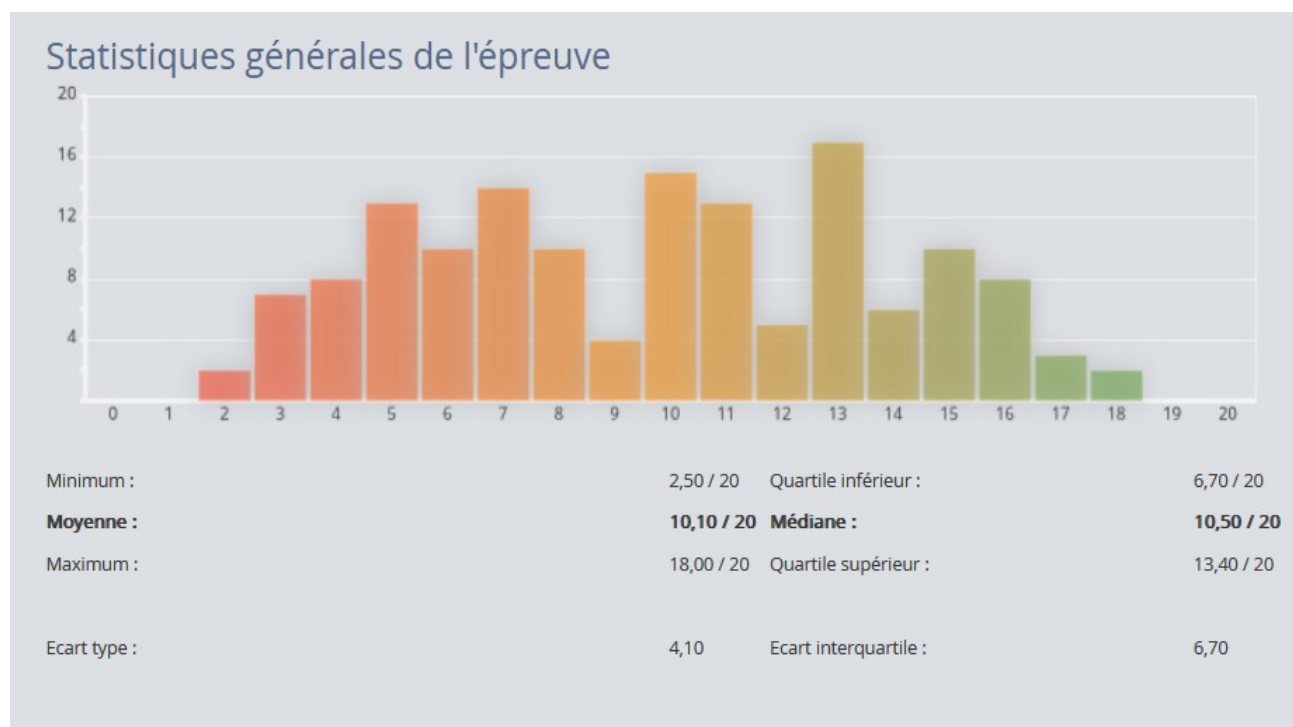


CONCOURS A-TB – 2023

Rapport de l'épreuve écrite d'Algorithmique et Informatique



Cette année le sujet portait sur les automates cellulaires, en présentant trois exemples relativement simples. Un automate cellulaire est un modèle simple d'automates de calcul : constitué d'un réseau de cellules, en général partie d'une droite (automate unidimensionnel), du plan (automate bidimensionnel) ou de l'espace (tridimensionnel) ; chaque cellule peut se trouver à l'instant t dans un nombre fini d'état, et son état à l'instant $t+1$ évolue selon des règles propres à l'automate en fonction des états de la cellule et des cellules voisines. Malgré un principe très simple, les automates cellulaires permettent de représenter des systèmes dynamiques discrets remarquablement variés, riches et complexes.

Le sujet présentait en :

- Partie 1 : un automate cellulaire unidimensionnel pour modéliser l'évolution du trafic routier dans un tunnel.
- Partie 2 : un automate cellulaire bidimensionnel pour modéliser les motifs fractals apparaissant sur certains coquillages (*Conus Textile*).
- Partie 3 : un automate cellulaire bidimensionnel pour modéliser l'évolution d'une épidémie au sein d'une

étable. Dans la partie 3, plus ouverte, on demandait au candidat de détailler librement un algorithme, alors que dans les parties 1 et 2 il était guidé pour le faire.

Le sujet était à la fois relativement accessible et de longueur bien adaptée à l'épreuve ; la plupart des candidats ont pu traiter le sujet dans son intégralité, ou quasiment. Il faisait appel à des compétences de base du programme officiel, essentiellement articulées cette année autour des listes. Il comportait des questions de compréhension, d'écriture de code plus ou moins élémentaire, de QCM, et des questions de complétion de code.

Cette année encore on constate une hétérogénéité marquée du niveau des candidats, une part de niveau relativement faible, celle des candidats ne sachant pas écrire un code répondant à une problématique simple, mais une part importante aussi de bons à très bons candidats. En moyenne la réussite cette année à l'épreuve est convenable.

