Informatique

Présentation du sujet

Le sujet porte sur la prévention des collisions aériennes, à travers l'étude des stratégies réellement mises en œuvre dans la gestion d'un trafic en expansion constante. Il est composé de trois parties. La première introduit la problématique, au moyen de la présentation d'une base de données sur les vols en Europe. La seconde précise l'organisation des routes aériennes autour de la notion de couloir de circulation. Enfin la dernière étudie le système effectif (*Traffic Collision Avoidance System*) utilisé pour assurer la sécurité de la phase de croisière.

Chaque partie évalue des compétences complémentaires : la première évalue la faculté à interroger une base de données au moyen de requêtes SQL. La seconde est d'essence algorithmique, avec de nombreuses évaluations de complexité ; elle valide également l'acquisition des fondamentaux du langage. La troisième laisse plus d'initiative aux candidats, en leur demandant à travers l'écriture de fonctions, de concevoir un algorithme répondant à un problème précisément posé.

Analyse globale des résultats

L'épreuve d'informatique est une épreuve simple : les meilleurs candidats ont traité de façon satisfaisante plus de 95% du problème. Elle vise à valider un socle minimum de compétences informatiques que doit posséder le futur ingénieur, mises en œuvre dans un contexte réel. Les candidats en ont bien compris l'enjeu : la moyenne brute de l'épreuve est voisine de 10/20. Les notes sont bien étalées, gage d'une évaluation de qualité.

Au delà de la correction d'un algorithme ou d'une fonction, le jury attache une grande importance à la qualité du code : respect de l'indentation et de la syntaxe, utilisation de l'expressivité du langage pour obtenir des programmes concis, introduction de variables pertinentes, choix intelligent des noms de variables, décomposition d'un algorithme en plusieurs parties naturelles, etc. sont autant de savoir-faire qui rende le code facile à lire et à comprendre. Le barème valorise ces qualités tout au long du problème.

Commentaires sur les réponses apportées et conseils aux futurs candidats

Partie I

L'écriture de requêtes sur les bases de données est une compétence relativement bien acquise. Le jury déplore que 10% des candidats fassent l'impasse sur cette partie pourtant bien pondérée. La notion d'auto-jointure, nécessaire dans la dernière question, est celle qui a posé plus de difficultés aux candidats.

Partie II

II.A, B – Ces questions demandaient d'écrire quelques fonctions itératives simples, avec une ou plusieurs boucles, et ont été bien traitées par la majorité des candidats. Le sujet demandait de donner la complexité des fonctions demandées. Si la notion est bien comprise, son expression est trop souvent maladroite (« la complexité est $9n^2 + 3n$ »), ou lourdement justifié par un décompte exhaustif des opération élémentaires qui s'étale parfois sur plusieurs pages, là où une analyse

de la structure des boucles est tout aussi convaincante et efficace. De nombreux candidats en oublient qu'une fonction de complexité linéaire dans un boucle conduit en principe à une complexité quadratique.

II.C – Les fonctions demandées étaient un peu plus longues et nécessitaient un peu plus de réflexion sur la signification des données manipulées. Les candidats ayant une lecture trop superficielle de la problématique du sujet s'y sont égarés.

II.D – La question, plus libre, exposait la mise en œuvre de la technique du recuit simulé. Il s'agit de mettre en œuvre une succession d'étapes élémentaires, répétées, qui conduisent à diminuer une fonction de cout. Tous les candidats (60%) qui se sont donné la peine de s'approprier la méthode ont fourni des solutions globalement valides et ont été récompensés. Le jury incite les futurs candidats à ne pas renoncer devant une apparente difficulté. Trop peu de candidats ont mentionné que la solution proposée n'est pas optimale.

Partie III

Cette partie appelait à décrire les procédures mises en œuvre réellement pour éviter les collisions. Elles faisait appel aux qualités de compréhension et de bon sens des candidats. Dans ces questions moins guidées, le jury juge autant la structure et la lisibilité du code que son aspect purement fonctionnel.

Pour illustrer ce propos, la question **III.B.4)** demandait de travailler sur les coordonnées et la vitesse d'un avion. 40% des candidats qui ont abordé la question ont posé :

```
id, x, y, z, vx, vy , vz, t0 = intrus
```

Il va de soi que la suite du code est claire et concise, et qu'elle évite des expressions telles que :

```
if intrus[1]*intrus[4]+intrus[2]*intrus[5]+intrus[3]*intrus[6]>0:
```

plus difficile à lire, comprendre, vérifier et même écrire. Le barème bonifie les réponses des candidats ayant pris cette initiative.

