Composition d'Informatique (2 heures), Filières MP et PC

Rapport de MM. Olivier LY et Dominique ROSSIN, correcteurs.

Les statistiques et le bilan donnés ci-après concernent uniquement les candidats admissibles.

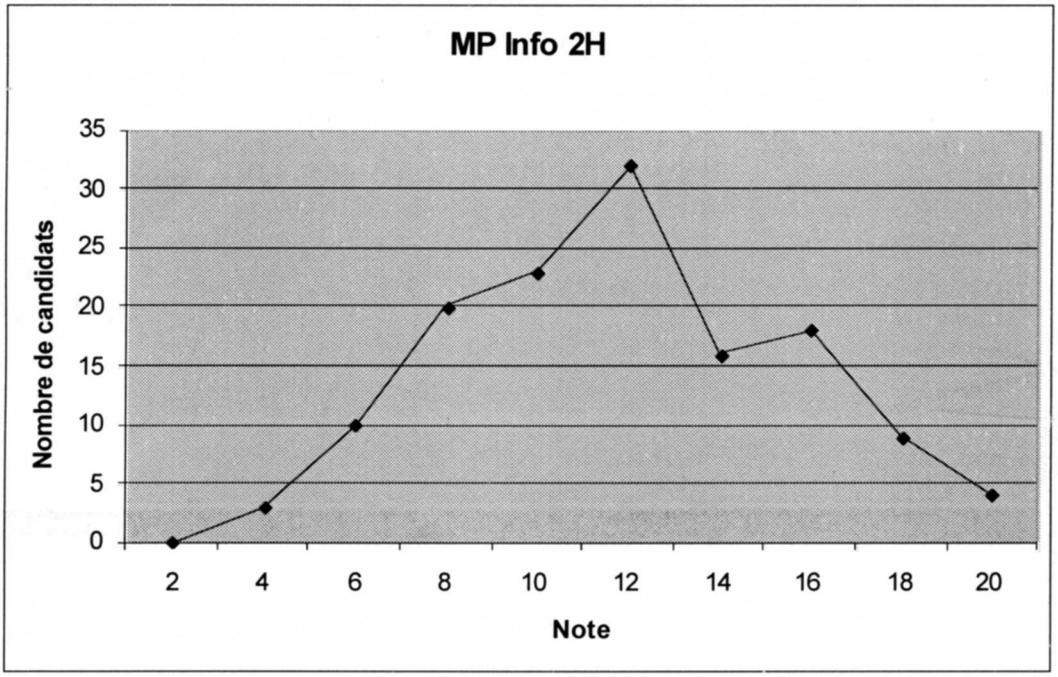
I. Bilan

La répartition des notes pour les 135 candidats français MP admissibles est la suivante :

Français PC

$0 \leqslant N < 4$	3	2,2%
$4 \leqslant N < 8$	30	22,2%
$8 \leqslant N < 12$	55	40,7%
$12 \leqslant N < 16$	34	25,2%
$16 \leqslant N \leqslant 20$	13	9,6%
Total	135	100,0%
Nombre de copies : 135		
Note moyenne : 10,73		
Écart-type : 3,75		

Une analyse plus fine des notes donne la distribution suivante



Le langage de programmation utilisé est majoritairement Maple. Néanmoins, certains candidats ont choisi Java, Mathematica ou encore Pascal comme l'indique le tableau suivant :

Pascal	Mathematical	Maple	Java
2	1	101	31

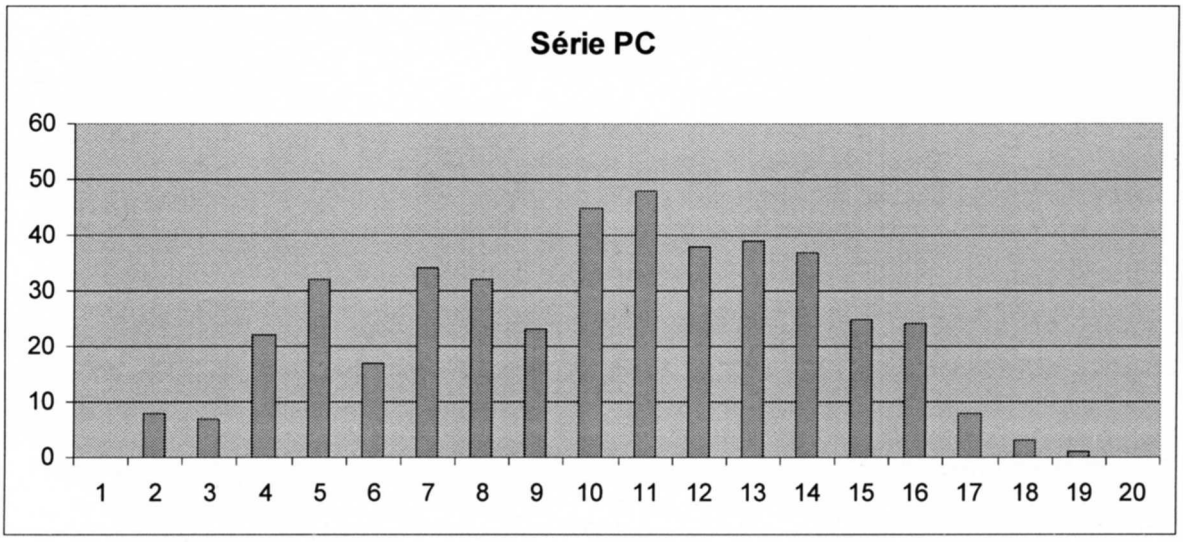
Remarquons que cette année un nombre assez important de candidats ont choisi de réaliser leur épreuve dans le langage Java.

