# Composition d'Informatique (4 heures), Filière MP

**Rapport de MM. Jean-Christophe Filliâtre et Jean-Pierre Tillich, correcteurs.**

## L'épreuve

Il s'agissait dans ce problème d'étudier une procédure de semi-décision pour vérifier la positivité d'un polynôme à coefficients entiers sur l'intervalle $[0, 1]$, fondée sur les polynômes de Bernstein. L'épreuve contenait l'intégralité des éléments nécessaires à la réalisation de cette procédure, formant ainsi un ensemble cohérent. La **partie I** introduisait la structure de grands entiers, représentés par des listes de chiffres dans une base fixée (mais non spécifiée). Les candidats devaient programmer les opérations élémentaires d'addition, de soustraction, de comparaison, de multiplication par 2 et de division euclidienne par 2. La **partie II** introduisait la notion de nombres dyadiques, sur la base des grands entiers de la partie précédente, et ne consistait qu'à programmer les trois opérations d'addition, de soustraction et de division par 2. La **partie III** introduisait une structure de données pour des listes manipulées par les deux bouts, c'est-à-dire où les opérations d'insertion et de suppression peuvent être réalisées aux deux extrémités. La **partie IV** réunissait enfin les structures de données des parties précédentes, en représentant un polynôme comme une liste de nombres dyadiques, ces nombres étant vus comme des coefficients dans la base des polynômes de Bernstein. Sur la base de cette représentation, un algorithme de raffinement était fourni et les candidats devaient en déduire une méthode de semi-décision sur le principe « diviser pour régner » et montrer son incomplétude.

## Remarques générales

Les notes des 753 candidats français se répartissent selon le tableau suivant, avec une moyenne de 10,1 et un écart-type de 4,0. Le nombre de notes éliminatoires, c'est-à-dire inférieures à 2, est de 16.

| | | |
|---|---|---|
| $0 \leqslant \mathrm{N} < 4$ | 61 | 8,1 % |
| $4 \leqslant \mathrm{N} < 8$ | 157 | 20,8 % |
| $8 \leqslant \mathrm{N} < 12$ | 277 | 36,8 % |
| $12 \leqslant \mathrm{N} < 16$ | 201 | 26,7 % |
| $16 \leqslant \mathrm{N} \leqslant 20$ | 57 | 7,6 % |
| Total | 753 | 100 % |
| Nombre de copies : 753 | | |
| Note moyenne : 10,13 | | |
| Écart-type : 4,04 | | |

Parmi les candidats français, 8,4 % ont choisi d'écrire leur programme en langage Pascal. La moyenne des copies Pascal est de 7,9, soit plus de deux points en dessous de la moyenne générale. Même si de bonnes copies ont été écrites en Pascal, l'impression générale est que la verbosité du langage Pascal constitue un handicap conséquent dans ce type d'épreuve contenant un grand nombre de questions de programmation.

Presque toutes les fonctions demandées pouvaient être écrites en moins de 10 lignes. Les candidats doivent être conscients du fait qu'une réponse longue doit être expliquée en détail et que très souvent un programme très long contient un grand nombre d'erreurs.

Pour obtenir la note maximale, il était nécessaire de traiter le problème en entier.

**Commentaire détaillé**

Pour chaque question, sont indiqués entre crochets la moyenne de la question sur 20 ainsi que le pourcentage de candidats ayant obtenu la note 0 (que la question ait été traitée ou non). De nombreuses questions étaient notées de manière binaire ou ternaire, sur la base de 0/100% ou 0/60/100%.

## Partie I

**Question 1 [1,5% - 93%].** Cette question élémentaire avait pour but de factoriser une partie du code apparaissant de manière naturelle dans les questions suivantes aux endroits où il s'agissait de gérer la représentation d'entiers nuls en respectant l'invariant de l'énoncé. De manière étonnante, cette question n'a pas été comprise par l'immense majorité des candidats qui ommettent de renvoyer une liste vide dans le cas où $c = 0$ et où $n$ est la liste vide.

**Question 2 [12,1% - 26%].** Une partie des candidats n'a pas suivi l'indication de l'énoncé, consistant à rajouter un argument représentant la retenue, ce qui les a conduits à des solutions quadratiques. Pour obtenir la totalité des points, il fallait écrire une solution efficace de complexité linéaire.

**Question 3 [10,8% - 46%].** Vu la simplicité de cette question, seuls les candidats ayant proposé une solution correcte et de complexité linéaire se sont vus attribuer des points.

**Question 4 [6,2% - 55%].** Cette question s'apparentait beaucoup à la question 2, mais comportait une difficulté supplémentaire, qui était d'assurer l'invariant dans le cas où la soustraction produit des chiffres de poids fort égaux à 0.

**Question 5 [6,5% - 58%].** Cette question, un peu plus délicate, comportait deux difficultés : d'une part le respect de l'invariant dans le cas où $n = 1$ ; et d'autre part la gestion de la propagation du reste, plus subtile quedans le cas de l'addition et de la soustraction.

