Informatique

Présentation du sujet

Le sujet est construit autour d'un des thèmes du programmes de seconde année, le traitement des images. Il s'intéresse à la mise en œuvre de méthodes numériques visant à concevoir des photomosaïques, images composées à la manière d'une mosaïque d'une multitude de petites images appelées vignettes. Le sujet comporte 32 questions réparties sur 4 parties et fait largement appel aux connaissances algorithmiques et pratiques du programme de première année :

— la première partie traite du codage des images en termes de pixels et de codage RGB pour se terminer par l'écriture d'une fonction de conversion d'une image en niveaux de gris ;

— la deuxième partie étudie plusieurs solutions algorithmiques de redimensionnement d'images de complexités temporelles différentes. La partie se termine par une synthèse discutant des usages respectifs de ces solutions ;

— la troisième partie aborde le thème des bases des données par l'écriture de requêtes sélectionnant une image source et des vignettes ;

— la quatrième partie aboutit à la construction d'une photomosaïque. Les deux dernières questions laissent une part importante à l'initiative des candidats.

Outre la maitrise des connaissances informatiques du programme, l'écriture syntaxiquement correcte de codes et l'analyse de leurs performances, le sujet évalue l'aptitude des candidats à porter un regard critique sur des propositions de codes. Ce sujet a très largement permis au jury d'évaluer la qualité et le niveau de compétences de chaque candidat.

Analyse globale des résultats

Au regard de la longueur et de la difficulté de l'épreuve, le jury est satisfait du niveau général des copies. Les connaissances informatiques semblent globalement acquises, les langages Python et SQL convenablement maitrisés. Quelques rares candidats ont visiblement négligé la formation, tentant de répondre aux questions ne relevant pas immédiatement du domaine de l'informatique. Ces copies conduisent à des notes généralement très faibles.

La moitié des candidats de la filière PC traite pratiquement 65 % des questions. Un très faible pourcentage ne traite que moins de 20 % des questions.

De nombreux candidats ont fourni des copies d'excellente qualité. Le jury regrette le niveau parfois très bas d'autres copies. Il serait souhaitable que les candidats mesurent l'importance de la formation initiale en informatique pour la suite de leurs études mais également pour leurs cultures d'ingénieur et de citoyen.

Commentaires sur les réponses apportées et conseils aux futurs candidats

Les compétences en matière de programmation élémentaire semblent acquises chez le plus grand nombre de candidats. Néanmoins, le jury souhaite attirer l'attention des futurs candidats sur les points suivants.

— La notion de *type* est essentielle en informatique. Il convient d'en tenir compte lors de la manipulation d'objets informatiques. En particulier, les candidats doivent s'interroger sur la pertinence et les limites de certaines opérations effectuées sur ou entre objets de même type.

— La *complexité* est souvent estimée au regard du nombre d'opérations effectuées dans tout ou partie d'un code. Un minimum d'explications est attendu pour justifier le résultat qui doit, en outre, être exprimé

avec les notations strictes de l'énoncé. Ainsi, affirmer que la présence de deux boucles imbriquées induit une complexité quadratique, souvent notée $O(n^2)$ sans préciser la nature de n, est insuffisant. Ces questions attendent une argumentation fondée menant à l'écriture de complexités sous la forme, par exemple, $O(h \times w)$ puis $O(n)$ après avoir rappelé que $n = h \times w$ (cf. question 9).

— La lecture et l'*analyse de codes* comptent parmi les activités de tout futur ingénieur. Elles requièrent plus qu'un simple survol du code et plus encore qu'un commentaire de type *paraphrase*. Il convient d'abord de préciser le rôle et les objectifs d'une fonction ou d'un bout de code puis d'identifier des blocs structurels importants du code et d'en expliquer leur fonction.

— Les *requêtes SQL* doivent faire l'objet d'une attention particulière. Il s'agit de répondre exactement à la question, sans oublier d'attributs dans la réponse, sans oublier les jointures, etc. Si les candidats maitrisent l'écriture de requêtes très élémentaires, de nombreuses réponses sont souvent incomplètes, voire mal écrites, en raison d'une lecture incomplète ou erronée des questions.

— Les codes sont globalement syntaxiquement corrects et lisibles. Leur lecture révèle toutefois une écriture au fil de l'eau. Il serait souhaitable qu'avant même d'écrire une fonction, chaque candidat s'interroge sur l'*organisation* et la *structure logiques des codes* qu'il propose. À cela s'ajoute la présence de commentaires parfois inutiles dans le corps du code. Un commentaire n'a de sens que s'il apporte une information utile et pertinente pour comprendre le code. Les docstrings (documentation placée immédiatement après la définition d'une fonction) sont toujours utiles en pratique mais généralement pas attendues sur une copie dans le cadre d'une épreuve de concours en temps limité.

