20. Travaux pratiques d'informatique

20.1. Introduction

L'épreuve de travaux pratiques d'informatique en MPI est une épreuve d'algorithmique et de programmation, avec utilisation d'un ordinateur, d'une durée de 3 heures. À partir d'un sujet imposé, elle demande de traiter sur machine des questions de programmation, d'effectuer des choix de modélisation et d'aborder certains aspects théoriques de l'informatique, tout en communiquant très fréquemment avec le jury.

L'objectif de l'épreuve est d'évaluer les capacités de programmation, la maîtrise des méthodes classiques du programme, les capacités de modélisation, d'abstraction et d'inventivité, ainsi que l'application de bonnes pratiques que l'on est en droit d'attendre de futurs ingénieures et ingénieurs.

Organisation de l'oral

Des ordinateurs configurés avec un environnement de travail adapté aux sujets demandés sont fournis par le concours pour cette épreuve. Pour la session 2025, l'environnement était une distribution GNU/Linux Ubuntu 22.04. Un passage vers une distribution Debian 13 est envisagé pour les sessions futures, sans que cela n'ait d'impact majeur sur l'environnement proposé.

Avant le début de l'épreuve, le jury laisse une dizaine de minutes aux candidats afin de prendre en mains librement cet environnement de travail, de bien choisir son éditeur, de vérifier qu'il sait interpréter ou compiler en OCaml et en C et de se familiariser avec la documentation qui est proposée avec Zeal pour les langages OCaml/C/SQL. Pendant ce temps, qui n'entre pas en compte dans l'évaluation, il est possible de poser des questions techniques sur l'environnement aux membres du jury, de s'échauffer en écrivant de petits programmes simples ou encore de préparer des fonctions utilitaires générales.

Les sujets sont fournis en version papier ainsi qu'en version numérique au format pdf. Les sujets sont généralement accompagnés de fichiers auxiliaires, comprenant notamment :

- des fichiers sources, complets ou à compléter selon les cas ;
- des fichiers de données à exploiter ;
- des scripts d'aide à la compilation.

Ces fichiers sont mis à disposition dans un dossier de travail spécifique à chaque candidat, dont l'accès est déverrouillé grâce à un code donné en début d'épreuve par le jury. Les candidats peuvent travailler directement dans ce dossier et il ne leur est demandé aucun transfert de fichier.

Les candidats sont responsables des sauvegardes de leur travail, qu'ils doivent effectuer à intervalles réguliers.

Les sujets sont organisés en une suite de questions, à traiter généralement de façon linéaire, sauf mention explicite du contraire. Les questions sont réparties en trois catégories :

- des questions de programmation à réaliser sur la machine ;
- des questions à préparer pour une présentation orale ;
- des résultats à reporter ou des questions de rédaction à réaliser sur un compte rendu.

Le compte rendu sert uniquement de support pendant l'interaction orale et est évalué par le jury directement pendant l'épreuve. On n'attend pas une rédaction détaillée comme on pourrait en trouver sur une copie à l'écrit, d'éventuelles imprécisions pouvant être levées par la discussion orale.

Comme indiqué dans le rapport précédent, ni le compte rendu, ni le code source ne font l'objet d'une évaluation en tant que telle après l'épreuve. Ceux-ci peuvent cependant être consultés a posteriori lorsque de nouveaux éléments y ont été apportés entre la dernière interaction avec le jury et la toute fin de l'épreuve. En effet, le jury effectue un dernier passage auprès des candidats un peu avant la fin de l'épreuve, mais ne peut interroger tous les candidats simultanément. Il arrive donc que certaines questions soient traitées sans que le jury n'ait eu le temps de les évaluer pendant la séance, auquel cas il consulte le compte rendu et les programmes sources pour finaliser son évaluation.

Le concours fournit toujours les feuilles et les brouillons utiles à l'épreuve. Les candidats doivent se munir du matériel d'écriture usuel (stylos, crayons, gomme, règle). Aucun autre matériel n'est autorisé. Pour certaines applications numériques demandées par les sujets, les candidats ont convenablement réussi à utiliser les fonctions de calcul des ordinateurs.

Les sujets proposés suivent tous le format des épreuves en vigueur depuis la session 2023. Ce format permet d'évaluer correctement le niveau des candidats et il n'est pas prévu d'évolution majeure pour la session 2026.

