MP

CONCOURS COMMUN INP

RAPPORT DE L'ÉPREUVE ÉCRITE D'INFORMATIQUE

1/ PRÉSENTATION DU SUJET

L'épreuve est constituée de trois parties indépendantes.

La première porte sur l'étude des partitions non croisées. L'objectif est de vérifier l'assimilation des notions de base de programmation (boucles, instructions conditionnelles, etc.) et une certaine maîtrise du langage Python. Elle couvre une partie du programme de « l'Informatique Pour Tous ».

La deuxième s'intéresse au problème de satisfiabilité des formules de Horn. Le thème de ce problème correspond donc au chapitre Logique.

La dernière partie consiste à caractériser l'ensemble de mots donnant le même arbre binaire de recherche après insertion dans l'arbre vide. Elle permet d'aborder différentes notions (Récursivité, Listes, Arbres).

2/ REMARQUES GÉNÉRALES

Le sujet semble avoir été d'une longueur et d'un niveau de difficulté parfaitement adapté, permettant une bonne sélection des candidats tant sur les aspects preuve que programmation impérative et fonctionnelle.

La moyenne de l'épreuve est de 10,95 et l'écart type est de 3,61. Le sujet a donc bien permis de classer les différents candidats.

De façon générale, les erreurs proviennent régulièrement des points suivants :

- lecture un peu trop rapide et non complète de certaines questions (exemple : on demande une fonction récursive et la fonction écrite ne l'est pas) ;

- un manque de rigueur et de précision dans la rédaction de certaines preuves (exemple : conclusion manquante, hypothèse de récurrence non précisée, numéro d'une question précédente non mentionnée) ;

- une confusion de syntaxe entre le Python et le OCaml (exemple : entre le && en OCaml et le and en Python).

3/ REMARQUES SPÉCIFIQUES

PARTIE I

La partie est globalement bien traitée et ne comporte pas de difficulté majeure.

Q1, **Q2**, **Q3**, **Q4** : pas de problème particulier.

Q5 : La syntaxe « For L in P » est peu utilisée.

Q6 : Très peu d'étudiants ont pensé à utiliser « A ».

Q7 : Les preuves sont souvent incomplètes.

Q8 : Quelques oublis de justifications élémentaires (souvent le caractère disjoint, plus rarement produit cartésien).

Q9 : Beaucoup d'erreurs sur la gestion des indices de liste Python, des fonctions récursives inefficaces au lieu de la programmation dynamique (non sanctionné).

PARTIE II

Le point qui a pu poser problème est l'algorithme de propagation unitaire.

Q10 : Des rédactions parfois trop lourdes (exemple : utilisation de tables de vérité).

Q11, **Q12** : Pas de problème particulier.

Q13, **Q14** : Les preuves données pouvaient être longues, tout en manquant de rigueur et de précision.

Q15 : Aucun problème.

Q16, **Q17**, **Q18** : Plutôt bien traitées.

PARTIE III

Q19, **Q20**, **Q21**, **Q22**, **Q23** : pas de problème particulier.

Q24 : Beaucoup de récurrences mal rédigées.

Q25 : La réflexivité est souvent omise ; pas de problème pour les autres axiomes.

Q26 : Les démonstrations ont souvent été incomplètes (exemple : des cas non traités).

Q27 : Pas de problème.

Q28 : (a) La question était simple mais certaines démonstrations manquaient de précision.
(b) Pas de problème.
(c) Pas toujours traitée.
(d) et (e) peu abordées et les justifications apportées étaient souvent floues.

Q29 : Bien traitée en général quand on y arrive.

Q30 : Pas de problème.

Q31 : Le mélange de deux mots n'a pas toujours été compris (exemple : des mots obtenus pas de la bonne longueur, un ensemble obtenu de mauvais cardinal).

Q32 : Plutôt bien traitée. Quelques erreurs de signatures (le constructeur « :: » est parfois maltraité).

Q33, **Q34**, **Q35** : Ces questions de programmation en OCaml ont plutôt été bien traitées lorsqu'elles ont été abordées.