e Épreuve : Analyse de système informatique

Le programme de cette épreuve est constitué des titres Info-1 à Info-5 ci-après.

Deux textes décrivant une classe de systèmes informatiques sont proposés au candidat qui doit choisir l'un des deux. La compréhension de ces textes, et leur exploitation dans cette épreuve, requièrent les connaissances en informatique correspondant aux matières enseignées en DEUG MIAS ou dans l'option informatique des classes préparatoires auxquelles s'ajoutent celles du programme.

L'objectif de l'épreuve est d'évaluer la capacité des candidats à mettre en place un processus d'analyse d'un système informatique dans un contexte applicatif. Ce processus s'appuie sur les notions au programme.

Les langages informatiques C, Caml et Java seront disponibles pour cette épreuve et sa préparation. Le rapport du Jury précisera la nature de l'environnement logiciel.

Programme de l'option D

L'ensemble du programme correspond à 250 h de formation (cours et/ou TD et/ou TP) de niveau Licence et première année de Master, à partir des acquis des deux premières années de Licence ou de l'option informatique des classes préparatoires. L'objectif de cette option est de s'assurer que les candidats maîtrisent les fondements essentiels et structurants de la science informatique.

Le programme n'est pas rédigé comme un plan de cours, il décrit les notions que les candidats doivent maîtriser. Chaque partie est accompagnée de quelques références bibliographiques de base.

Info-1 : Algorithmique fondamentale

Cette partie insiste sur les notions de preuve et de complexité des algorithmes. Elle est relativement indépendante de tout langage de programmation, mais le candidat doit être capable de mettre en oeuvre sur machine les structures de données et les algorithmes étudiés.

- Structures de données. Approche abstraite : manipuler les types de données classiques par leurs interfaces et non leurs implantations. Types abstraits : listes, piles, files, arbres, graphes. Structures de dictionnaire de recherche et de file de priorité.
- Complexité et preuve d'algorithmes. Analyse des algorithmes : notations grand-o O(.), Thêta et Omega. Analyse dans le pire cas. *Exemple d'analyse en moyenne : le tri rapide.* Principes de preuve : assertions, pré-post conditions, invariants, terminaison des boucles.
- Algorithmes de tri et de recherche. Méthodes de tri par comparaison (tri-fusion, tri-tas, tri rapide), arbre de décision et borne inférieure du tri par comparaisons. Méthodes de recherche séquentielle et dichotomique. Représentation de dictionnaires de recherche. Arbres binaires de recherche. * Exemples de méthodes d'équilibrage et de hachage et analyse de leur complexité.*
- Algorithmes de graphes et réseaux. Parcours de graphes : parcours en largeur (tri topologique, plus court chemin Dijkstra) ; parcours en profondeur (composantes fortement connexes) ; arbres couvrants de poids minimum (Prim, Kruskal) ; plus courts chemins : fermeture transitive, programmation dynamique (Floyd-Warshall).

Références

Introduction à l'algorithmique, T. H. Cormen, C. E. Leiserson, R. L. Rivest, Dunod.
Algorithms, R. Sedgewick, Addison-Wesley.
Éléments d'algorithmique, D. Beauquier, J. Berstel, Ph. Chrétienne, Masson.
Types de données et algorithmes, C. Froidevaux, M.-C. Gaudel, M. Soria, McGraw-Hill-InterEditions.

Info-2 : Automates et langages

- Automates finis. Langages reconnaissables. Lemme d'itération. Existence de langages non

reconnaissables. Automates émondés, complets. Automates déterministes. Algorithme de déterminisation.
- Propriétés de clôture des langages reconnaissables. Expressions rationnelles. Langages rationnels. Résolution d'équations linéaires gauches. Théorème de Kleene.
- Automate minimal. Résiduel d'un langage par un mot. Algorithme de minimisation. * Utilisation des automates finis. Automates sous-séquentiels (automates déterministes avec sortie) : définitions et exemples d'utilisation (multiplication par une constante, addition de deux entiers, recherche et remplacement de motifs, applications linguistiques, etc.)*
- Automates à pile. Langages algébriques. Existence de langages non algébriques. Grammaires algébriques, simplification des grammaires algébriques, forme normale de Greibach. Équivalence entre automates à pile et grammaires algébriques. Propriétés de clôture des langages algébriques.

Références

Cours et exercices d'informatique, L. Albert, Vuibert.
Théorie des langages et des automates, J.-M. Autebert, Masson.
Éléments d'algorithmique, D. Beauquier, J. Bers tél. et Ph. Chrétienne, Masson.
Introduction to Automata Theory, Languages and Computation, J. Hopcroft et J. Ullman, Addison-Wesley.
Semigroups and combinatorial applications, G. Lallement, Wiley and sons.
Combinatorics on Words, Encycopledia of Mathematics, Lothaire, Addison-Wesley.

