Le sujet proposé aux candidats pour l'épreuve d'option d'Informatique du Concours Commun Polytechnique était dans l'esprit des sujets proposés lors des sessions précédentes en intégrant une partie liée à l'algorithmique impérative et à la programmation en Python enseignées en Informatique pour tous.

Ce sujet couvre la plupart des aspects du programme de l'option Informatique : modélisation et résolution en logique des propositions, automates et théorie des langages, algorithmique fonctionnelle récursive sur les listes et les arbres avec formalisation et preuve des algorithmes, calcul de complexité. Il couvre également les aspects essentiels de l'Informatique pour tous : algorithmique impérative itérative et récursive avec formalisation et preuve des algorithmes, calcul de complexité. Le sujet s'attache à faire le lien entre les deux formes d'algorithmique fonctionnelle et impérative en demandant de traduire d'une forme à l'autre un algorithme donné. Le sujet s'attache également à distinguer la formalisation des algorithmes et leur implantation dans un langage donné en s'appuyant sur la différence de représentation des indices de séquence et de tableau en Python.

Les correcteurs ont noté une baisse surprenante dans la taille et la qualité des copies rédigées par les candidats. En effet, après étude du sujet, en se basant sur leurs expériences des années précédentes, les correcteurs estimaient que les copies seraient plus longues et les notes plus élevées. Rappelons que la durée de l'épreuve d'option Informatique du Concours Commun Polytechnique a augmenté d'un tiers (de 3h à 4h) pour cette session. Rappelons également que le volume global des enseignements de la discipline Informatique a également augmenté d'un tiers (approximativement de 160 h à 224 h selon le site de l'Union des Professeurs de classes préparatoires Scientifiques avec une hypothèse de 32 semaines annuelles) et que le programme d'Informatique pour tous a été redéfini dans un esprit similaire à celui de l'option Informatique. Pour les correcteurs, il était donc raisonnable d'espérer que la qualité des copies serait supérieure car les candidats devraient avoir acquis une plus grande maîtrise de la discipline. Des questions correspondant à l'Informatique pour tous (programmation impérative en Python) ont été introduites mais celles-ci étaient de difficultés raisonnables et dans un esprit similaire aux questions d'option Informatique et ne devaient pas poser de problèmes spécifiques aux candidats.

Les années précédentes, la taille moyenne des copies était de l'ordre de trois feuilles doubles et demi. Les correcteurs espéraient donc une taille moyenne de l'ordre de cinq feuilles doubles. Malgré cette évolution positive des enseignements et du concours, la taille moyenne des copies a été de l'ordre de deux feuilles doubles et demi, soit la moitié de la taille espérée par les correcteurs. La difficulté de l'épreuve était pourtant similaire aux années précédentes. Le sujet était un peu long par rapport aux capacités des candidats des sessions précédentes. Il s'est révélé beaucoup trop long par rapport à l'évolution des capacités des candidats issus des réformes successives des collèges, lycées et Classes Préparatoires aux Grandes Ecoles. La moyenne et l'écart type sont respectivement de 8,16 et 2,89.
Outre la taille réduite des copies, les correcteurs ont noté une baisse de qualité dans le traitement rigoureux de l'informatique. Ils souhaitent rappeler d'une part que cet aspect est essentiel dans la

discipline et d'autre part que les programmes de l'Informatique pour tous et de l'option Informatique lui accordent une juste place et que celle-ci doit être maîtrisée par les candidats.

De la même manière que la modélisation Mathématique a joué un rôle essentiel dans la compréhension du monde réel et dans le développement des métiers de l'ingénieur, le traitement Mathématique de l'Informatique est nécessaire à son développement. Les problèmes traités par l'Informatique sont discrets. Il n'est pas possible de s'appuyer sur des caractéristiques de continuité et de dérivabilité, comme dans de nombreux métiers de l'ingénieur, pour déduire la correction d'un système depuis un nombre fini de tests. La preuve, manuelle ou automatique, est la seule solution pour vérifier la correction d'un programme. La programmation n'est que l'étape finale d'un processus de spécification et de conception dans lequel la modélisation mathématique est fondamentale. Les candidats doivent donc maîtriser les techniques de modélisation et de preuve pour l'Informatique qui constituent la différence entre les métiers d'ingénieur et de technicien. L'Informatique joue un rôle important dans le développement de la plupart des systèmes, tous les futurs ingénieurs doivent avoir une vision rigoureuse de cette discipline pour interagir efficacement avec les spécialistes. Cette vision doit être acquise dès leur initiation à l'Informatique en Classes Préparatoires aux Grandes Ecoles. Le programme a donné une place à cet aspect que les candidats doivent maîtriser.