Autre illustration : la question **III.C.2)** demandait de trier une liste où une seule valeur n'était pas à sa place. Certains proposent une solution quadratique, ce qui est visiblement maladroit. D'autres une solution linéaire, ce qui est mieux. D'autres enfin s'attachent, dans l'esprit de la question, à minimiser le nombre d'opérations élémentaires. Il va de soi que le jury pondère ces différentes approches, même si elles répondent toutes à la question.

Conclusion

L'impression globale de sérieux de la préparation perçue l'an passé perdure pour cette deuxième session. Pour poursuivre dans cette voie, le jury invite les étudiants et leurs formateurs à insister sur la lisibilité du code, en profitant en particulier des traits du langage qui favorisent celle-ci.

Informatique

Présentation du sujet

Le sujet porte sur le thème du trafic aérien, et en particulier la gestion optimisée de l'attribution des niveaux de vol par Eurocontrol et le système anti-collision TCAS.

La première partie traite des requêtes d'extraction des données de vol utiles, à partir des bases de données stockant les caractéristiques des vols programmés et des aéroports.

La deuxième partie s'intéresse à l'attribution des niveaux de vol aux compagnies aériennes. Plusieurs algorithmes sont envisagés, permettant de détecter les conflits et de minimiser les couts de changement de niveaux de vol. Le nombre important de vols à gérer à l'échelle européenne nécessite une attention particulière au regard de la complexité des solutions envisagées.

La troisième partie porte sur le système anti-collision TCAS, embarqué dans les avions, qui traite les données de vol reçues de la part des autres avions environnant afin d'alerter le pilote en cas de danger de collision.

Analyse globale des résultats

Le sujet est de longueur raisonnable pour le temps imparti. De nombreux candidats abordent la totalité du sujet.

À nouveau cette année, le jury se réjouit du niveau satisfaisant des copies. Le langage est bien maitrisé et permet de traduire les solutions aux questions sans difficultés. Seule une petite proportion des candidats (de l'ordre de 5%) a de réelles difficultés à construire un programme, en respectant les bases de syntaxe.

Les petites erreurs syntaxiques n'ont pas été retenues par le jury comme un élément discriminatoire, dans la mesure où elles ne cachent pas des erreurs de fond. Les réponses pertinentes d'un point de vue algorithmique sont valorisées.

Les questions relatives aux bases de données, placées en début de sujet, sont relativement bien réussies cette année.

Commentaires sur les réponses apportées et conseils aux futurs candidats

Au regard des copies évaluées, le jury propose aux futurs candidats de prêter attention aux remarques suivantes.

L'indentation en python délimite les blocs d'instructions et doit apparaitre clairement dans la rédaction.

L'initialisation d'une variable dans une boucle ou hors de la boucle n'a pas les mêmes conséquences pour l'algorithme.

Le nombre d'itérations d'une boucle doit être bien réfléchi, en notant que l'instruction `range(n)` parcourt `n` itérations indicées de `0` à `n-1`.

La somme de deux listes `L1+L2` conduit à la concaténation des listes, la somme de deux tableaux (`numpy.ndarray`) `A1+A2` conduit à la somme des éléments du tableau.

La concision et l'élégance des programmes sont appréciées dans l'évaluation. Les candidats qui réinvestissent les fonctions déjà codées sont valorisés par rapport à ceux qui recopient les lignes de code équivalentes. Bien souvent, une condition booléenne bien choisie permet d'éviter de longues listes de conditions aux instructions identiques.

Des noms de variables explicites aident à la compréhension du code. De trop nombreux candidats utilisent des noms de variables non significatifs (`a`, `b`, `c`, ...) ce qui nuit à la compréhension du programme. La clarté du programme (en particulier le choix des noms de variables) ainsi que la présence de commentaires sont prises en compte dans l'évaluation.

Dans une démonstration ou dans l'écriture d'un code, une justification minimale est attendue.

L'ordre des questions importe. Prendre soin de rédiger les réponses aux questions en respectant leur ordre dans le sujet.

