Composition d'Informatique (2 heures), Filière PC
(XEC)

1. Bilan général

À titre de rappel, cette épreuve n'est corrigée que pour les candidats admissibles. Le présent rapport ne concerne que la filière PC.

Cette année, le nombre total de candidats admissibles dans cette filière est 551. La note moyenne est 10,90 avec un écart-type de 2,86. Les tableaux ci-dessous donnent la répartition détaillée des notes par série, ainsi que la synthèse calculée sur l'ensemble des copies corrigées. La note minimale est 0,80 et la note maximale 18,40.

X - ENS

	Série 1		Série 2		Série 3		Série 4		**Synthèse**	
$0 \leq N < 4$	7	5%	6	4%	9	6%	4	2%	**26**	**4%**
$4 \leq N < 8$	20	14%	18	13%	14	9%	25	18%	**77**	**13%**
$8 \leq N < 12$	61	43%	43	33%	55	38%	50	36%	**209**	**37%**
$12 \leq N < 16$	46	32%	55	42%	62	43%	53	38%	**216**	**39%**
$20 \leq N \leq 20$	6	4%	8	6%	4	2%	5	3%	**23**	**4%**
Total	140	100,0%	130	100,0%	144	100,0%	137	100,0%	**551**	**100,0%**

	Série 1	Série 2	Série 3	Série 4	**Synthèse**
Nombre de copies	140	130	144	137	**551**
Note minimale	2,20	2,20	0,80	1,30	**0,80**
Note maximale	17,30	18,40	17,70	17,50	**18,40**
Note moyenne	10,67	11,08	10,98	10,88	**10,90**
Ecart-type	2,69	3,10	2,71	2,94	**2,86**

Le langage de programmation PYTHON a été utilisé pour la totalité des copies.

Les notes se répartissent selon le tableau suivant, avec une moyenne de 10,98 et un écart-type de 3,53 (pour les candidats français de l'École polytechnique) :

$0 \leqslant$ N $<$ 4	23	4,41 %
$4 \leqslant$ N $<$ 8	74	14,20 %
$8 \leqslant$ N $<$ 12	191	36,66 %
$12 \leqslant$ N $<$ 16	210	40,31 %
$16 \leqslant$ N $\leqslant$ 20	23	4,41 %
Total	521	100 %
Nombre de copies : 521		
Note moyenne 10,98		
Écart-type : 3,53		

2. Commentaires

L'épreuve de cette année portait sur le calcul de l'enveloppe convexe d'un nuage de points dans le plan, un problème classique de géométrie algorithmique. Les premières questions introduisaient un ordre sur les points du nuage. Dans la suite, on étudiait deux algorithmes de calcul de l'enveloppe utilisant cet ordre : le premier dit « du paquet cadeau » d'une complexité $O(n \cdot m)$ où n est le nombre de points du nuage et m le nombre de points de l'enveloppe convexe ; et le second dit « par balayage » d'une complexité $O(n \cdot \log(n))$.

D'une manière générale, les aspects suivants sont pris en compte (par ordre d'importance) dans la correction :

1. la correction de l'*algorithme* proposé vis-à-vis de la spécification fournie ;
2. l'efficacité de la solution ;
3. la lisibilité du code (on favorise une solution simple devant une solution compliquée équivalente, on apprécie le code bien indenté et structuré,...) ;
4. la correction des détails d'*implémentation* ;
5. les explications accompagnant le code sous la forme de commentaires.

3. Commentaires détaillés

Pour chaque question, un tableau récapitule les taux de réussite avec les conventions suivantes :

- 0 signifie qu'aucun point n'a été donné pour la question (question non traitée ou totalement fausse) ;
- $< 0,5$ signifie que moins de la moitié des points ont été donnés ;
- 0,5 signifie que plus de la moitié des points ont été donnés ;
- 1 signifie que la totalité des points ont été donnés.

Question 1 :

	0		<0,5		≥0,5		1		Total	
Question 1	33	5,9%	203	36,8%	91	16,5%	224	40,6%	551	100,0%

On demandait d'écrire une fonction de recherche du point d'ordonnée minimale (en choisissant le point d'abscisse minimale en cas de non unicité de la réponse).

C'était une question facile, très proche de l'algorithme classique de recherche d'un minimum dans une séquence non vide. La seule difficulté était d'implémenter précisément la spécification demandée.

- Certaines réponses renvoyaient l'ordonnée minimale plutôt que l'indice du point d'ordonnée minimale.
- Une réponse compliquée à une question si classique est assez déroutante pour les correcteurs.

– Certaines réponses partaient du présupposé que le tableau de points était trié par abscisses croissantes, ce qui n'était pas le cas. Une telle réponse accompagnée d'un commentaire explicitant cette hypothèse - même si elle est invalide - était préférée à l'absence de commentaire.
– Initialiser l'algorithme avec la valeur 0 pour l'abscisse minimale courante était incorrect : rien n'indiquait que les ordonnées étaient positives.