Programme des épreuves orales

Les sujets posés permettent d'évaluer l'ensemble des notions présentes dans les programmes d'informatique des classes de MP2I et de MPI. Chaque sujet porte sur une ou plusieurs parties spécifiques de ces programmes. L'ensemble de tous les sujets couvrait globalement l'intégralité du programme des deux années.

Les langages évalués sont C, OCaml et SQL. Les sujets peuvent utiliser, selon la pertinence par rapport au thème abordé, un seul, deux ou trois langages. Même si aucun sujet n'évalue uniquement le langage SQL, la partie consacrée à ce langage peut constituer une part importante du barème et il est impératif de ne pas faire une impasse totale sur ce langage. La très grande majorité des sujets évalue au minimum deux langages. Le langage à utiliser pour chaque question est quasiment tout le temps imposé, mais certaines parties ont parfois été laissées librement au choix des candidats.

Évaluation des épreuves orales

Les sujets proposés restent volontairement longs. Les candidats les plus efficaces ont pu tout de même aborder la quasi intégralité de leur contenu. Les sujets comportent tous des parties d'application immédiate du cours, assez guidées, destinées à évaluer les compétences de base en programmation et les connaissances de cours, ainsi que des parties d'ouverture favorisant la prise d'initiative et permettant d'évaluer l'autonomie et la capacité de prise de recul des candidats.

L'évaluation de l'épreuve ne consiste cependant pas exclusivement, et loin de là, en une mesure du plus grand nombre de questions traitées. Le jury a valorisé non seulement l'efficacité en termes de programmation, mais également l'intérêt porté à la discipline de programmation (compiler fréquemment, s'assurer que le code fonctionne, proposer spontanément des tests). L'épreuve de travaux pratiques d'informatique doit mener à un programme qui fonctionne effectivement. Le jury préfère des prestations avec un peu moins de questions traitées, mais dans lesquelles les programmes écrits avaient été correctement compilés et soigneusement testés.

20.2. Analyse globale des résultats

Les prestations des candidats sont dans leur très large majorité d'excellente facture, avec encore une hausse du niveau par rapport à la session 2024. Les exigences d'évaluation et de notation ont donc été revues à la hausse pour obtenir une moyenne et un écart-type cibles, de manière à pouvoir correctement classer les candidats. Il ne faut donc pas voir une baisse de niveau dans la diminution notable de la moyenne de l'épreuve entre la session 2025 et la session 2024.

Le niveau observé en programmation est très satisfaisant et la syntaxe des langages est généralement bien maîtrisée. Les connaissances de cours sont globalement complètes, même si le jury constate toujours quelques faiblesses récurrentes qui restent très discriminantes dans l'évaluation. Le jury insiste encore et toujours sur le fait que le contenu du cours de MP2I et de MPI doit être parfaitement connu, au point d'être restitué efficacement. Les sujets continuent de proposer des questions de cours, sans lequel il est vraiment difficile d'aborder correctement les questions de programmation qui suivent.

Les candidats semblent encore mieux préparés à l'épreuve que précédemment, en maîtrisant quasiment toujours les attendus et les modalités de l'épreuve. Dans la très grande majorité des cas, les compétences visées par les programmes de MP2I et MPI sont acquises, les aspects théoriques sont maîtrisés et la technique est manipulée avec efficacité. Le jury adresse, encore une fois, ses félicitations à l'ensemble des candidats et à leurs enseignants.

20.3. Commentaires sur les réponses apportées et conseils aux futurs candidats

Maitrise du cours

Le jury considère que la maîtrise du cours demeure un élément absolument essentiel : il encourage les candidats à bien apprendre le cours afin de traiter rapidement et efficacement les questions qui s'y rapportent. Les questions de cours restent discriminantes et peuvent être bloquantes pour traiter la suite du sujet. Dans de tels cas, le jury est contraint de faire des rappels de cours très importants, ce qui se ressent nécessairement sur la note obtenue.

Le jury observe toujours des confusions : les candidats confondent des algorithmes de recherche de motifs avec des algorithmes de compression, ou bien confondent les méthodes algorithmiques usuelles (diviser pour régner, programmation dynamique, etc.) sans en connaître vraiment le principe.

