

# Rapport des correcteurs - Composition d'informatique 2h, filière PSI

## 1 Commentaires généraux

Le sujet portait sur la résolution d'un système de contraintes graphiques. Il s'agissait de reconstruire une petite image à partir de contraintes sur le placement des pixels par ligne et par colonne. La partie I portait sur les routines de vérification d'une solution. La partie II s'intéressait à la résolution par recherche exhaustive. La partie III portait sur les algorithmes de placement de blocs de pixels. Enfin, la partie IV portait sur la résolution par programmation dynamique.

L'énoncé commençait par un ensemble de consignes clairement détaillées, en particulier sur les opérations Python autorisées.

### 1.1 Présentation et lisibilité des réponses

Le jury rappelle que le code demandé est relu. Il est donc nécessaire de produire du code lisible. En particulier les noms de variables doivent être judicieux et le code doit être commenté. Par exemple, les noms de variable monolettre *i*, *j*, *x*, *y* sont à proscrire. Par ailleurs le code demandé ne dépasse *jamais* une demi-page.

### 1.2 Maîtrise du langage Python

Plusieurs copies montraient une incompréhension inquiétante des boucles `for`. Dans plusieurs copies, des candidats ont voulu modifier la valeur de *i* à l'intérieur d'une boucle `for i in range....`. C'est une pratique fortement déconseillée, dangereuse dans la pratique et en dehors de toute règle de bonnes pratiques, et ce quelque soit le langage utilisé. Cela a surtout été le signe d'une incompréhension totale du fonctionnement des boucles `for` dans chacune des copies concernées. Dans le même registre, le fonctionnement de l'opération `range` semble confus pour divers candidats. Si un programme appelle `for i in range(4)`, la variable *i* va prendre successivement les valeurs 0, 1, 2 et 3. Enfin, beaucoup de candidats *ignorent le concept de donnée mutable*, pourtant central dans le langage Python. C'était une source fréquente d'erreurs dans les copies, notamment dans la question 6.

### 1.3 Bilan général

Globalement, le niveau est meilleur que l'année précédente mais reste très lacunaire. Beaucoup de candidats maîtrisent mal l'algorithmique et le langage python et se montrent incapables d'écrire un code propre et lisible. En particulier, la notion de donnée *mutable* n'est presque jamais comprise, ce qui est inquiétant. Les enseignements devraient peut-être insister davantage sur ce point.

## 2 Commentaires détaillés

### Partie 1

**Question 1** Beaucoup de copies annoncent une complexité en  $O(n^2)$  sans définir *n*. Ce n'est pas parce que deux boucles sont imbriquées que la complexité est quadratique !

**Question 2** Quelques maladresses du type `if cases_lignes == cases_col: return true else: return false` au lieu de `return cases_lignes == cases_col`. Parfois `false` n'est pas retourné.

**Question 3** Le cas `len(cle) == 0` est très rarement traité.

**Question 4** Les exemples demandés sont corrects la plupart du temps. Par contre, les étudiants ont très rarement vu comment corriger la fonction.

## Partie 2

**Question 5** Si beaucoup de candidats trouvent  $k = n \text{ div } nc$ , la plupart expriment le modulo comme  $l = n - k.nc$ , sans voir qu'il s'agit de  $n \bmod nc$ . Souvent, la division euclidienne  $n = k.nc + l$  est simplement écrite sans aller plus loin alors qu'on demandait explicitement  $k$  et  $l$  *en fonction de  $n$* .

**Question 6** Certains tentent de construire toutes les combinaisons et ensuite de les tester alors qu'il vaut mieux les tester à la volée pour limiter l'empreinte mémoire. Beaucoup de récursions fausses. Oubli qu'une liste est *mutable*, qu'elle est donc passée par adresse et qu'il faut donc en stocker une copie. Ce dernier point montre une incompréhension inquiétante de Python et devrait être davantage travaillé pour les prochaines sessions. Certains réinventent la copie de liste, oubliant qu'une fonction prédéfinie est proposée dans l'énoncé.

**Question 7** Beaucoup d'explications confuses et de code non commenté alors que ça n'était pas demandé.

## Partie 3

**Question 8** Très peu de tests de bord, quelques réponses ne respectent pas la complexité. Certains remplissent une liste avec les conflits et retournent le premier élément alors qu'il suffisait de s'arrêter au premier.

**Question 9** Très peu de tests de bord, parfois la fonction `conflict` est utilisée alors que l'énoncé explicitait qu'il ne fallait pas s'en servir.

## Partie 4

**Question 10** Le cas `c-s-1<0` a été très peu traité, beaucoup de candidats se contentent de recopier les indications de l'énoncé. La plupart des candidats n'ont pas dépassé cette question.

**Questions 11 et 12** Questions très peu traitées, et très mal traitées.

**Question 13** Question bien traitée dans l'ensemble.

**Question 14** Question très rarement traitée.