Question 2 :

	0		<0,5		≥0,5		1		Total	
Question 2	2	0,3%	85	15,4%	40	7,2%	424	76,9%	551	100,0%

Cette question était une application immédiate de la règle de calcul de l'orientation.
– Un minimum de justification était attendue.
– On acceptait une justification graphique ou calculatoire.

Question 3 :

	0		<0,5		≥0,5		1		Total	
Question 3	19	3,4%	18	3,2%	66	11,9%	448	81,3%	551	100,0%

On demandait aux candidats de traduire la règle de calcul de l'orientation sous la forme d'un programme informatique.
– Bien qu'une façon simple et efficace de calculer l'orientation ait été explicitée dans le sujet, certaines réponses ne la suivaient pas. En effet, on a vu des calculs de coefficients directeurs de droites et des analyses de cas sur la position des points vis-à-vis de ces droites : c'était trop compliqué et souvent incorrect.
– Certaines réponses calculaient le signe de x en renvoyant $\frac{x}{|x|}$. On rappelle que ceci n'est défini que si x n'est pas nul.

Question 4 :

	0		<0,5		≥0,5		1		Total	
Question 4	3	0,5%	381	69,1%	165	29,9%	2	0,3%	551	100,0%

Dans cette question, les candidats sont invités à prouver qu'une relation binaire est un ordre total sur le nuage de points. La reflexité et la totalité n'ont pas posé de problème particulier. La propriété d'antisymétrie découlait de l'hypothèse générale (trois points du nuage ne pouvaient être alignés). La transitivité était subtile : elle n'est pas valide en général mais l'est ici car le point i fait partie de l'enveloppe convexe.
– Les preuves ne sont pas toujours très soignées : il faut veiller à donner toutes les étapes du raisonnement et surtout résister à l'envie d'affirmer la conclusion trop hâtivement, même lorsqu'elle est donnée par le sujet. On apprécie les étudiants questionnant le sujet à l'aide d'un contre-exemple (même faux) pour le cas de la transitivité.

– Un grand nombre de candidats se sont perdus dans une preuve calculatoire de la transitivité en tombant dans des erreurs de raisonnement dans l'analyse des cas ou dans des manipulations incorrectes d'inégalités. On rappelle d'ailleurs qu'on ne peut pas multiplier les deux membres d'une inégalité par un scalaire sans prendre en compte son signe.

Question 5 :

	0		<0,5		≥0,5		1		Total	
Question 5	125	22,6%	83	15,0%	190	34,4%	153	27,7%	551	100,0%

La question 5 portait sur la recherche du point maximal pour la relation d'ordre total définie dans la question précédente. Cette recherche devait s'exécuter en temps linéaire par rapport à la taille du nuage.

Le sujet invitait les candidats à parcourir tous les points différents de l'argument i et à utiliser une inégalité large. Les candidats qui ont suivi cette voie ont pour la plupart obtenu tous les points. Une autre possibilité consistait à choisir un point initial différent de i et à utiliser une inégalité stricte ; dans ce cas, il n'était pas nécessaire d'éviter i lors du parcours de points.

– De nombreux candidats ont combiné les deux méthodes pour aboutir à un algorithme incorrect.
– Certaines réponses cherchaient une orientation positive plutôt que négative.
– Les algorithmes quadratique, même corrects, ont été moins bien récompensés que les algorithmes linéaires.

Question 6 :

	0		<0,5		≥0,5		1		Total	
Question 6	171	31,0%	40	7,2%	127	23,0%	213	38,6%	551	100,0%

On demandait ici de dérouler l'exécution de l'algorithme de la question précédente.

– La description de l'algorithme devait mentionner la séquence des points considérés, l'état de la variable contenant le maximum, les tests effectués à chaque étape et l'issue de ces tests.
– Il faut veiller à rédiger la réponse à une telle question en suivant une approche systématique, mimant l'interprétation du programme par la machine.

Question 7 :

	0		<0,5		≥0,5		1		Total	
Question 7	67	12,1%	27	4,9%	220	39,9%	237	43,0%	551	100,0%

Cette question portait sur la réalisation de l'algorithme du paquet cadeau en partant du point le plus bas (résultat de la question 1) et en appliquant la fonction de la question 5 tant que l'enveloppe n'était pas refermée.

– Il s'agissait d'écrire correctement une boucle **while** : quand la forme des itérations n'est pas un simple parcours d'une séquence, s'assurer que la boucle commence et termine n'est pas immédiat. Ces deux types d'erreur ont été rencontrées.
– La boucle s'écrivait plus naturellement avec un **do-while**, malheureusement absent en PYTHON : en effet, l'insertion des points dans la liste représentant l'enveloppe convexe devait être en avance d'une étape sur le test à effectuer. Cette difficulté a conduit certains candidats à insérer deux fois le point le plus bas dans la liste ou bien à ne pas l'insérer du tout.

