ORAL de MATHÉMATIQUES et ALGORITHMIQUE
(ex ORAL de MATHÉMATIQUES II)

Les futurs candidats trouveront dans ce rapport des remarques et des conseils qui pourraient leur être utiles pour leur futur passage. Nous ne pouvons que les inciter à consulter également le site de la Banque PT : http://www.banquept.fr/spip.php?article237 où ils trouveront les mémentos, disponible lors de l'oral, et les exercices types.

INTITULÉ

La durée de cet oral « Mathématiques et algorithmique » est de 1 heure, préparation incluse.

Il comporte deux exercices :

- l'un porte sur le programme de mathématiques de la filière PTSI/PT ;
- l'autre exercice porte sur les items 2, 3 et 5 du programme d'informatique.

OBJECTIFS

Le but d'une telle épreuve est d'abord de contrôler l'assimilation des connaissances des programmes de mathématiques et d'informatique (items 2, 3 et 5) de toute la filière (première et deuxième années). Il semble que certains candidats aient « *oublié* » ce qui a été vu en première année, voire les connaissances de base qui font partie du programme des classes du lycée (seconde, première, terminale).

Cette épreuve permet aussi d'examiner :

- la capacité d'initiative du candidat ;
- son aisance à exposer clairement ses idées ;
- sa réactivité dans un dialogue avec l'examinateur ;
- son aptitude à mettre en œuvre ses connaissances et son savoir-faire informatique pour résoudre un problème (par la réflexion et non par la mémorisation de solutions toutes faites) ;
- sa maîtrise des calculs nécessaires et du langage de programmation ;
- sa faculté à critiquer, éventuellement, les résultats obtenus et à changer de méthode en cas de besoin.

ORGANISATION

Cette dernière session s'est déroulée dans des conditions identiques aux sessions précédentes. Comme les autres années, elle a eu lieu au centre de Paris de « *Arts et Métiers ParisTech* », Boulevard de l'Hôpital à Paris (13^{e}).

Les candidats avaient donc deux exercices à résoudre, classiques et ne faisant appel à aucune astuce particulière :

- un exercice de mathématiques « *au tableau* », portant sur le programme de mathématiques des deux années de la filière PT (algèbre, analyse, géométrie et probabilités), c'est-à-dire sur les programmes PTSI et PT ;
- un exercice d'algorithmique « *sur machine* », portant sur le programme d'informatique : algorithmique (items 2 et 5) avec l'utilisation du langage Python et simulation numérique (item 3) avec l'utilisation de l'environnement de simulation numérique (les bibliothèques Numpy/Scipy/Matplotlib de Python ou l'atelier logiciel Scilab). Pour ce deuxième exercice, les candidats disposaient d'un ordinateur, sur lequel avaient été installés Python 3.4 et ses bibliothèques ainsi que Scilab 5.5 (aides incluses), et du mémento, rendu public bien avant l'oral. L'environnement de développement était IDLE, comme annoncé depuis un an, muni de l'extension IDLEx qui permet d'avoir les numéros de ligne. Quelques candidats ont avoué avoir

préparé l'oral avec Spyder ou Pyzo, ce qui est un peu surprenant. Nous ne pouvons que conseiller de se placer dans les conditions de passage de l'oral (IDLE, avec l'extension IDLEx éventuellement, + Mémento) tout au long des deux années de préparation. ***Aucun candidat n'a demandé à programmer en Scilab.***

COMMENTAIRES GÉNÉRAUX

Il est conseillé aux candidats de bien lire le sujet : certains perdent du temps à répondre à des questions qui ne sont pas posées.

Certains candidats semblent avoir oublié qu'ils sont à un oral d'un concours recrutant de futurs ingénieurs, c'est-à-dire de futurs cadres supérieurs : on attend d'eux rigueur, expression claire (à l'écrit comme à l'oral), autonomie, capacité d'écoute, réactivité et combativité. Un oral n'est pas un écrit où le candidat est debout au tableau ou assis devant un ordinateur.