Les copies étaient en grande majorité bien présentées et rédigées. Certains candidats persistent à ne faire aucun effort de présentation malgré la prise en compte de cet aspect dans la notation et oublient de numéroter les pages de leurs copies.

Partie I : Logique et calcul des propositions

Il s'agit de modéliser un problème décrit en langue naturelle, puis de résoudre le modèle obtenu. Ce type d'exercice est classique dans l'épreuve d'option Informatique du Concours Commun Polytechnique depuis sa création.

Cette partie a été plutôt bien traitée par les candidats. La modélisation simple (Q I.1 et Q I.2) n'a posé aucun problème (88 % de succès). La modélisation plus complexe de la réponse partielle (Q I.4) a permis de départager les candidats (68 % de succès). La qualité de la résolution connait une baisse significative par rapport aux années précédentes. L'approche syntaxique (formule de De Morgan) a été bien traitée (60 % de succès). Certains candidats connaissent mal la définition de l'implication et ne détaillent pas leur résolution. L'approche sémantique (table de vérité) a connu moins de succès (44 %). Ceci est en partie lié à la modélisation plus difficile même si le barème avait pris en compte ce risque. Quelques candidats n'utilisent pas la technique demandée, voire échangent les deux techniques. Certains candidats tentent encore une résolution informelle en langage naturel.

Cette partie est importante dans tous les métiers d'ingénieur pour la modélisation des exigences et l'étude de leur complétude et cohérence. Il est essentiel que les candidats puissent modéliser et résoudre rapidement des problèmes décrits en langage naturel. Ils doivent étudier à la fois des problèmes simples et plus complexes et être capable de faire des hypothèses supplémentaires pour expliquer si le problème leur semble ambigu.

Partie II : Automates et langages

Il s'agit d'étudier un opérateur de manipulation de langages et d'automates : le calcul de la racine carré. Ce type d'exercice est classique dans l'épreuve d'option Informatique du Concours Commun Polytechnique depuis sa création. La définition pour les automates de l'opérateur étudié était plus complexe que ceux considérés dans les sessions précédentes du concours. Ce n'était pas le cas de la définition pour le langage. De plus, la compréhension de la définition n'était réellement nécessaire que pour les questions II.5 et II.6 (11 % de succès).

Cette partie a été traitée convenablement par les candidats avec une baisse de qualité significative dans la partie preuve. Cette baisse est malheureusement constante depuis plusieurs années. Certains candidats dont la solution est correcte structurent mal leur raisonnement en ne distinguant pas explicitement le cas d'arrêt et le cas de poursuite, en n'explicitant pas l'utilisation des hypothèses.

La proposition d'une expression régulière correspond à un langage défini par un automate (Q II.1 et Q II.6) est bien traitée (62 % de succès pour Q II.1, 11 % de succès pour Q II.6 à cause de la difficulté de compréhension de la racine carré d'un automate dans Q II.5).

La preuve par récurrence sur la longueur d'un mot, ou par induction sur la structure d'un mot, pose des problèmes aux étudiants. La question Q II.2 est quasiment une question de cours qui est déjà apparue plusieurs fois dans les épreuves des sessions précédentes. Elle n'a été réussie qu'à 30 %, ce qui est faible par rapport à l'importance de ce type de preuve en Informatique. Il faut noter que le sujet contenait une erreur de frappe remplaçant une quantification "il existe" par un "quelque soit". Cette erreur figurait également dans les sujets de sessions précédentes et n'avait pas été signalée par les candidats à l'époque. Elle a, cette fois, été remarquée par quelques candidats mais n'a posé aucun problème pour les autres candidats ne l'ayant pas remarquée. Ceux-ci ont construit une preuve par équivalence contenant une erreur dans le sens suffisant. Le barème n'a pas considéré ceci comme une erreur car il s'agissait d'une faute de frappe. Les questions Q II.8 et Q II.9 n'ont été que peu traitées correctement (8 % de succès). De nombreux candidats ne construisent pas explicitement la récurrence/induction mais font appel à la notation « ... » pour indiquer qu'il faut appliquer autant de fois que possible la définition pour conclure. Ce n'est bien entendu pas acceptable. Il est nécessaire que les candidats maîtrisent la rédaction d'une preuve par récurrence/induction.