— Des points dits *transversaux* ont été attribués pour valoriser la clarté des explications, la qualité rédactionnelle, le respect de la syntaxe de Python, la lisibilité des codes et les commentaires pertinents. Le jury est particulièrement attentif à ces compétences transversales.

Signalons par ailleurs quelques erreurs de syntaxe générales :

— écriture *à l'envers* des affectations, `10 = a` ;

— mauvaise gestion des `range` en ajoutant 1 à la valeur finale pour parcourir toute la liste ;

— l'incrémentation avec `+=` devient parfois `±` ;

— le symbole `*` de la multiplication est souvent omis.

I Pixels et images

Q1. Cette première question a mené à des réponses de qualité variable. Près d'une fois sur deux, le décompte du nombre de couleurs est incorrect et l'application numérique simple est souvent omise.

Q2. De nombreuses réponses sont erronées en raison essentiellement d'un manque de rigueur. La réponse attendait d'une part un objet de type clairement défini, d'autre part une proposition qui respecte les contraintes liées au codage RGB.

Q3. Très peu de candidat ont traité convenablement cette question. La principale erreur est liée à l'absence de prise en compte du type des objets manipulés, conduisant inévitablement à des erreurs dans les calculs demandés. Beaucoup de copies tentent d'écrire les opérations sous forme binaire.

Q4. Cette question est globalement bien traitée. Une attention particulière doit, là encore, être portée sur le type du résultat renvoyé. Beaucoup d'erreurs sont liées à un manque de rigueur dans le suivi des consignes. L'énoncé demandait *la meilleure approximation entière* (qui n'est ni la partie entière, ni le quotient euclidien d'une division) d'une moyenne, retournée sous le type `np.uint8`.

Q5. Cette question est comprise par l'ensemble des candidats mais les réponses sont souvent imprécises ou incomplètes. Deux points structuraient la réponse : un premier point détaillait le contenu de `source.shape`, un second point précisait la signification et le contenu de `source[0,0]`. La rédaction est souvent trop vague. Parler de *largeur* et de *longueur* d'une image est ambigu.

Q6. Cette question de codage est bien réussie par les candidats. Certaines réponses utilisant `a.shape` oublient parfois qu'en raison de la nature même de `a`, cette instruction renvoie un triplet. Le type des éléments du nouveau tableau renvoyé par la fonction est parfois oublié. Enfin, l'énoncé demandant de retourner une image en niveaux de gris qui est un tableau à deux dimensions, il est incorrect de modifier le tableau `a` passé en argument.

II Redimensionnement d'images

Q7. Dans cette question globalement comprise par les candidats, les explications sont parfois confuses même si les résultats sont corrects. Le jury est attentif à la qualité rédactionnelle et aux explications fournies. Même s'il fait preuve de bienveillance, les réponses succinctes, de type *steno*, sont pénalisées.

Q8. Cette question est globalement bien traitée. Une erreur récurrente est observée dans les arguments calculés qui permettent de sélectionner les éléments du tableau `A`. Certaines réponses ont tendance à utiliser `W` et `H` sans les définir dans la fonction. Des confusions sont également faites dans l'utilisation des dimensions `w` et `h`, parfois interverties.

Q9. Bien qu'à priori relativement simple, cette question a révélé la difficulté de nombreux candidats à argumenter leurs calculs de complexité. Une telle question attend une réponse détaillée : opérations prises en compte, décomptes du nombre de ces opérations, expressions du résultat final en respectant les notations strictes de l'énoncé. La seule réponse à la question ne suffit pas à obtenir tous les points.

Q10. Trop souvent l'explication se résume à une simple paraphrase des lignes du code, sans montrer une compréhension de celles-ci. Parler des valeurs « autour de `A[i,j]` » est bien trop vague. L'exercice de lecture et d'analyse d'un code attend bien évidemment plus qu'une lecture ligne à ligne. Comme le signale le début de ce rapport, le rôle du code analysé doit être précisé. S'agissant dans le cas présent d'une fonction, étant données des informations en entrée, le résultat renvoyé et son type doivent être indiqués. Ensuite, il convient d'identifier les blocs structurels importants et les étapes clés des calculs effectués dans le corps de la fonction. Ainsi, par son argumentation, le candidat montre sa capacité à prendre du recul par rapport à la question.

Q11. Comme pour la question 9, cette question a révélé les difficultés d'une présentation claire et rigoureuse des nombres d'opérations menant à l'établissement d'une complexité temporelle. Cette question est peu réussie.