Question 8 :

	0		<0,5		≥0,5		1		Total	
Question 8	104	18,8%	42	7,6%	242	43,9%	163	29,5%	551	100,0%

La question 8 portait sur la justification de la complexité de l'algorithme du paquet cadeau. L'argument important était de remarquer que le nombre d'itérations était proportionnel au résultat, c'est-à-dire à la longueur m de l'enveloppe convexe.
– Une analyse de complexité soignée suit la structure du programme pour en réinterpréter chaque étape à l'aide d'une fonction de coût. Certains raisonnements apportaient des arguments dans un ordre aléatoire rendant difficile la compréhension de leur cheminement.
– Il faut rappeler le sens des quantités considérées (ici n le nombre de points du nuage et m la longueur de l'enveloppe) sans quoi il est difficile pour le correcteur d'être sûr que le candidat associe correctement ces quantités à l'exécution du programme.

Question 9 :

	0		<0,5		≥0,5		1		Total	
Question 9	53	9,6%	59	10,7%	133	24,1%	306	55,5%	551	100,0%

Cette question de cours demandait un algorithme de tri de complexité en $O(n \cdot \log(n))$. La réponse attendue, et la plus courante, était le tri fusion. Les tris par tas, par tournoi, par blocs ont également été acceptés. Le tri rapide est quadratique dans le pire cas, les candidats qui l'ont nommé ont donc été sanctionnés (moins sévèrement s'ils précisaient que la complexité n'était pseudo-linéaire qu'en moyenne). De la même façon, les candidats ayant nommé un tri quadratique – tri bulle, par insertion, par sélection – avec sa complexité correcte ont reçu plus de points que ceux prétendant que leur tri était pseudo-linéaire.

Question 10 :

	0		<0,5		≥0,5		1		Total	
Question 10	105	19,0%	124	22,5%	215	39,0%	107	19,4%	551	100,0%

Cette question difficile portait sur la fonction mettant à jour l'enveloppe supérieure pour l'insertion d'un nouveau point à son extrémité. L'algorithme s'appuyait sur une

pile pour représenter l'enveloppe supérieure et, pour un point P donné, mettait à jour le sommet de cette pile en y retirant les points capturés par l'enveloppe étendue par P. Cet algorithme peut s'écrire de façon itérative ou récursive. Une des difficultés de l'exercice était de formuler correctement la condition d'arrêt de l'algorithme.

- Le sujet imposait l'usage des opérations sur les piles (plutôt que celles sur les listes utilisées pour implémenter ces piles). Cette contrainte – l'idée de voir la pile comme un type abstrait – a été mal comprise et peu respectée.
- La version récursive de cet algorithme était plus simple à formuler que sa version itérative. Pourtant, les candidats ont quasi-exclusivement suivi une approche itérative.

Question 11 :

	0		<0,5		≥0,5		1		Total	
Question 11	136	24,6%	1	0,1%	3	0,5%	411	74,5%	551	100,0%

Le traitement de l'enveloppe inférieure était symétrique à celui de l'enveloppe supérieure. Cette question visait à préciser cette symétrie.

- Il n'était pas nécessaire de recopier l'algorithme de la question 10. Indiquer ce qu'il fallait changer à la réponse de la question 10 était suffisant.

Question 12 :

	0		<0,5		≥0,5		1		Total	
Question 12	191	34,6%	60	10,8%	188	34,1%	112	20,3%	551	100,0%

L'algorithme de construction de l'enveloppe convexe par balayage consiste en un parcours des points triés par abscisses croissantes et à leur insertion dans les enveloppes supérieures et inférieures. Ces deux enveloppes sont ensuite accolées pour former l'enveloppe convexe. Les candidats devaient donner la fonction réalisant cet algorithme.

- L'initialisation des deux piles n'a pas toujours été réussie : il fallait fournir une réponse cohérente avec les questions 10 et 11.
- Les algorithmes de recollement des candidats étaient parfois excessivement complexes : il suffit de déverser une pile dans l'autre en prenant garde à ne pas insérer deux fois les extrémités des deux moitiés de l'enveloppe convexe.

Question 13 :

	0		<0,5		≥0,5		1		Total	
Question 13	467	84,7%	39	7,0%	42	7,6%	3	0,5%	551	100,0%

Pour conclure cette épreuve, on demandait une analyse de complexité de l'algorithme de balayage. Il fallait montrer que le processus de construction des enveloppes supérieures et inférieures était un algorithme linéaire en le nombre de points du nuage. Le coût total de l'algorithme était alors dominé par celui du tri.

L'argument permettant de justifier le coût linéaire de la construction des enveloppes était subtile : il fallait remarquer qu'un point du nuage ne pouvait être inséré et retiré qu'au plus une fois des deux piles.

- Les candidats ayant prouvé correctement que leur algorithme était quadratique ont été recompensés.
- Il n'était pas possible de montrer que chaque étape de mise à jour des enveloppes supérieures et inférieures était logarithmique... et encore moins que $O(n^2) < O(n \cdot \log(n))$!