Une attitude passive et sans réactions aux sollicitations et aux indications de l'examinateur a toujours une conséquence négative importante au niveau de la note finale ; de même, quand le candidat ne tient pas compte des remarques de l'examinateur et s'entête dans un raisonnement qui a très peu de chances d'aboutir.

COMMENTAIRES CONCERNANT L'EXERCICE DE MATHÉMATIQUES

Les erreurs, les comportements et les maladresses des candidats étant toujours les mêmes, ce rapport reprend l'essentiel des rapports précédents.

Certains candidats perdent du temps à répondre à des questions qui ne sont pas posées, en ayant lu le sujet trop vite. D'autres – ou les mêmes – donnent l'impression de « jouer la montre » en passant un temps important sur la (ou les) première(s) question(s), en général simple(s), et n'ont donc pas le temps nécessaire pour aborder les questions suivantes, plus intéressantes pour tester leurs connaissances. Cette attitude est évidemment sanctionnée.

Dans de nombreux exercices, un dessin ou un schéma est le bienvenu ; peu de candidats y pensent, y compris en géométrie où il est quasiment obligatoire même s'il n'est pas explicitement demandé dans le sujet.

Beaucoup de candidats ne connaissent pas leur cours. Il est demandé des définitions précises et des énoncés de théorèmes complets. Trop souvent, ils se contentent d'une définition floue ou d'une propriété en sus et place de la définition. Exemples : une valeur propre n'est pas seulement une racine du polynôme caractéristique ; une famille liée de vecteurs n'est pas seulement une famille à déterminant(s) nul(s).

De nombreux candidats, pour répondre à la question posée, cherchent à « replacer » une solution vue lors d'un exercice au cours de l'année. Les justifications ressemblent alors à des récitations. Il n'y a pas d'analyse du problème et, en conséquence, pas de réflexion sérieuse.

Même si une certaine technicité est indispensable, les examinateurs aimeraient surtout que les candidats comprennent ce qu'ils font et ce que signifient les notions utilisées, ce qui est loin d'être toujours le cas.

Enchaîner des calculs, voire simplement calculer, semble difficile pour certains. Des compétences qui devraient être acquises dans l'enseignement secondaire sont trop souvent inexistantes : par exemple, écrire l'équation d'une droite ou étudier une fonction très simple.

Trop de candidats manquent de logique et de bon sens. Prenons un exemple : pour étudier l'intersection de deux droites d'équations respectives $A = 0$ et $B = 0$, le candidat résout l'équation $A = B$ et, au vu du résultat exact qu'il trouve pour cette unique équation, ne s'étonne pas spontanément de la non-conformité du résultat avec la réalité.

Le nouveau programme de probabilités a été bien pris en compte par les candidats et ne pose pas de problèmes particuliers. Néanmoins, les savoir-faire utilisés demandent des justifications qui nécessitent la connaissance des définitions et théorèmes et la vérification de leurs hypothèses (par exemple lorsque la notion d'indépendance est en jeu). L'utilisation d'un arbre ou d'un graphe ne constitue pas une preuve.

Plus encore que d'autres années, l'absence de justifications dans la résolution des exercices a été constatée. Cela découle bien évidemment de la méconnaissance du cours, mais aussi du fait que certains candidats ne sont pas convaincus de la nécessité de telles justifications.

