4. INFORMATIQUE

4.1. Informatique pour tous

Remarques générales

Le jury souhaite attirer l'attention des candidats et de leurs formateurs sur quelques points généraux relevés cette année.

- Les copies sont en général bien présentées, et sauf rares exceptions, l'indentation est assez claire. Rappelons cependant que celle-ci est capitale pour juger de la validité d'un programme, et qu'un doute sur celle-ci est défavorable au candidat. Des espaces clairs utilisant les carreaux ou quelques barres verticales bien placées semblent un bon compromis, et il ne faut pas hésiter à commencer bien à gauche de la copie pour ne pas se retrouver coincé en fin de ligne.
- Certaines copies utilisent des notions ou des fonctions spécifiques à PYTHON, mais imparfaitement maîtrisées. Il est préférable de bien maîtriser ce que demande le programme, plutôt que de tenter une syntaxe hasardeuse utilisant des notions ou fonctions pas clairement maîtrisées, qui rendent le programme invalide.
- L'utilisation de range n'est pas suffisamment maîtrisée. On a l'impression que beaucoup de candidats ne prennent pas le temps de se demander entre quelles bornes exactes les éléments de l'itérateur doivent prendre leurs valeurs, et un parcours d'éléments décroissants donne lieu à des syntaxes souvent fausses. Rappelons que range(a,b,-1) avec a>b permet de parcourir l'ensemble des valeurs de a inclus à b **exclu** par pas de -1. Dans le même ordre d'idée, beaucoup trop de candidats écrivent for i in range(len(L)) : avant de faire appel à L[i+1] dans la boucle sans paraître conscients du problème.
- La question 4 faisait appel explicitement à une notion du programme : l'exploitation de données dans un fichier texte. Les candidats traitant cette question sont rares, et ceux qui la réussissent entièrement sont une infime minorité. Le jury peut s'attendre à ce que cette technique pourtant fondamentale dans les applications pratiques soit traitée de manière plus satisfaisante. La syntaxe nécessaire n'est pas lourde, et elle doit être mieux maîtrisée.
- Il est dommage que dans de nombreuses copies, on trouve des appels à des fonctions complexes au sein d'une boucle, alors qu'un seul appel avant la boucle suffit. Le jury redemande explicitement aux candidats de s'entraîner à éviter ces appels multiples inutiles qui augmentent souvent artificiellement la complexité d'un programme.

Commentaires spécifiques à chaque question

Q1 : Peu de bonnes réponses parfois totalement fausses pour un calcul élémentaire.
Q2 : Il est très surprenant de constater que beaucoup de candidats ne maîtrisent pas le sens du préfixe giga, le confondant avec méga. Cela n'est pas acceptable à ce niveau de formation.
Q3 : La notion de gain **relatif** est trop mal maîtrisée. Le nombre de candidats incapables de conclure simplement que la suppression d'un caractère sur 8 permet d'obtenir un gain relatif d'espace de 1/8 est malheureusement élevé.
Q4 : Question très souvent non traitée (voir les commentaires généraux).
Q5 : Question bien traitée en général.
Q6 : Question bien traitée, sauf par les candidats divisant la somme par len(L)-1, et nous avons été surpris de rencontrer parfois l'opérateur // pour faire la division par len(L), qui n'était pas du tout adapté à cette question. On a par ailleurs constaté un certain nombre de fois l'utilisation d'un compteur pour compter le nombre d'éléments de la liste, alors que la boucle liste était parcourue avec range(len(L))…