Pour ce qui concerne la filière PC, la répartition des 446 candidats français admissibles est la suivante :

Français PC

$0 \leqslant N < 4$	29	6,5%
$4 \leqslant N < 8$	118	26,5%
$8 \leqslant N < 12$	148	33,2%
$12 \leqslant N < 16$	135	30,3%
$16 \leqslant N \leqslant 20$	16	3,6%
Total	446	100,0%
Nombre de copies : 446		
Note moyenne : 9,82		
Écart-type : 3,79		

La distribution précise est la suivante :



Le langage de programmation utilisé est très majoritairement Maple. Bien que certains candidats comme dans la filière MP ont choisi Java ou encore Mathematica.

II. Commentaires

De nombreuses questions nécessitaient l'écriture d'un programme. Le critère principal d'évaluation de ces questions est la conformité du programme aux spécifications imposées par l'énoncé, que ce soit du point de vue de la forme que du fond (algorithmes employés, complexité en temps et en espace).

Un problème que se posent fréquemment les candidats concerne l'impossibilité dans certains langages de programmation de respecter des aspects de l'énoncé comme la numérotation des indices des tableaux qui commence à 0. Une bonne solution employée par la majorité des candidats consiste simplement à spécifier au début de sa copie qu'il a connaissance de ce décalage d'indice dû au langage utilisé et qu'en conséquence il considèrera que les indices des tableaux commencent à 1.

De même, certains langages de programmation ne permettent pas de modifier certains paramètres d'entrée. Bien entendu, cela a été pris en compte lors de la correction et l'intégralité des points a été accordée si les variables avaient la bonne valeur en fin de fonction.

III. Commentaires détaillés

Les résultats par question ainsi que les coefficients respectifs des questions sont donnés ci-dessous :

Question	1	2	3	4	5	6	7	8	9	10	9
Barème	1	1	1	1	2	2	2	1	1	2	2
Réussite	37%	63%	64%	55%	16%	17%	35%	44%	44%	7%	18%
Moyenne/10	6,78	7,84	8,04	7,93	4,64	5,81	5,85	6,83	5,71	2,7	2,29
Zéro	0,7%	12%	6%	9%	9%	8%	17%	13%	29%	56%	72

Le sujet traitait de problème de compression de données et de manière pratique ne nécessitait que de savoir effectuer des parcours simples de tableau à 1 dimension.

Question 1 :

Certains candidats ont remarqué que dans cette question on pouvait limiter les indices de i entre 0 et 255 comme l'énoncé le suggérait dans le préambule au lieu de l'intervalle 0..n. Cette remarque a bien sûr donné lieu à un bonus. De manière étonnante, certains candidats ont échoué à cette question.

Question 2 :

Cette question ne comportait qu'une difficulté, celle de lire l'énoncé et 12% des candidats ne l'ont pas fait.

Question 3 :

Cette question demandait de savoir repérer les plages contiguës de chiffres identiques. Peu de grosses erreurs sur cette question mais des petits problèmes au début ou à la fin du texte ou alors des codes trop alambiqués.

Question 4 :

Dans cette question, trop de candidats ne se sont pas rendu compte qu'il pouvait être intéressant de faire deux passes sur le tableau, la première en appelant la fonction de la question précédente de manière à connaître la taille du tableau à créer, et une seconde pour le remplir.

Question 5 :

Peu de candidats ont réussi cette question de manière optimale. En effet, beaucoup calculent explicitement les rotations, ce qui demande d'allouer la place des tableaux puis de les parcourir. Une autre solution rencontrée a été de doubler la taille du tableau de départ et de recopier le texte t deux fois dans ce tableau ce qui évite de gérer les problèmes de modulo. La solution attendue était évidement de parcourir grâce à deux marqueurs le tableau et de comparer à chaque étape les valeurs en faisant attention au fait que dès qu'un marqueur sort du tableau, on doit le réinitialiser à 0.

Question 6 :

Cette question était assez difficile car nécessitait de comprendre la fonction triRotations qui était décrite. Il fallait de plus penser à aller chercher le dernier caractère de chacune des rotations. Peu de candidats ont pensé à réaliser l'ensemble des opérations.

Question 7 :

Cette question était liée à la précédente. La difficulté était de bien comprendre la phrase qui expliquait que triRotations effectuait (n ln n) appels à la fonction de comparaison et il fallait donc multiplier ces complexités.

Question 8 :

La question était très similaire à la première, on pouvait donc y faire appel.

Question 9 :

La question était assez simple. Il suffisait de remettre les caractères dans l'ordre alphabétique. Il suffisait donc d'utiliser la question précédente pour connaître la fréquence d'apparition de chaque lettre.

Question 10 :

Sans aucun doute la question la plus difficile de l'énoncé. Elle nécessitait la compréhension de toutes les questions précédentes.

Question 11 :

Cette question demandait l'écriture de la fonction finale de décodage. Elle était réalisable même sans avoir réussi la question précédente ce que peu de candidats ont remarqué. Les problèmes rencontrés sont de diverses natures : création du tableau, recherche du premier caractère...