Parmi les lacunes rencontrées, on peut citer :

- établir des inégalités et utiliser des encadrements ;
- savoir s'il faut utiliser une condition nécessaire ou une condition suffisante ;
- démontrer qu'une application est bijective ;
- utiliser la formule du binôme ;
- calculer avec des nombres complexes ;
- connaître les fonctions trigonométriques et les formules élémentaires (addition, produit) ;
- calculer un produit matriciel, faire un changement de bases ;
- calculer un déterminant simple ;
- réduire une matrice ;
- calculer un équivalent ;
- calculer une dérivée et faire l'étude d'une fonction ;
- étudier la convergence d'une série numérique ou d'une intégrale impropre ;
- déterminer le rayon de convergence d'une série entière, sans utiliser systématiquement le critère de D'Alembert, hors programme ;
- effectuer un changement de variables dans un calcul de dérivées partielles ;
- reconnaître si une équation différentielle est linéaire ou non ;
- résoudre une équation différentielle linéaire ;
- écrire l'équation d'une droite ou d'un cercle dans le plan ;
- écrire l'équation d'une tangente en un point d'un arc paramétré ;
- écrire l'équation d'une droite, d'un plan ou d'une sphère dans l'espace ;
- différencier la probabilité de « $\mathcal{A}$ et $\mathcal{B}$ » de la probabilité de « $\mathcal{A}$ sachant $\mathcal{B}$ ».

De manière générale, aborder un exercice de géométrie est toujours aussi difficile pour un très grand nombre de candidats ; le programme actuel de la filière PT contient encore une partie non négligeable de géométrie. Par exemple, nombreux sont les candidats qui ne savent pas écrire l'équation d'une tangente ou d'une normale à une courbe plane, voire les confondent. La caractérisation des surfaces pose problème.

De même, les exercices comportant l'utilisation des nombres complexes semblent redoutables pour beaucoup.

En algèbre linéaire, pas loin de la moitié des candidats ignorent les définitions fondamentales (sous-espace vectoriel, application linéaire, *etc.*). L'algèbre linéaire est pourtant une partie très importante du programme de mathématiques et a de nombreuses utilisations pour un ingénieur.

En analyse, la notion d'équivalent, le calcul et l'utilisation d'un développement limité ne vont pas de soi.

Parmi les théorèmes les moins sus, citons ceux sur la continuité et la dérivabilité des fonctions définies par une intégrale, où l'hypothèse de domination est trop souvent oubliée, le théorème des probabilités totales avec la vérification du système complet d'événements.

Le chapitre sur les fonctions de plusieurs variables semble délaissé : montrer une continuité ou calculer des dérivées partielles en un point où la fonction est définie par une valeur semble hors de portée de beaucoup de candidats, certains ne voyant même pas qu'il y a un problème.

COMMENTAIRES CONCERNANT L'EXERCICE D'ALGORITHMIQUE

Une majorité de candidats a travaillé cette épreuve, ce qui explique une moyenne sur l'exercice d'algorithmique supérieure à celle de l'exercice de mathématiques. D'autres candidats, beaucoup moins bien préparés et heureusement minoritaires, semblaient découvrir l'environnement de développement sans faire la distinction console/éditeur et ne semblaient même pas connaître l'existence d'exercices type.

Cependant, l'équipe d'examinateurs a pu constater des points à améliorer :

