
Liste de compétences à maîtriser en fin de MP2I

Général

- Connaître l'orthographe des mots liés à des notions essentielles du cours
- Accorder en genre et en nombre les adjectifs, les participes, les pronoms...
- Ne pas confondre son brouillon et la copie que l'on rend

Outils mathématiques

- Formaliser un problème concret comme un problème d'optimisation ou de décision
- Maîtriser les objets mathématiques usuels : ensemble, multi-ensemble, suite...
- Maîtriser les notions de mot, sous-mot, préfixe, suffixe, alphabet, concaténation, mot vide
- Prouver qu'une relation binaire est réflexive, symétrique, transitive, d'ordre, d'équivalence...
- Maîtriser les notations O , Ω et Θ
- Comparer des comportements asymptotiques

Étude des algorithmes

- Simuler le comportement d'un algorithme itératif
- Prouver la correction d'un algorithme itératif grâce à des invariants de boucle
- Prouver la terminaison d'un algorithme itératif grâce à des variants de boucle
- Prouver la correction et la terminaison d'un algorithme récursif par induction sur ses entrées
- Dénombrer les appels récursifs lors d'un appel d'un algorithme récursif
- Établir la complexité pire cas d'un algorithme

Logique propositionnelle

- Mettre une formule de la logique propositionnelle sous forme normale
- Montrer que des formules sont équivalentes
- Montrer des petits résultats théoriques à partir des définitions du cours de logique

Structures de données et algorithmique

- Connaître les structures de données abstraites : pile, file, liste, tas, ensemble, dictionnaire
- Connaître les définitions et le vocabulaire des graphes
- Connaître les différentes représentations des graphes en machine
- Définir un algorithme "glouton"
- Déterminer si un algorithme "glouton" est optimal (preuve d'optimalité ou contre-exemple)
- Définir un algorithme de type "diviser pour régner"
- Établir la complexité d'un algorithme "diviser pour régner"
- Définir un algorithme de programmation dynamique

- Établir la complexité d'un algorithme de programmation dynamique
- Prouver que l'optimum d'un pb. d'optim. vérifie certaines propriétés par argument d'échange
- Définir un algorithme de parcours d'un graphe (adapté à la résolution d'un problème particulier)

Programmation générale

- Donner une description brève et précise de chaque fonction
- Commenter de manière utile son code
- Produire un jeu de tests pour tester une fonction
- Donner le graphe de flot de contrôle d'un programme

Programmation en C

- Définir et utiliser des `struct` en C
- Simuler l'évolution de la pile lors de l'exécution d'un programme C
- Distinguer le passage par valeur du passage par référence
- Savoir créer dans une fonction des objets persistants, existant hors de cette fonction
- Allouer l'espace mémoire et le libérer
- Maîtriser la notion de pointeur : les types associés, l'accès et la modification
- Manipuler des tableaux, à une ou plusieurs dimensions
- Être à l'aise avec des structures de données composées de cellules : listes chaînée, arbres...

Programmation en Ocaml

- Déterminer le type d'une expression en Ocaml
- Maîtriser les types paramétrés en Ocaml
- Produire l'arbre de syntaxe d'une expression Ocaml
- Définir des nouveaux types en Ocaml : types somme, type produits, type récursifs
- Maîtriser la manipulation des listes en Ocaml, notamment les filtrages
- Déterminer si une fonction récursive est récursive terminale
- Proposer une fonction récursive terminale pour calculer une suite récurrente
- Définir des fonctions mutuellement récursives, notamment pour traiter les arbres généraux

Bases de données

- Proposer un modèle entité associations à partir d'une situation concrète décrite en français
- Interroger une base de données à travers des requêtes en SQL