ÉPREUVE ORALE D'INFORMATIQUE

1. Remarques générales

L'épreuve orale d'informatique, au choix avec la chimie, dure 50 minutes : 25 minutes de préparation suivies de 25 minutes d'exposé devant l'examinateur.

Pendant la 1ère partie de l'exposé (10 minutes) le candidat est amené à présenter la résolution d'un exercice tiré au sort et préparé pendant les 25 minutes préalables. Pendant la seconde moitié de l'exposé, il peut au choix présenter un projet préparé tout au long de son année en classe préparatoire (10 minutes plus un temps pour des questions), ou travailler sur un exercice non préparé proposé par l'examinateur.

- ✓ L'objectif des exercices proposés est de vérifier la capacité du candidat à pouvoir transformer un problème élémentaire en un algorithme, à déterminer les étapes permettant de mettre en œuvre cet algorithme et à identifier les fonctions et types de structures nécessaires à sa programmation. Le programme qui en résulte est écrit dans le langage de programmation Python.

  Les exercices se présentent sous forme de problèmes généraux ne faisant pas appel nécessairement à des notions mathématiques, physiques ou biologiques…

  Pendant l'exposé, le candidat est convié à présenter une solution pour résoudre le problème posé et à répondre à des questions qui peuvent être liées à la solution exposée, prolongements, variantes, efficacité de l'algorithme proposé… Les interventions de l'examinateur sont destinées à obtenir des précisions, corriger des erreurs ou de mauvaises démarches, elles ne sont jamais faites pour perturber le candidat.

  L'évaluation tient compte d'aspects strictement informatiques :
  - exactitude de l'algorithme présenté
  - maîtrise des concepts de programmation manipulés
  - efficacité du programme, prise en compte des cas particuliers

  Plus généralement d'autres qualités ont aussi été appréciées :
  - vivacité et rapidité suite aux remarques de l'examinateur
  - aptitude à défendre les solutions proposées
  - capacité à relier le problème à des problèmes plus généraux

- ✓ Dans la 2ème partie de l'exposé, le candidat présente un projet réalisé pendant l'année scolaire. Les candidats ont pu s'appuyer sur une présentation projetée sur leur ordinateur ou imprimée. Certains n'ont utilisé aucun support hormis le script du programme implémenté, à éviter.

  L'exposé sur le projet a notamment pour objectif de mettre en évidence la capacité du candidat à présenter clairement :
  - le sujet sur lequel il a travaillé
  - les hypothèses et limites fixées pour sa résolution
  - l'analyse effectuée et les solutions algorithmiques mises en œuvre pour le réaliser
  - les difficultés rencontrées et les perspectives pouvant être envisagées
  - éventuellement également des éléments de gestion de projet : répartition des tâches, problèmes organisationnels…

  L'évaluation tient compte de :
  - la qualité et la clarté de la présentation
  - l'ampleur du projet : difficulté du sujet, recherche bibliographique, nombre de méthodes implémentées, analyse des résultats …
  - une estimation de l'investissement apporté sur le projet (nombre de participants au projet, durée sur l'année, nombre de lignes de code …)
  - la qualité du code : organisation en fonctions, organisation des instructions conditionnelles, des itérations, utilisation d'"outils" python tels que le "slicing", les listes en compréhension, concision du code, etc …
  - la mise en œuvre de techniques de programmation avancée telles que la récursivité ou la programmation orientée objet, la présence d'une interface graphique
  - la qualité de la présentation du script, la présence de commentaires pertinents

Il apparaît que l'ensemble des candidats a choisi cette option en connaissance de cause, et à part quelques exceptions, ils ont les compétences permettant de résoudre les exercices, ce qui donne une moyenne de 14,47 à l'épreuve. Un certain nombre de candidats montre une très bonne maîtrise des concepts manipulés et une grande aisance à écrire un algorithme. Les examinateurs tiennent à souligner que même si certains candidats ont parfois été décontenancés par le sujet et n'ont pas trouvé forcément la bonne solution au départ, les interrogateurs ont tout de même pu évaluer leur capacité à rebondir aux remarques, leur réactivité pour rectifier le tir et proposer une solution au problème posé et leurs compétences en programmation.

