ECOLE POLYTECHNIQUE

CONCOURS D'ADMISSION 2026

Epreuve orale d'Informatique, Filière PT

L'épreuve orale d'informatique a pour but de tester les connaissances des candidats sur le programme officiel d'informatique (première et seconde années) pour la filière PT.

L'épreuve se déroule de la façon suivante. Le candidat dispose d'une heure de préparation sur un sujet donné par l'examinateur. Une fois cette heure écoulée, le candidat présente pendant quarante-cinq minutes ses résultats devant l'examinateur qui va traiter les questions dans l'ordre du sujet en lui demandant des précisions et lui posant des questions ou des exercices complémentaires.

À aucun moment de l'épreuve un ordinateur n'est utilisé, il ne s'agit donc absolument pas d'une épreuve sur machine. Cependant, les examinateurs sont particulièrement attentifs à la capacité du candidat à écrire un programme syntaxiquement correct dans le langage Python. Ainsi, le candidat doit être capable pour chaque question demandant l'écriture d'un programme de proposer un code Python valide.

Les sujets comportaient systématiquement des questions de nature algorithmique. Certaines portaient directement sur des algorithmes classiques vus en cours (par exemple des algorithmes sur les listes — recherche, tri — ou de l'algorithmique des graphes) mais il y avait également des questions demandant aux candidats d'élaborer des algorithmes plus originaux.

La grande majorité des candidats a démontré une bonne connaissance du programme. Le jury n'a pas eu à déplorer de difficulté dans l'écriture de code Python syntaxiquement correct. Cette épreuve comportant une longue préparation (une heure), nous recommandons aux candidats d'en profiter pour bien rédiger en amont leurs réponses afin de ne pas se retrouver à devoir développer en direct une idée (même correcte) trop peu détaillée. Les questions sont traitées dans l'ordre par les examinateurs qui n'hésitent pas à passer le temps nécessaire avant de passer à la question suivante : il est donc illusoire d'espérer que l'on sautera une question non traitée par le candidat même si les examinateurs prennent par ailleurs soin d'aborder lors de l'interrogation orale le plus de questions possible que le candidat a eu le temps de préparer (ainsi, au début de l'interrogation, le candidat est invité à indiquer très brièvement les questions qu'il a abordées lors de la préparation).

Le grappillage n'est absolument pas récompensé : il ne sert donc à rien d'aller résoudre des questions faciles de la fin du sujet si l'on n'a pas convenablement traité le début du sujet. Comme pour une épreuve écrite, il est vivement recommandé aux candidats (trop peu le font) de ne pas hésiter à employer dans leurs algorithmes/programmes des fonctions annexes qui allègent leur code et simplifieront d'une part la présentation et d'autre part l'analyse de complexité. Il est également conseillé d'annoncer à l'oral l'idée générale de l'algorithme avant de se lancer dans une écriture détaillée du code. En particulier, il vaut mieux ne pas commencer la présentation de sa solution par l'écriture complète des fonctions auxiliaires. Il peut être utile de prendre des exemples. Le jury a systématiquement demandé de prouver la correction des algorithmes non triviaux donnés par les candidats. C'est un point qui a souvent posé des difficultés, et tout particulièrement pour les algorithmes récursifs. Par ailleurs, c'est souvent lors de cette étape que le candidat réalise de lui-même que son algorithme est faux : ainsi, nous ne saurions que trop recommander aux candidats d'anticiper la question de la correction pour ainsi détecter lors de la préparation d'éventuelles erreurs. Signalons que donner un algorithme, dont on sait pertinemment qu'il est faux, en espérant que les examinateurs ne s'en rendront pas compte n'est pas une bonne idée : la bonne attitude étant plutôt d'annoncer que l'on n'a pas su trouver une réponse correcte et d'expliquer les idées envisagées et leurs limites.

Le jury a également souvent demandé aux candidats de préciser la complexité de leurs programmes. C'est également un point qui a posé problème aux candidats (et qui, tout comme la capacité à établir la correction des algorithmes, permet de discriminer les candidats), alors que dans certains cas il n'y avait pas de réelles difficultés. Il est ici important de souligner que pour des algorithmes complexes (typiquement récursifs) nous ne demandions pas la complexité exacte mais simplement d'établir des relations de récurrence sans forcément les résoudre (sauf si cela est explicitement au programme).

On a vu cette année encore trop souvent des candidats faire des fautes de typage en particulier lors de la manipulation de listes de listes. Un trop grand nombre de candidats ne s'appuient pas sur un exemple pour expliquer un raisonnement complexe ou pour chercher au tableau la solution à une question non traitée. Trop de candidats utilisent des opérations comme la concaténation de deux listes, les slices de Python ou la concaténation de deux chaînes de caractères en les considérant comme des opérations unitaires. On déplore que de trop nombreux candidats soient complètement démunis lorsqu'il s'agit de donner une preuve formelle d'une propriété mathématique sur les objets manipulés, ce qui de facto les pénalise pour concevoir les algorithmes qui en découlent. Ces difficultés à raisonner mathématiquement sur un algorithme ont souvent eu pour conséquence la production d'algorithmes faux que le jury a systématiquement sanctionnée.