La présentation d'une copie fait également partie des compétences attendues d'un candidat à une école d'ingénieur. Le correcteur n'attribue les points qu'aux éléments de réponse qu'il parvient à lire et à comprendre.

Les variables utilisées dans une fonction doivent être définies dans cette fonction ou être explicitement définies comme variables globales (soit par le sujet, soit par le candidat). Beaucoup de candidat ont utilisé la variable `n` sans la définir, ce qui a soulevé une ambiguïté, `n` pouvant être le nombre de vol comme le nombre de sommets.

Les candidats sont invités à lire attentivement l'annexe contenant certaines fonctions utiles pour traiter le sujet.

Enfin, le jury invite les candidats à ne pas se décourager en milieu d'épreuve lorsqu'une question difficile n'est pas réussie. Les sujets comportent de nombreuses parties indépendantes permettant de valoriser les compétences d'un candidat.

Première partie

Les questions sur les bases de données sont abordées par pratiquement tous les candidats, et souvent de façon satisfaisante.

La fonction `count` dans la première question est souvent oubliéee, ou remplacé par autre chose (`len` par exemple) faute de savoir quoi mettre.

La question **I.D** conduit souvent à des jointures inappropriées, en particulier avec la table `aeroport`.

Deuxième partie

Cette partie débute avec des fonctions nécessitant des parcours de listes et des conditions. Les sous-parties ±QII.A et **II.B** sont bien abordées par la plupart des candidats. Le paramètre `n` n'étant pas défini comme une variable globale, il convenait de le définir dans les fonctions à partir des dimensions des tableaux. Certains candidats confondent l'affectation (`=`) et le test d'égalité (`==`). De même, le test de différence a parfois été noté `=!` au lieu de `!=`. Le jury a été indulgent cette année vis-à-vis de ces erreurs, mais souhaite attirer l'attention des candidats sur ces nuances.

Un bon nombre de candidat ne semblent pas à l'aise avec la notation « grand O » et préfèrent parler de complexité en $9n^2$ plutôt qu'en $O(n^2)$.

Les sous-parties **II.C** et **II.D** sont très souvent abordées mais sans recueillir la totalité des points. Certains candidats oublient dans la fonction `cout_du_sommet` de ne pas compter les sommets supprimés, ou au contraire ne comptent pas les sommets non encore attribués.

L'algorithme de recuit simulé (sous-partie **II.D**) montre une forte disparité entre les meilleurs candidats qui proposent un algorithme juste et concis, d'autres qui proposent un algorithme très long suite à des répétitions, et d'autres encore qui n'abordent pas la question. À noter que la syntaxe `liste1 = liste2` en python ne permet pas d'obtenir une copie de `liste2` mais fournit simplement deux noms pour accéder au même objet.

Troisième partie

La sous-partie **III.A** est généralement bien abordée, les candidats sachant plutôt bien comment les nombres sont représentés en mémoire, même si les réponses comportent souvent de petites erreurs de calcul.

Les sous-parties **III.B** et **III.C** s'intéressent au calcul et à la mise à jour de la liste d'intrus, ordonnée en fonction de la dangerosité. Ces questions sont moins abordées et les réponses rarement correctes. Beaucoup de candidats n'ont pas lu l'annexe et n'utilisent pas la fonction `time`.

La question **III.C.2** aborde le cas d'un tri dans une liste déjà triée, où seule une ligne est à replacer. Cette question nécessitait quelques explications sur la stratégie de tri adoptée.

La question **III.C.3** n'a pas été comprise par la plupart des candidats qui l'ont abordée.

La sous-partie **III.D** permet de conclure l'étude sur les performances du TCAS. Ces questions, lorsqu'elles sont abordées, sont souvent correctement traitées.

Conclusion

Le sujet aborde la majeure partie du programme d'informatique commune. Le choix d'un sujet s'appuyant sur une application concrète de l'informatique assure une cohérence avec la formation d'ingénieur. Cette approche sera reconduite sur des problématiques de simulation ou d'algorithmique courantes en informatique, à partir du programme des 3 semestres d'informatique.

Les bons résultats à cette épreuve montrent que les étudiants, soutenus par leurs professeurs, ont su montrer des compétences affirmées 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.