- Dans l'utilisation de l'éditeur et de la console, nous avons pu constater des déséquilibres, dans un sens ou dans l'autre. Rappelons qu'il est fortement conseillé de tester les fonctions ou les portions de code écrites dans le programme au fur et à mesure de leur écriture, au besoin en faisant quelques tests simples dans la console. De manière générale, nous pouvons déplorer que les candidats ne lisent pas ou lisent mal les messages d'erreurs affichés lors de l'exécution d'instructions.
- Quelques candidats ont obtenu des solutions qui fonctionnaient sans mettre en œuvre une véritable démarche algorithmique ; il leur a été alors demandé de justifier leurs choix et d'expliquer comment ils pourraient améliorer leur code.
- Trop peu de candidats ont été capables d'extraire des données à partir d'un fichier ASCII ; les notions de répertoire de travail et d'arborescence de fichiers ne semblent pas acquises.
- Beaucoup trop de candidats ont essayé de « *tout faire à la main* ». Soulignons donc que les candidats peuvent utiliser les fonctions intrinsèques `max`, `min`, `sum`, `sorted`, `reversed`, *etc*. Ils ne devront montrer qu'ils connaissent et savent mettre en œuvre les algorithmes associés à ces fonctions, s'ils figurent dans le programme d'informatique, que si on leur demande expressément. De la même manière, on pourra utiliser « `a in L` » pour tester si `a` est un élément de la liste `L` ou tester si `a` est une sous-chaîne de la chaîne de caractères `L`.
- Dans le même ordre d'idée, les « *listes en compréhension* » ont été peu utilisées, sauf par les meilleurs candidats, alors qu'elles constituent souvent une alternative efficace aux boucles `for`.
- Pour les structures de boucles, les deux principaux défauts relevés sont : l'utilisation systématique, par quelques candidats heureusement peu nombreux, d'une boucle `while` dans toutes les situations, et, beaucoup plus répandue, l'indexation systématique des éléments d'une liste, même lorsque ce n'est pas nécessaire : « `for i in range(len(L)) :` » au lieu de « `for e in L :` ».
- La manipulation des listes pourrait être globalement améliorée. Trop peu de candidats ont su utiliser `L[d:f:p]` au lieu de perdre du temps à faire une boucle. On préférera `L.append(a)` à `L+=[a]`, en raison du surcoût informatique de la deuxième formulation (copie de la liste `L`) ; les candidats n'ont pas été pénalisés cette année sur ce point particulier. Rappelons que pour les chaînes de caractères, en revanche, seule l'écriture `chaine1+=chaine2` fonctionne, les chaînes de caractères étant des objets non modifiables, sans méthode `append` et `extend`.
- Trop de candidats ne font pas clairement la différence entre `True`/`False` (booléens), `"true"`/`"false"` (chaîne de caractères), et `1`/`0` (entiers). Plus généralement, le type `boolean` n'est pas toujours bien maîtrisé. Cette année, si `b` est le nom d'un booléen, l'écriture pléonastique « `if b == True :` » au lieu de « `if b :` », ainsi que l'écriture « `if b == False :` » au lieu de « `if not b :` », n'ont pas donné lieu à pénalité.
- La grande majorité des candidats sait définir une fonction. Cependant, quelques-uns confondent `return` et `print`. Plus généralement, il serait souhaitable que soit parfaitement assimilée la distinction entre ce que fait la fonction (en particulier si elle modifie des objets mutables passés en

argument), ce qu'affiche la fonction, et ce que renvoie la fonction. Exemples d'écriture fautives « classiques » sur les listes, révélant une mauvaise compréhension de cette distinction : « `L=L.append(a)` », « `L=L.reverse()` », et « `L=L.sort()` », qui font que `L` désigne au final `None`, l'objet de type `NoneType`. Un autre défaut relevé parfois : l'introduction dans la fonction d'un nom identique à celui de la fonction pour désigner un autre objet.

- Certains algorithmes comme les algorithmes d'Euclide, de dichotomie, les méthodes des trapèzes, d'Euler ou de Newton sont explicitement au programme d'informatique des classes PTSI ou PT. Un certain nombre de candidats n'en avaient aucune connaissance. Cela les a pénalisés de manière conséquente.
- Les instructions de conversion de type (`float`, `int`, `str`, `list`) ont été parfois utilisées à tort et à travers, ou au contraire n'étaient pas connues, ou confondues avec des fonctions mathématiques (`floor`, `ceil` des modules `math` ou `numpy`) ou la fonction intrinsèque `round`.
- Pour l'utilisation des modules, l'écriture « `from <module> import *` » est vivement déconseillée en raison des conflits de noms qui peuvent survenir. Nous préconisons « `from <module> import fct1,fct2,…` » pour utiliser quelques fonctions d'un module et « `import <module> as <alias>` » pour utiliser de nombreuses fonctions d'un module.