Q12. Cette question a été soit très bien réussie, soit pas réussie du tout ou non traitée. Elle s'appuie sur des connaissances de cours simples.

Q13. Cette question plus délicate nécessitait une réflexion préalable à l'écriture de la fonction. Malheureusement, ce travail de préparation, trop souvent clairement absent au vu des productions, a abouti à l'écriture de fonctions incomplètes ou fausses. Quelques trop rares copies ont proposé de bonnes solutions. Une lecture attentive de l'énoncé aurait également pu éviter certaines erreurs comme par exemple le dimensionnement incorrect `H*W` d'un tableau alors qu'il était attendu `(H+1)*(W+1)`.

Q14. Cette question a fait l'objet d'un traitement variable. Les réponses présentent les mêmes défauts que ceux énoncés pour la question 10.

Q15. Cette question amène les mêmes commentaires que ceux des questions 9 et 11.

Q16. Cette question amène les mêmes commentaires que ceux de la question 10.

Q17. Quand elle traitée, cette question est peu réussie. Elle nécessitait de prendre du recul par rapport à l'ensemble des questions de la deuxième partie.

Q18. Cette question marquant la fin de la deuxième partie, quelques candidats ont fourni des réponses partielles en discutant la qualité des images obtenues.

III Sélection des images de la banque

Q19. Cette question est globalement très bien traitée par l'ensemble des candidats.

Q20. Cette question est bien traitée par l'ensemble des candidats. De nombreuses copies utilisent `USING` proposé en annexe. Signalons malgré tout quelques erreurs liées à une mauvaise écriture de la jointure. Attention également aux erreurs de syntaxe comme l'utilisation de `==` dans la clause `WHERE`.

Q21. Cette question nécessitait l'écriture de deux jointures. À ce sujet, les réponses sont inégales. Un nombre non négligeable de candidats ne maitrise pas l'écriture de jointures multiples. Par ailleurs, certaines clauses `WHERE` de fin de requête sont maladroitement écrites, avec un `OR` parfois hasardeux. Plusieurs candidats ont pensé que `PH_auteur` était le prénom de l'auteur alors que le type `integer` est clairement mentionné dans la table `Photo`. Néanmoins, plus de la moitié des candidats apporte une réponse tout à fait satisfaisante à cette question.

Q22. Cette question nécessitait de jointer convenablement trois tables avec deux conditions entre les tables `Photo` et `Present` par exemple. Une erreur fréquente réside dans l'oubli d'une de ces deux conditions. L'extraction de l'année, expliquée dans l'annexe, est souvent fausse.

Q23. Cette question a été peu traitée. Une réponse pouvait être formulée en un `INTERSECT` avec un `EXCEPT` ou un `INTERSECT` avec un `COUNT`.

Q24. Cette question laissait une place importante à la prise d'initiative. Néanmoins, la réponse proposée devait respecter le cahier des charges exprimé par les points a. et b. de l'énoncé, ce qui a conduit à des réponses parfois incomplètes.

Q25. Suite naturelle de la question 24, cette question est convenablement traitée par moins de la moitié des candidats. Peu de copies proposent une réponse satisfaisante et rigoureuse.

IV Placement des vignettes

Q26. Le plus simple était d'utiliser convenablement la fonction `procheVoisin` définie à la question 8. Les arguments passés à cette fonction sont parfois incorrects. Le traitement global de cette question est donc très variable.

Q27. Cette question est peu réussie en raison essentiellement de la non prise en compte du dépassement de capacité. Les entiers utilisés n'étant pas signés, `abs(a-b)` est généralement différent de `abs(b-a)`. Une solution consistait à redéfinir les images `a` et `b` en tableaux de type `np.int64` ou bien à calculer des différences de la forme `int(a[i,j])-int(b[i,j])`.

Q28. Cette question a globalement été réussie.

Q29. Cette question abordée par beaucoup de candidats nécessitait l'écriture d'un code organisé, appelant des fonctions définies aux questions précédentes. Les réponses sont variables en raison d'un manque d'organisation préliminaire des idées qui aurait permis une écriture plus aisée du code.

Q30. Peu de candidats ont abordé cette question avec succès.

Q31. et **Q32**. Les deux dernières questions, relativement ouvertes, laissaient la part belle aux propositions des candidats. Si la question 31 a permis de lire quelques propositions pertinentes de stratégie, son implantation dans la question 32 n'a fait l'objet que de très rares réponses correctes et complètes.

Conclusion

Les résultats à cette épreuve montrent que les étudiants, soutenus par leurs professeurs, ont acquis des compétences certaines en informatique. Le jury encourage les futurs candidats à travailler l'informatique en alliant réflexion sur feuille de papier et mise en œuvre des algorithmes sur ordinateur.