Informatique

Présentation du sujet

Le sujet proposé traite d'un point de vue informatique les risques de collision aérienne. Une première partie permet de traiter des données en utilisant des requêtes SQL. La deuxième partie présente différents algorithmes de régulations de vols via l'utilisation de listes et de matrices. La troisième partie traite de la surveillance pratique des différents vols et fournit un traitement informatique des données récoltées.

Analyse globale des résultats

Le sujet est de longueur correcte et certains candidats ont traité l'intégralité du sujet. Il a permis un étalement des notes satisfaisant. On peut noter que par rapport à la session précédente, les élèves semblent avoir pris conscience de l'importance de cette nouvelle épreuve et il est donc très rare que le langage Python ne soit pas maitrisé. On note donc assez peu d'erreurs syntaxiques, ce qui est une bonne évolution.

Néanmoins, un effort doit être fait pour la rédaction des différentes fonctions et la mise en valeur des arguments proposés. Il est important que les fonctions soient bien indentées et que le code soit lisible. Dissocier le code des commentaires est apprécié.

Commentaires sur les réponses apportées et conseils aux futurs candidats

Partie I

Cette partie a été abordée par la grande majorité des candidats mais le langage SQL n'est pas encore totalement maitrisé. Un effort est à faire au niveau de la compréhension des jointures et du respect de la syntaxe.

I.A – L'utilisation de la fonction `COUNT` n'est pas assimilée par la totalité des candidats et on peut constater dans quelques copies une confusion avec la fonction `SUM`.

I.B – La jointure lorsqu'elle est évoquée est plutôt bien traitée même si les aéroports parisiens ne limitent pas à Charles De Gaulle.

I.C – Quelques confusions logiques entre « ou » et « et ».

I.D – Question plus difficile qui a été mal traitée en général, même si l'énoncé fournit un exemple d'auto-jointure.

Partie II

Partie abordée par la quasi-totalité des candidats. La notion de complexité n'est pas correctement assimilée et la notation O est parfois inexistante ou non comprise. Certaines variables ou quantités sont utilisées sans être préalablement définies. De plus, la structure de la matrice proposée dans l'énoncé n'a pas toujours été comprise.

II.A – Des candidats ne tiennent pas compte du fait que la matrice utilisée est symétrique et comptent deux fois certaines quantités. Beaucoup utilisent une variable `n` qu'ils n'ont pas pris le soin de définir.

II.B – Beaucoup de fonctions écrites ne comportent pas de `return`. Quelques erreurs concernant l'utilisation des listes sont à noter. La syntaxe `L=[0,0,...,0]` n'est pas correcte en python.

Une complexité exponentielle n'est pas forcément rédhibitoire ! Le nombre de vols journaliers devait être cité et il était intéressant de donner des exemples chiffrés pour pouvoir donner un ordre de grandeur du nombre d'opérations élémentaires effectuées.

II.C – Quelques algorithmes fournis sont un peu lourds et parfois pénibles à décrypter. Toute fonction non déjà définie utilisée dans une copie doit être précisée.

II.D – Traitée par une grande partie des candidats. Question qui ne posait pas de réels problèmes si l'on avait bien lu et compris l'énoncé.

Partie III

III.A et **III.B** – Plutôt bien traitées en général même si on a pu trouver des résultats allant de l'ordre du kilo-octet au téra-octet. On peut noter parfois une confusion entre Go et Gio.

Pour l'extrapolation de la trajectoire, beaucoup de candidats oublient la condition initiale ce qui a aussi une influence sur les questions suivantes.

On a aussi pu constater parfois l'utilisation d'une fonction `produit_scalaire` qui simplifiait le travail mais qui n'a jamais été définie.

Peu de candidats pensent à utiliser des variables locales ce qui fait que le code fourni est souvent lourd.

III.C – Peu de candidats ont correctement traité cette partie qui nécessitait de rapidement assimiler l'énoncé.

Un algorithme de tri était proposé. Il n'était donc pas souhaité voir un algorithme du cours que les candidats souhaitaient absolument replacer.

III.D – Tous les candidats qui ont abordé cette partie ont eu des points. On a souvent pu constater un oubli de la vitesse relative entre les avions.

Conclusion

Le jury est satisfait du niveau global et de la qualité des copies. Cette épreuve a permis aux candidats de valoriser l'enseignement d' informatique suivi au cours de leurs années de préparation. Le jury recommande aux futurs candidats de s'investir de manière sérieuse dans cette discipline dès la première année et d'écrire et d'analyser régulièrement des algorithmes, aussi bien sur feuille que sur machine.