ANALYSE DES RÉSULTATS

1435 candidats présents, répartis en 9 jurys, ont passé cet oral.

Même si les notes publiées des candidats sont globales, nous donnons les résultats, exercice par exercice, à des fins statistiques :

	Général	*Mathématiques*	*Algorithmique*
Moyenne (sur 20)	**10,73**	*10,30*	*11,17*
Écart-type	**3,89**	*4,81*	*5,02*
Note minimale	**1**	*0*	*0*
Note maximale	**20**	*20*	*20*

La répartition des notes est la suivante :

		Mathématiques					
Algorithmique		[0,4[	[4,8[	[8,12[	[12,16[	[16,20]	*Total Algo*
	[0,4[	**1.0%**	**1.9%**	**1.6%**	**1.3%**	**0.5%**	***6.3%***
	[4,8[	**1.5%**	**5.4%**	**4.9%**	**4.5%**	**1.7%**	***17.9%***
	[8,12[	**1.8%**	**5.2%**	**7.2%**	**5.4%**	**2.9%**	***22.6%***
	[12,16[	**1.7%**	**5.2%**	**7.2%**	**10.5%**	**4.4%**	***28.9%***
	[16,20]	**1.5%**	**3.1%**	**5.4%**	**7.6%**	**6.8%**	***24.3%***
	Total Math	***7.5%***	***20.8%***	***26.2%***	***29.3%***	***16.2%***	[100.0%]

CONSEILS AUX FUTURS CANDIDATS :

Les conseils que l'on peut donner aux futurs candidats sont des conseils de « bon sens » que leur ont certainement déjà donnés leurs enseignants. Ce sont, bien sûr, toujours les mêmes :

- Travailler de manière régulière tout au long de l'année, que ce soit en mathématiques et en informatique. La pratique régulière des outils de programmation et de simulation numérique est très importante, dans les conditions de passage de l'oral, en s'habituant à utiliser l'aide des logiciels et le mémento.
- Étudier soigneusement son cours, connaître les définitions des notions rencontrées et les hypothèses précises d'application des théorèmes. Un énoncé de théorème n'est pas un texte vague que l'on peut utiliser comme incantation lors d'un exercice.
- À propos de chaque chapitre, faire un petit nombre d'exercices bien choisis et ne pas se contenter d'en lire une solution, aussi parfaite soit-elle. L'apprentissage des mathématiques ou de l'informatique passe obligatoirement par la pratique. Il faut souvent avoir « séché » sur une question pour en comprendre la solution.
- Ne pas faire d'impasse dans les programmes, y compris ceux de $1^{ère}$ année… Bien sûr, les compétences rencontrées lors de l'enseignement secondaire doivent être acquises.
- Lors de la résolution d'un exercice, réfléchir pour savoir quelles parties du cours sont concernées, quels théorèmes vont s'appliquer, quelles méthodes sont possibles : ne jamais se lancer sans réflexion dans un calcul.
- Apprendre à présenter ses calculs et ses résultats sur un tableau de manière ordonnée et propre : le tableau ne doit pas être un brouillon lisible seulement par son auteur. Ne pas hésiter à faire un dessin ou un schéma.
- S'entraîner à expliquer clairement d'une voix posée et audible le fil conducteur de ses calculs ou de sa démonstration lors d'une prestation orale, et cela sans « jouer la montre », c'est-à-dire en évitant de passer un temps important sur des questions très simples.
- S'entraîner au calcul : par exemple, utiliser les nombres complexes, réduire une matrice 3×3, calculer un développement limité ou une intégrale, résoudre une équation différentielle linéaire, donner l'équation d'une droite (d'un plan) passant par deux (trois) points...
- Après avoir obtenu un résultat, avoir un minimum d'esprit critique pour ne pas l'accepter s'il semble absurde ou impossible. C'est une qualité importante pour un futur ingénieur.