La manipulation des propriétés des expressions régulières pour transformer une expression en une autre expression équivalente plus adaptée à la réponse à la question Q II.3 n'est pas suffisamment maitrisée. Le calcul de la racine carré du langage est donc souvent erroné (35 % de succès). De nombreux candidats ne semblent pas faire une distinction claire entre les opérateurs de répétitions + et *. Ils utilisent souvent * à la place de + et ajoutent ainsi le mot vide au langage.

La manipulation formelle des langages sous la forme d'ensembles de mots n'est pas suffisamment maitrisée. De nombreux candidats concluent que la racine du carré d'un langage est égale au carré de la racine de ce langage alors qu'il ne s'agit que d'une inclusion (39 % de succès pour la question II.4).

Les candidats ont peu traité la question Q II.7 (12 % de succès) alors qu'il s'agissait quasiment d'une question de cours ne demandant pas une compréhension finie de la définition de l'opérateur racine d'un automate.

Les candidats ont peu traité la question Q II.10 (12 %). Il s'agit de la dernière question qui s'appuie sur les questions précédentes peu traitées par les candidats, ce qui explique ce résultat même s'ils pouvaient exploiter les résultats de ces questions sans les avoir traitées.

Partie III : algorithmique et programmation

Certains candidats persistent à ne pas respecter les consignes de programmation en langage CaML. Le respect des consignes est un point important du métier d'ingénieur qui doit être compris et acquis dès le début de leur formation. Cela concerne les éléments du langage mais aussi les contraintes de parcours unique des structures de données pour maîtriser la complexité des algorithmes.

Exercice : le tri à bulles

Cet exercice s'appuie sur une implantation en Python de l'algorithmique impératif itératif du tri à bulles. Il consiste à étudier les caractéristiques de cette implantation (exécution sur un exemple, preuve de correction, terminaison, complexité) puis à programmer un algorithme similaire en CaML sous une forme fonctionnelle récursive. Cet exercice permet de manipuler les structures de liste en CaML et de tableau en Python.

Les candidats n'ont rencontré aucune difficulté pour implémenter un algorithme simple fonctionnel récursif en CaML (Q III.1, 80 % de succès) et à interpréter le programme Python fourni (Q III.2, 79 % de succès). De nombreux candidats n'ont pas écrit explicitement ce que l'exécution du programme sur l'exemple particulier allait afficher mais ont décrit dans le cas général l'affichage réalisé par le programme. Ceci est inhabituel par rapport aux questions du même type pour le langage CaML qui sont toujours traitées correctement. Cette réponse a été considérée correcte même si ce n'était pas l'intension du sujet qui devra être formulé différemment.

Les candidats ne semblent pas habitués à réaliser une preuve de correction d'algorithmes itératifs à base d'invariants (Q III.3, 15 % de succès). La plupart ne déduisent même pas le résultat attendu de l'invariant en fin de boucle. La preuve ne contenait aucune difficulté. Il est nécessaire que tous les candidats sachent formaliser ainsi les arguments de correction. Il s'agit de la base de nombreuses méthodes formelles exploitées dans l'industrie des systèmes critiques (aéronautique, ferroviaire). Il faut noter que les candidats maîtrisent bien le lien entre le codage entre 0 et n-1 des indices de tableau en Python et la formalisation manipulant des indices entre 1 et n. Ceci est nécessaire car le changement de codage entre un algorithme et un langage de programmation est fréquent. Cette question s'est révélée discriminante. Certains candidats semblent avoir considéré que les invariants donnés par le sujet étaient supposés corrects alors qu'ils devaient prouver leur correction. Certains candidats se contentent de dire qu'il est évident qu'ils sont corrects alors qu'il leur est demandé d'expliciter la preuve de correction. Il est essentiel que les candidats comprennent qu'il leur est demandé de montrer qu'ils maitrisent des techniques de preuve sur des exemples simples car la durée de l'épreuve est limitée et qu'ils doivent donc construire explicitement la preuve même si celle-ci leur semble évidente.

La preuve de terminaison d'un algorithme itératif a été très bien traitée (Q III.4, 67 % de succès). Certains candidats ne se limitent pas à citer qu'il s'agit de boucles « pour » dont le calcul se termine par construction et argumentent de la terminaison d'une boucle « tant que » générale. Le calcul de complexité a été bien traité (Q III.5, 55 % de succès). Le sujet n'indiquait pas clairement quelle était

l'unité de mesure de la complexité : l'opération de comparaison, d'affectation, les deux, … Les candidats distinguent ou pas les meilleurs et pires cas selon la mesure choisie mais ils oublient souvent d'indiquer quelle mesure ils ont choisi.