Info-3 : Calculabilité, décidabilité et complexité
Les candidats doivent avoir assimilé les aspects centraux de la théorie de la calculabilité et de la complexité. Ils doivent maîtriser l'importance de la notion de calcul et de ses limites intrinsèques.
- Définitions et exemples de fonctions primitives récursives et récursives. Fonction d'Ackerman. Non stabilité des fonctions primitives récursives par passage à l'inverse.
- Définitions des machines de Turing. Équivalence des machines à un et plusieurs rubans. Exemples. Complexité en temps et en espace. Codages des entrées. Equivalence avec les fonctions récursives.
- Universalité. Théorème s-n-m. Théorème de la récursion de Kleene. Définitions et caractérisations des ensembles récursifs, récursivement énumérables.
- Indécidabilité. Théorème de l'arrêt. Théorème de Rice. Exemples.
- Machines de Turing non déterministes. Classes P et NP. NP-complétude. Théorème de Cook. * Exemples de problèmes NP-complets.*

Références

Complexité et décidabilité, J.-M. Autebert, Masson.
Logique mathématique 2. Fonctions récursives, théorème de Gödel, théorie des ensembles, théorie des modèles, R. Cori et D. Lascar, Dunod.
Complexité et décidabilité, P. Dehornoy, Springer.
Mathématiques de l'informatique, P. Dehornoy, Dunod.

Info-4 : Logique et démonstration
- Bases de logique : langages, formules, substitution, règles d'inférence, preuves (système de Hilbert, déduction naturelle, calcul des séquents). Calcul propositionnel, calcul des prédicats du premier ordre.
- Sémantique : structure, vérité d'une formule, notion de cohérence et de complétude, interprétation de Herbrand, théorème de complétude du calcul des prédicats du premier ordre, théorème de compacité, théorème de Lowenheim-Skolem.
Exemples de théories : égalité, arithmétique de Peano, théorie des ensembles. Exemples de théories décidables, indécidables.
- Démonstration automatique. Calcul propositionnel, unification et filtrage du premier ordre. Preuves équationnelles : *réécriture, confluence, confluence locale, terminaison (faible et forte), paires critiques, lemme de Newman, complétion de Knuth-Bendix.* Preuves au premier ordre : résolution, programmation logique, méthode des tableaux.

Références

Logique, réduction, résolution, R. Lallement, Masson.

Introduction à la logique : théorie de la démonstration, R. David, K. Nour et C. Raffalli, Dunod.
Info-5 : Programmation, langages, compilation
Il est attendu des candidats qu'ils maîtrisent les bases nécessaires pour l'enseignement raisonné de la programmation. La maîtrise pointue de telle ou telle technique de compilation n'est pas exigée. En revanche, il est attendu des candidats une culture générale sur le domaine. Le programme n'impose aucun langage de programmation particulier, mais les candidats sont supposés maîtriser au moins un langage et son environnement de programmation parmi CAML, Java ou C.
- Principes de la programmation impérative, fonctionnelle, logique et orientée objet. Langages typés ou non typés. Mise en œuvre par compilation ou interprétation. *Notions d'environnement et de liaison, notions de portée et de durée de vie.*
- Principes de sémantique. Approches opérationnelle, dénotationnelle (directe), axiomatique (logique de Hoare). * Équivalence de programmes selon ces sémantiques, notion de compositionnalité. *
- Analyse lexicale et syntaxique. Grammaires LL(1).* Aperçu sur les grammaires LALR. Utilisation d'outils de génération automatique de type lex et yacc.*
- Analyse statique et application. Typage. Analyse de flot de données. *Exemples de transformations simples de programmes au niveau source.*
- Génération de code et optimisation. Compilation d'un langage impératif simple pour un modèle de machine abstraite à pile.*Architecture à registres et optimisation.*
Références
Compilateurs : principes, techniques et outils, A. V - Aho, R. Sethi et J. D. Ullman, Masson.
Programming language pragmatics, M. L. Scott, Morgan Kaufmann.

Mécanique

Le programme de la session 2005, publié au B.O. spécial n° 7 du 1er juillet 2004, est reconduit pour la session 2006.

Musique

Dissertation

La virtuosité et l'invention musicale (statuts, modalités, gestes, techniques, écritures)
La virtuosité, qui dans son acception la plus commune désigne l'adresse d'un exécutant, sa vélocité, son brio technique, peut aussi bien se référer au talent et à l'habileté dans l'élaboration de certaines compositions particulièrement remarquables. En ce deuxième sens, la virtuosité se développe dans l'invention personnelle dont fait preuve l'artiste sur la base d'un savoir et de techniques communs. Elle distingue alors une initiative individuelle éminemment créatrice des habitudes compositionnelles ou interprétatives. Dans les deux acceptions, le statut de la virtuosité correspond à un besoin de se singulariser et se réfère à une notion d'excellence.
Les gestes constitutifs au fondement de la création autant que de l'interprétation virtuoses concernent aussi bien leurs expressions corporelles que leurs élaborations intellectuelles. Les modalités et les circonstances selon lesquelles virtuosité et invention manifestent leur présence, les techniques et les genres dans lesquels elles se développent, les écritures, vocales et instrumentales, et jusqu'aux nouvelles technologies, qu'elles mettent en œuvre seront étudiés au sein des musiques de traditions orales ou semi-orales (folklores européens et musiques extra-européennes, jazz et musiques populaires actuelles, musiques médiévales et de la Renaissance, etc...) aussi bien que dans la musique écrite occidentale - cette diversité de répertoires donnant tout son sens à la question ci-dessus libellée. L'étude inclura une mise en relation des aspects de cette problématique musicale avec les autres modes d'expression artistique.
Les questions 1 et 2 publiées au B.O. spécial n° 5 du 20 mai 2004 sont inchangées.

Philosophie

Écrit

2ème épreuve. Composition de philosophie se