Les notes s'étalent de 2,5 à 18 sur 20, avec une moyenne d'environ 10,1, et un écart-type de 4,1 ; la médiane est à 10,5 et les quartiles inférieurs et supérieurs à 6,7 et 13,4.

La distribution observée dénote une part importante de candidats de niveaux satisfaisants, mais aussi de candidats de niveau inférieur aux attentes.

En résumé le déroulement de l'épreuve de cette année est satisfaisant. Candidats et préparateurs sont encouragés à maintenir un bon niveau d'exigence dans la préparation de l'épreuve. Les domaines perfectibles concernent l'écriture de code syntaxiquement et logiquement correct ; notamment sur les points suivants :

- L'indexation correcte de la fonction *range* au sein d'une boucle *for*. Elle pose problème à l'immense majorité des candidats. On peut parfois mettre à profit la propriété des listes, chaînes de caractères et autres séquences, d'être itérables pour implanter une boucle *for* sans utiliser la fonction *range*, et le jury persiste à encourager cette approche, lorsqu'elle est possible.
- L'ajout d'un élément en fin de liste. Rappelons à ce sujet qu'il faut privilégier l'usage de la méthode *append*. L'instruction $L = L + [a]$ est à proscrire (car beaucoup moins efficace) et l'instruction $L += [a]$ à déconseiller (une majorité de candidats y oubliant les crochets), au seul profit de *L.append(a)*.
- Parcours de liste ou de chaînes de caractère, au sein d'une boucle *for*, somme de termes consécutifs d'une liste numérique.
- Usage convenable de l'extraction de liste ou chaînes : $C[i:j]$ extrait la sous-séquence de *C* allant de l'indice *i* inclus à l'indice *j* exclu.
- Recherche de minimum, et ses variantes, comme la recherche d'indices où il est atteint.
- Portée des variables : variables locales ou globales.
- Le bon usage des connecteurs logiques *and*, *or*, *not*.

Le jury insiste sur le fait que ce qui discrimine les bonnes copies des moins bonnes, c'est l'écriture de code algorithmiquement et syntaxiquement correct, même sur des algorithmes très basiques ! C'est d'évidence le point à travailler en priorité : coder. Une réussite convenable à l'épreuve ne peut se passer d'une pratique régulière, investie et rigoureuse de la programmation par les candidats durant leurs 2 à 3 années de préparation.

Les candidats peinent aussi largement à identifier qu'un code écrit ou donné dans une question intermédiaire est en général appelé à être utilisé dans les questions suivantes ! C'est un écueil sur lequel nous conseillons aux préparateurs d'insister lourdement.

Dans le détail.

PRELIMINAIRE. Loi de Bernoulli

Il ne s'agissait que d'écrire une fonction simulant une variable de Bernoulli. Elle n'a été réussie que par environ la moitié des candidats. Une erreur récurrente consiste à simuler $B(1-p)$ plutôt que $B(p)$.

PARTIE 1. Trafic routier dans un tunnel.

On présentait ici un exemple, avec variante, d'un automate cellulaire 1-dimensionnel modélisant l'évolution du trafic dans un tunnel.

Q2 et Q3. Question élémentaires sur les listes ; très bien réussies ; encore quelques confusions avec nombre d'éléments et plage d'indices.

Q4. Compréhension des règles d'évolution de l'automate. Bien réussie.

Q5. Complétion du code opérant cette évolution ; réussite variable à cette question.

Q6. Complétion du code opérant une évolution plus complexe (tenant compte d'un facteur aléatoire) : réussite encore variable.

Q7. Complétion du code de la fonction principale d'évolution de l'automate : question bien réussie.

PARTIE 2. Motif sur un coquillage.

Q8. Compréhension de la règle d'évolution de l'automate cellulaire : question très bien réussie.

Q9. Complétion du code donnant l'évolution de l'état d'une cellule. Question assez mal réussie.

Q10. Complétion de la fonction principale d'évolution de l'automate. Question bien réussie.

Q11. QCM ; très bien réussi.

PARTIE 3. Epidémie dans une étable.

Dans cette partie, plus ouverte, on présente un automate pour modéliser la propagation d'une épidémie ; il comporte un facteur aléatoire. On demandait au candidat de proposer un algorithme.

Une version Python du code de l'algorithme n'était pas demandée. Mais une proposition d'algorithme doit être très précise, de sorte que donner le code Python de cet algorithme était peut-être la meilleure façon d'y répondre.

Une réponse sans code devait rester assez semblable dans sa structure au code Python ; on a vu trop de propositions d'algorithmes sous la forme : « on écrit un programme qui fait évoluer chaque cellule selon la règle d'évolution » ou autres paraphrases de l'énoncé ; une telle réponse n'est en rien acceptable.

Sans surprise, la réussite à cette question a été très variable, et seuls les meilleurs candidats ont pu retirer des points à cette question, bien que de nombreux candidats s'y soient essayés.