De nombreux candidats n'ont pas abordé l'écriture fonctionnelle récursive en CaML de l'algorithme du tri à bulles (Q III.6, 32 % de succès). Cette question a donc été discriminante alors qu'elle ne devrait pas poser de difficulté car le passage d'une version itérative à une version récursive est automatisable. Ce point doit être compris par les candidats car le passage d'une forme de programmation à une autre est habituel. Les candidats de l'option Informatique doivent avoir une vision claire des relations entre les enseignements d'Informatique pour tous et d'option Informatique.

Problème : le tri par tas

Ce problème étudie la structure d'arbre en tas et son utilisation pour le tri récursif fonctionnel d'une liste en CaML et le tri en place d'un tableau en Python en représentant l'arbre avec un tableau. Ce problème est progressif introduisant les arbres binaires, puis les arbres parfaits et enfin les arbres en tas. Il étudie les caractéristiques d'une implémentation en CaML d'une fonction de construction d'un arbre parfait à partir d'une liste d'entiers (exécution sur un exemple, preuve de correction, terminaison, complexité). Les exercices de programmation en CaML sont de complexité croissante. Ce problème permet de manipuler les structures de liste et d'arbre en CaML et de tableau en Python.

Les candidats ont bien traité les questions liées à la profondeur d'un arbre binaire (Q III.8 et III.9, succès de 86 % et 65 %) et la numérotation des nœuds d'un arbre parfait (Q III.12, III.13, III.14, III.16, III.17, III.18 et III.19, succès de 75 %, 66 %, 68 %, 50 %, 49 %, 44 % et 49 %). Ces questions n'étaient pas complexes mais demandaient une bonne compréhension de la structure d'arbre. Certains candidats ont introduit des décalages de +/- 1 dans les expressions calculées sans justifications. De nombreux candidats ont effectué des récurrences correctes alors qu'ils n'en ont pas été capables dans la partie II. Il est essentiel qu'ils maîtrisent l'induction sur d'autres structures algébriques que les nombres entiers naturels comme les mots, les listes ou les arbres.

En ce qui concerne la programmation fonctionnelle récursive en CaML, la question élémentaire III.7 n'a posé aucun soucis aux candidats (succès de 94 %) ; ils ont bien traité la question III.11 (succès de 57 %) ; par contre, ceux-ci ont rencontré des difficultés dans les questions III.10, III.15 et III.20 (succès de 17 %, 19 % et 15 %). Outre les solutions partielles, la contrainte de parcours unique de la structure n'était en général pas satisfaite. La majorité des candidats n'a pas réussi à mettre en place un parcours en largeur pour la question III.10. Il est nécessaire que tous les candidats maitrisent à la fois le parcours des arbres en profondeur et en largeur. De nombreux candidats oublient des cas dans la question III.15. L'utilisation du filtrage en explicitant tous les cas au lieu d'une imbrication de conditionnelles aurait permis d'éviter ces oublis. Cette approche est préférable dans la plupart des cas. Dans le même esprit, les candidats devraient éviter l'utilisation des fonctions « hd » et « tl » ou l'écriture « let Nœud(…) = expression in » qui les conduisent à oublier des cas.

Les candidats n'ont quasiment pas traité les questions III.22 à III.25 (succès de 0 %, 0 %, 4 % et 0 %) consistant à étudier les caractéristiques de la fonction récursive CaML de construction d'un arbre parfait à partir d'une liste d'entiers ; ainsi que les questions III.26, III.27, III.32 et III.33 (succès de 8 %, 2 %, 0 % et 0 %) de programmation fonctionnelle récursive en CaML et les questions III.28 à III.31

(succès de 2 %, 1 %, 5 % et 1 %) de programmation impérative récursive et itérative en Python. Il semble que ceci soit en partie lié à la longueur de l'épreuve .

BILAN

Cette session du Concours Commun Polytechnique s'est adressée aux candidats issus des réformes successives des collèges, lycées et Classes Préparatoires aux Grandes Ecoles même si cet effet a été limité par la présence des étudiants en 5/2. Le sujet a pris en compte les évolutions du programme mais semble avoir sous-estimé l'évolution des capacités des candidats dont les vitesses de recherche de solutions et de rédaction semblent réduites par rapport aux sessions précédentes. Les copies ont ainsi été beaucoup plus courtes que ce qui avait été estimé par les correcteurs.