**Question 6 [19,1% - 4%].** Cette question a été correctement traitée dans l'immense majorité des cas.

**Question 7 [17,2% - 8%].** Cette question ne posait pas de difficulté majeure mais de nombreux candidats ont proposé une solution inutilement longue.

**Question 8 [12,6% - 32%].** De nombreux candidats ont cherché à écrire directement une fonction de multiplication par deux des entiers naturels, alors qu'il suffisait de réutiliser la fonction d'addition, sans perte d'efficacité. Par ailleurs, certains candidats ont proposé une solution très inefficace en effectuant plusieurs fois les mêmes appels récursifs.

**Question 9 [13,9% - 23%].** Cette question a été relativement bien traitée, mais nécessitait un peu de soin dans la gestion du signe et du cas d'arrêt.

## Partie II

Un nombre assez significatif de candidats ne semble pas avoir compris la notion de nombre dyadique, en particulier le fait que l'exposant $b$ pouvait être négatif.

**Question 10 [16,9% - 16%].** Les mauvaises réponses sont dues principalement à une mauvaise compréhension de la notion de nombre dyadique.

**Question 11 [10,9% - 33%].** Cette question comprenait plusieurs difficultés :

- traiter correctement le cas où les exposants des deux nombres dyadiques diffèrent, en ramenant le plus grand au niveau du plus petit par multiplication de la mantisse par une puissance de 2 ;
- ne pas oublier la possibilité d'exposants négatifs, et garantir donc que la fonction `mul_puiss2_z` est bien appelée avec un exposant positif ;
- ne pas oublier de respecter l'invariant sur les nombres dyadiques qui exige que la mantisse soit nulle ou impaire.

**Question 12 [15,3% - 19%].** Il s'agissait d'une question très facile dès lors que l'on réutilisait la fonction précédente.

## Partie III

**Question 13 [19,4% - 3%].** Cette question a été correctement traitée par la quasi-totalité des candidats.

**Question 14 [13,8% - 28%].** Cette question réclamait de comprendre deux choses : d'une part que la liste de gauche pouvait être vide ; et d'autre part que, dans ce cas-là, la liste de droite ne contient qu'un seul élément.

**Question 15 [13,9% - 31%].** Il s'agissait d'une question très simple consistant uniquement à inverser le rôle des listes gauche et droite. Certains candidats ont, à tort, inversé le contenu de ces listes.

**Question 16 [10,2% - 37%].** Cette question comptait parmi les questions de programmation les plus longues à rédiger. De nombreuses copies ont perdu ici des points sur des détails, en particulier en oubliant d'inverser une liste ou, au contraire, en en inversant abusivement une autre.

**Question 17 [17,1% - 14%].** Cette question était très facile ; le seul écueil potentiel était l'oubli du respect de l'invariant.

**Question 18 [10,9% - 43%].** Cette question contenait les difficultés cumulées des questions 14 et 17, ce qui explique qu'elle a été relativement mal traitée.

**Question 19 [2,2% - 86%].** Cette question, relativement difficile, nécessitait d'identifier précisément les instants où le rééquilibrage de la paire de listesest effectué. Le choix $c = 3$ de l'énoncé était censé simplifier l'expression de ces instants. Malheureusement, peu de candidats ont obtenu cette expression.

## Partie IV

**Question 20 [11,5% - 20%].** Cette question contenait deux parties : une consistant à montrer par récurrence que les $B_i^k$ sont positifs sur $[0, 1]$, relativement bien traitée, et une autre consistant en une application numérique, entachée de nombreuses erreurs de calcul.

**Question 21 [7,7% - 61%].** Cette question avait pour objectif de donner l'idée de l'algorithme de la question 22. Il semble que beaucoup de candidats n'ont pas compris ce qui était attendu.

**Question 22 [4,9% - 70%].** Cette question consistait à écrire la procédure de semi-décision qui était l'objectif de tout le problème. L'algorithme, de type « diviser pour régner », était suggéré par l'énoncé. Les candidats ayant compris la question précédente ont souvent pris la bonne direction. Néanmoins, la réalisation a souvent souffert de nombreuses incorrections. D'une manière plus anecdotique, les correcteurs ont été surpris de voir des candidats tenter d'explorer exhaustivement un arbre binaire de hauteur 1000.

**Question 23 [2,3% - 85%].** Cette question pouvait être traitée à l'aide de preuves par récurrence sur la taille des polynômes. En particulier, celle visant à montrer la linéarité de la fonction `intègre` était délicate à rédiger. Une autre approche consistait à se ramener à l'interprétation par une somme de polynômes de Bernstein, mais il fallait alors établir que les polynômes de Bernstein forment un système libre.

**Question 24 [2,3% - 80%].** Cette question était relativement longue à rédiger. Il fallait montrer plusieurs points :

- d'une part que $p$ est positif sur $[0, 1]$ ;
- d'autre part que `raffine_g` $p$ contient au moins un coefficient négatif, ce qui garantit que `test_pos` ne s'arrête pas au premier niveau de récursion ;
- et enfin de montrer par récurrence que chaque niveau de récursion contient au moins un polynôme ayant au moins un coefficient négatif.