Q7 : Question pas toujours bien traitée, les erreurs les plus courantes étant l'oubli du pas, et surtout l'appel à liste_niveaux[i+1] dans la boucle alors que la boucle était construite avec range(len(liste_niveaux)).
Quelques distinctions de cas (if L[i]>L[i+1]… elif L[i+1]>L[i]…) inutiles nous ont surpris, ainsi que de très nombreux cas de décomposition des trapèzes en un rectangle et un triangle.
Q8 : Cette question était un bon test pour la maîtrise de la syntaxe élémentaire du langage. Les erreurs les plus courantes étaient un range allant un entier trop loin et un return -1 mal indenté qui terminait l'exécution de la fonction dès le premier test.
Q9 : Il y avait une petite erreur dans le sujet, puisque le calcul de la valeur moyenne de la liste était nécessairement de complexité linéaire. Il fallait entendre que la seule recherche du dernier PND devait être de complexité unitaire dans le meilleur des cas, ce que les candidats ont compris sans difficulté en général, mais souvent raté à cause d'une maîtrise insuffisante des bornes de recherches (par un range mal utilisé ou un while avec une condition inexacte).
Certains ont tenté d'inverser la liste pour utiliser le résultat de la question 8. Il fallait prendre garde à ne pas renvoyer un passage par la valeur moyenne en montée, ce qui n'était pas le but recherché.
Q10 : Question simple, pas toujours bien traitée.
Q11 : Question de mise en forme plus difficile. La solution la plus simple et élégante utilisait bien entendu la fonction construction_successeurs et le « slicing » de liste_niveaux, ce que les meilleurs candidats ont compris sans difficulté. À l'inverse, nous avons pu lire des solutions mal pensées, voire très alambiquées. On pourrait conseiller à certains candidats un petit temps de réflexion avant de partir tête baissée dans l'écriture d'un algorithme non trivial.
Q12 : Question assez facile, mais il fallait penser que la période de la vague devait être donnée en secondes. Attention également à la structure du résultat renvoyé : une liste de listes à deux éléments n'est pas une liste de deux listes.
Q13 : Cette question classique de recherche de maximum a été souvent bien traitée.
Q14 : Le tri était fait explicitement sur un élément de la sous-liste, il fallait initialiser le pivot en conséquence.
Q15 : Question peu traitée, la distinction des cas qu'elle nécessitait n'a pas toujours été bien comprise.
Q16 : Quelques confusions sur le tri par insertion : le rôle de la variable tmp n'a pas toujours été perçue clairement par les candidats. Il s'agit d'un algorithme au programme, une meilleure maîtrise de celui-ci est souhaitable.
Q17 : Question en général bien réussie. Il est plus surprenant que des candidats qui répondent correctement à cette question soient tombés, au cours de leurs programmes précédents, dans le travers qu'elle souligne.
Q18 : Question assez bien traitée dans l'ensemble, mais les réponses ont parfois manqué de précision dans la justification, et on a parfois dû subir des réponses délirantes, certains candidats allant jusqu'à évoquer des complexités en O(1/n)…
Q19 : La première requête SQL ne pose pas de problème, sauf à ceux qui ne se sont manifestement pas entraînés sur ce langage. Les autres requêtes, plus complexes, ont eu des succès variables. Parmi les erreurs rencontrées, l'appel à un attribut via **attribut.table** à la place de **table.attribut** . De plus, la condition de jointure après ON est bien une condition, et non seulement le nom d'un attribut.
Rappelons que la seule syntaxe exigible du programme pour effectuer une jointure est table1 JOIN table2 ON condition (jointure symétrique simple). Trop de candidats ont voulu se lancer dans des NATURAL JOIN et autres FULL JOIN sans maîtriser précisément leurs effets.
Q20 : Peu de bonnes réponses sur cette question. Mis à part les habituelles réponses de complexité délirantes (O(n!), $O(2^n)$… les candidats ont-ils conscience de ce qu'une telle complexité signifie concrètement?) on a aussi vu beaucoup de complexités en O(ln(n)). Rappelons qu'une complexité logarithmique implique que l'on n'ait pas besoin d'accéder à l'intégralité des données.

Q21 : Question difficile très rarement traitée.

Perles diverses

Comme chaque année, le jury a été ému de trouver dans les copies quelques perles inattendues, auxquelles nous avons décerné cette année les prix suivants :

- Prix « L'informatique pour les nuls » :
  « Q4 : On utilise Word pour ouvrir le fichier. Il y a une tab "insérer tableau". On copie les données, on les colle et on nomme la tab : liste niveaux. On s'assure qu'on choisit une place après la première ligne pour le tableau. »

- Prix « Piège à correcteur »
  « Q14 : Voir la copie. » (écrit sur la copie)

- Prix « SQL, WHAT ? »
  « Q19 : WHAT Hmax FROM tempete »

Nous souhaitons une bonne préparation à cette épreuve aux futurs candidats !

4.2. Informatique — filière MP

Remarques générales

Le sujet traite de la recherche des positions d'un motif dans un texte. Il fait appel, d'une part, à la notion formelle d'automate et, d'autre part, à des structures informatiques complexes que le candidat doit manipuler. L'ensemble permet de bien évaluer l'acquisition du programme des deux années de classe préparatoire.

Les candidats abordent l'ensemble des questions dans leur grande majorité. Ils finissent parfois le sujet (en passant les questions difficiles). Quelques (rares) excellentes copies ont pu être lues.

La présentation des copies est globalement satisfaisante.

Nous avons pu constater peu d'efforts de rédaction des quelques questions théoriques.

Beaucoup de candidats ne donnent pas d'arguments, ou se contentent d'arguments superficiels.

Pour ce qui est de la manipulation des objets de type élaboré, une certaine aisance a pu être globalement appréciée.

Certains codes sont parfois bien trop compliqués ou difficiles à comprendre.

Il est rappelé que les codes doivent être clairs. Utiliser l'indentation est un excellent moyen d'y parvenir.

Première partie : recherche naïve dans un texte.

Questions 1-2

Le but de ces deux questions est de faire découvrir la possibilité du recouvrement des positions du motif dans le texte. Beaucoup de candidats passent à côté de cette finesse et se trouvent ensuite pénalisés dans l'écriture des codes.

Questions 3 - 4