Le jury invite à faire preuve de précision dans l'utilisation des noms d'algorithmes au programme, dans une optique d'efficacité de la communication. La simple confusion de noms n'a cependant pas été sanctionnée quand il était clair que le candidat maîtrisait l'algorithme demandé et savait le décrire précisément. À contrario, redonner simplement en tant que mot-clé le nom d'un algorithme sans en connaître le principe n'a pas été valorisé.

Préparation technique

Le jury encourage de nouveau toutes les formations à proposer à leurs élèves, durant leur scolarité, un environnement de travail adapté au programme de MP2I et MPI, permettant au minimum d'aborder les points suivants :

– l'accès à un shell dans un terminal permettant d'invoquer les compilateurs mais également d'autres commandes, et de manière générale permettant de se familiariser avec le fonctionnement d'une ligne de commandes ;

– le fonctionnement d'un système POSIX, en particulier quant à la gestion des processus, des fils d'exécution, des flux standard et leur redirection, éventuellement avec des tubes ;
– la familiarisation avec des outils de compilation usuels (make ou dune). Leur usage fera toujours l'objet d'un rappel et aucune compétence spécifique n'est attendue, mais avoir déjà rencontré ces outils permet une plus grande efficacité.

Remarques concernant OCaml

Le jury maîtrise la bibliothèque standard du langage mais il n'exige pas que les candidats utilisent d'eux-mêmes des fonctions de haut niveau, par exemple celles provenant du module `List`. L'évaluation est par exemple indifférente à l'utilisation, qui n'est ni valorisée ni sanctionnée, de `List.fold_left` (et autres fonctions similaires). Le jury est surtout attentif à la clarté du code produit, à sa correction et à la maîtrise de ce code par les candidats.

Il est pertinent de réfléchir quelques instants à la manière de concevoir un code, pour éviter que le résultat soit inutilement compliqué et soit source d'erreurs difficiles à détecter et corriger.

Les candidats doivent savoir proposer un programme OCaml complet, qui compile dans son intégralité et pas uniquement ligne par ligne dans un REPL. Une aide à la compilation a systématiquement été fournie. Le jury accepte si nécessaire le double point-virgule (`;;`) pour séparer les phrases, même si ce dernier ne fait pas partie du langage à proprement parler ; il le propose aux candidats quand cela aide parfois à mieux cerner certaines erreurs de syntaxe difficiles à situer d'après le message du compilateur. Sur ce dernier point, le jury attire l'attention des candidats sur le fait que `;` est un opérateur binaire et non pas un terminateur d'instruction.

Les candidats doivent savoir identifier les erreurs de syntaxe dans leurs programmes, celles-ci se situent souvent bien avant la ligne à laquelle est indiquée l'erreur.

Remarques concernant C

La gestion de la mémoire est un aspect maîtrisé par de nombreux candidats, mais les oublis d'allocations avec `malloc` et surtout les oublis de libération avec `free` restent trop fréquents. Il est toujours pertinent de s'interroger sur la politique d'allocation de la mémoire, en particulier lorsque le sujet invite à programmer une petite API manipulant une structure de données, dont on se sert ensuite. Avoir une politique claire d'allocation *et de libération* est essentiel.

Lors de la préparation des candidats, il peut être utile d'indiquer comment compiler un programme avec `gcc -g -Wall -fsanitize=address` et d'interpréter la sortie en cas d'erreur de segmentation ou de libération oubliée. La compilation séparée a fait l'objet de rappels, mais le jury souligne que celle-ci doit être reconnue par les candidats qui sont donc supposés être un minimum familiers avec ce procédé.

Remarques concernant SQL

Le niveau de maîtrise est très hétérogène parmi les candidats interrogés. Le jury attire de nouveau l'attention sur le fait que plusieurs candidats semblent avoir fait l'impasse sur ce langage.

Les sujets proposent des requêtes de niveaux différents. Le jury s'attend à ce que les requêtes les plus faciles puissent être traitées par tous les candidats, notamment en ce qui concerne :

– l'usage des fonctions d'agrégation avec ou sans `GROUP BY` ;
– l'usage de jointures ;
– l'usage de `LIMIT` et `OFFSET`.

20.4. Conclusion

Le jury est toujours globalement très satisfait par les prestations des candidats. Il attire encore une fois l'attention sur l'importance de la compréhension et de la maîtrise du cours, sur les dangers de faire l'impasse sur le langage SQL, sur la nécessaire prise de recul quant aux notions abordées pendant l'année et sur l'intérêt d'une pratique très régulière sur machine tout au long du cursus.