La palette des projets présentés a été très variée et les sujets étaient intéressants, avec un peu plus de variété que l'année passée, en particulier la programmation d'un certain nombre de jeux. Nous avons cependant pu constater à nouveau une grande différence au niveau du temps consacré au projet au cours de l'année et des conditions de réalisation (nombre d'élèves impliqués, recherche biblio nécessaire, nombre de méthodes implémentées, interface graphique fournie ou non, etc.) et cela se traduit par de grosses différences dans le volume et la complexité du code présenté. Cependant, contrairement à l'année passée, nous n'avons pas ou très peu vu de projets basés pratiquement exclusivement sur la programmation d'une interface graphique qui ne permettent pas d'évaluer correctement les compétences en algorithmique.

2. Quelques points d'amélioration attendus

- Il est indispensable que le candidat présente le sujet de l'exercice dans son ensemble avant de rentrer dans le détail sans aucune introduction.
- De la même façon, avant de rentrer dans le détail, chaque question doit être introduite en présentant les résultats attendus, les données fournies et brièvement la méthode mise en œuvre.
- Il faut que les candidats prennent le temps de bien lire l'énoncé et de se poser les bonnes questions avant de se lancer dans sa résolution. Pour ceux qui l'ont fait spontanément, cela traduit une certaine prise de recul et une capacité de synthèse appréciable.
- Concernant l'utilisation de noms de variables "explicites", nous avons pu noter une nette amélioration par rapport à l'année passée, aussi bien dans les exercices présentés que dans les projets et c'est très appréciable. Il reste encore quelques élèves récalcitrants qui continuent à utiliser des noms de variables tels que M, N, ou x, y, z, a, b, m, n, qui ne favorisent pas une compréhension aisée et rapide des codes présentés.
- Au niveau programmation, quelques améliorations déjà indiquées l'année passée peuvent être apportées :
  - dans l'utilisation des instructions conditionnelles, en exploitant mieux la combinaison des *if … else* ou *elif* (en particulier dans les scripts des projets).
  - attention au vocabulaire utilisé, une instruction conditionnelle n'est pas une "boucle"…
  - privilégier l'utilisation de la méthode ***append*** plutôt que celle de l'opérateur + quand on veut ajouter un élément dans une liste. C'est beaucoup plus efficace.
  - on peut également faciliter l'écriture de certains programmes en utilisant l'instruction ***break.***
  - lorsqu'on demande de calculer un minimum ou un maximum, un certain nombre de candidats opte systématiquement pour un tri ou la construction d'une liste intermédiaire avec utilisation de la fonction *min* ou *max* de Python. C'est la plupart du temps une méthode trop coûteuse, il faut savoir rechercher un minimum ou un maximum avec un simple parcours d'une liste, ce qui permet par exemple de faire la somme des éléments en même temps.
- Les candidats semblent toujours peu à l'aise avec les chaînes de caractères et ont parfois été un peu perturbés par les exercices les mettant en œuvre. Ce problème est certainement dû au calendrier qui fait que les chaînes de caractères sont présentées et utilisées en début d'année et ont été un peu mises de côté après. Pour pallier cette difficulté, nous avons ajouté dans les énoncés le nécessitant un petit rappel sur les manipulations de base des chaînes de caractères.
- Certains candidats ne connaissent pas l'opérateur modulo "%" qui rend pourtant de nombreux services, tester si un nombre est pair par exemple…

- Peu savent utiliser également le *slicing* (découpage) de Python permettant d'extraire des sous-chaînes ou des sous-listes très facilement et rapidement. Ce pourrait être intéressant qu'il soit un peu plus travaillé pendant l'année.
- Pour la présentation du projet, peu de candidats présentent un "programme principal" avec l'enchaînement des fonctions à lancer pour pouvoir faire tourner le programme. Sans ce programme principal, il est difficile de savoir comment lancer son exécution et le tester.
- Essayer de faciliter la lecture du script en choisissant une impression adaptée avec un minimum d'instructions sur plusieurs lignes
- Les diapositives contiennent souvent beaucoup trop de texte, peu visible. Ne garder que des mots clés, les idées principales. Préférer une animation ou un dessin pour illustrer une méthode ou un algorithme plutôt qu'une capture d'écran avec du code.