Informatique

Présentation du sujet

Cette épreuve traite de la sécurité aérienne, en particulier du référencement des vols et de leur niveaux (altitude de croisière), de l'attribution de ces niveaux, et du traitement en temps réel des trajectoires pour éviter les collisions entre appareils.

Le sujet aborde les bases de données et requêtes SQL et l'algorithmique autour d'un graphe et d'autres listes ou tableaux de données. Des questions qualitatives, ou mettant en jeu des calculs simples, permettent de discuter l'applicabilité des méthodes établies aux contraintes réelles de la sécurité aérienne.

Analyse globale des résultats

La structure de base d'une requête SQL est maitrisée par la majorité des candidats. La syntaxe du langage Python est remarquablement mieux acquise que lors de l'épreuve du concours 2015. La présentation des copies, en particulier des algorithmes, est bonne. Cependant, de trop nombreux candidats ne font pas encore l'effort de s'approprier vraiment le sujet, afin d'en maitriser les structures des données à manipuler. Enfin, les candidats ont pratiquement tous abordés les questions non algorithmiques autour du stockage ou de l'évaluation des paramètres en fin du sujet.

Commentaires sur les réponses apportées et conseils aux futurs candidats

Partie I

La fonction d'agrégation `COUNT` doit être connue. De plus, il faut avoir conscience que seuls des extraits de tableaux sont donnés, ainsi pour la question **I.B** il faut obligatoirement réaliser une jointure entre les tables `vol` et `aeroport` : CDG et ORY ne sont à priori pas les seuls aéroports parisiens. La requête demandée en **I.D** n'a pratiquement jamais été correctement écrite.

Partie II

Le jury tient à rappeler que la fonction `print` permet d'afficher une valeur, et ne sert à rien d'autre. Cette fonction apparait dans de nombreuses copies alors que le sujet ne demande jamais de procéder à un affichage. Il y a bien souvent confusion entre `print` et l'instruction `return`.

Le parcours des éléments d'un tableau, ou liste de liste, grâce à une double boucle imbriquée, est globalement maitrisé. Cependant, la question **II.B.2**, pour laquelle certains éléments du tableau seulement sont consultés, a souvent posé problème. Il est impératif pour ce type de question de s'approprier les structures des données à manipuler : il parait essentiel de schématiser au brouillon les structures, ou de réfléchir sur de petits exemples, pour mener à bien la gestion des indices de parcours du tableau.

Une minorité des candidats a su estimer le nombre de régulations possibles pour n vol. Le manque de recherche au brouillon, ne serait-ce que par le calcul des quelques premiers éléments de la récurrence, est ici aussi probablement en cause.

La question **II.C.1**, de calcul du cout d'un sommet, a été source d'erreur : on pouvait certes vérifier que le sommet dont on calcule le cout n'était pas supprimé, mais il était surtout primordial de consulter l'état de chaque sommet adjacent avant d'ajouter le conflit correspondant.

La question **II.D** sur l'algorithme de recuit simulé a été parfois abordée ; elle a été traitée de manière excellente par quelques candidats.

Partie III

Le calcul de l'instant t_c du *closest point of approach* (CPA) n'a pratiquement jamais été correctement réalisé. Pourtant, l'écriture d'un schéma simple peut permettre d'établir une relation entre t_c et le projeté orthogonal du vecteur position sur le vecteur vitesse. Néanmoins, il est intéressant de voir que de nombreux candidats n'ayant pas déterminé l'expression de t_c ont quand même traité le cas de l'absence de collision possible dans la fonction `calculer_CPA`.

Les questions en fin de partie III autour de l'évaluation des paramètres du système de détection anti-collision ont été traitées plutôt correctement par la majorité des candidats.

Conclusion

Le jury encourage les candidats à redoubler d'efforts sur les questions difficiles de ce type de problème, notamment celles mettant en jeu plusieurs données de structures différentes (tableaux, listes) devant être parcourues concomitamment. De même, il faut s'entrainer à écrire des requêtes SQL dont la complexité va au-delà de la requête de base. Ces compétences ne peuvent évidemment qu'être le fruit d'un travail régulier et assidu au cours de la préparation du concours, et les efforts évoqués deviennent de plus en plus faciles, voire ludiques, à force de pratique.