INF280 : Préparation aux concours de programmation

Présentation du cours et du concours ACM-ICPC

Antoine Amarilli, Pierre Senellart

6 février 2018

Programmation compétitive

- Qu'est-ce que que c'est ?
  - Matériel : participants individuels ou en équipe
  - Entrée : spécification d'un problème de programmation

    Alice veut tondre sa pelouse rectangulaire avec un chemin aussi court que possible qui passe par chaque case et évite des obstacles...
  - Sortie : un programme qui répond au problème

Programmation compétitive

- Qu'est-ce que que c'est ?
  - Matériel : participants individuels ou en équipe
  - Entrée : spécification d'un problème de programmation

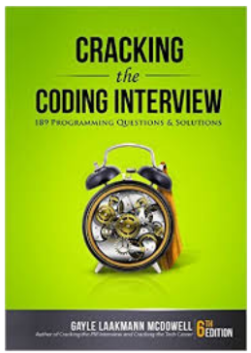
    Alice veut tondre sa pelouse rectangulaire avec un chemin aussi court que possible qui passe par chaque case et évite des obstacles…
  - Sortie : un programme qui répond au problème
- Pourquoi le faire ?
  - Pour le fun !
  - Pour apprendre à mieux programmer !
  - Pour la gloire ! (si on gagne…)
  - Pour trouver du travail !

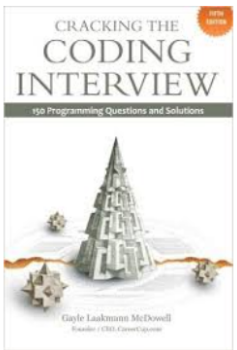
Programmation compétitive

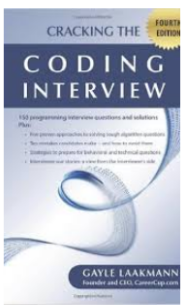
- Qu'est-ce que que c'est ?
  - Matériel : participants individuels ou en équipe
  - Entrée : spécification d'un problème de programmation

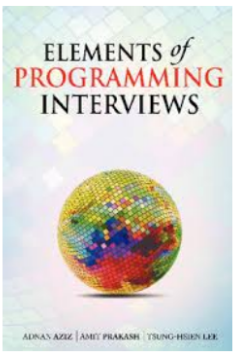
    Alice veut tondre sa pelouse rectangulaire avec un chemin aussi court que possible qui passe par chaque case et évite des obstacles…
  - Sortie : un programme qui répond au problème

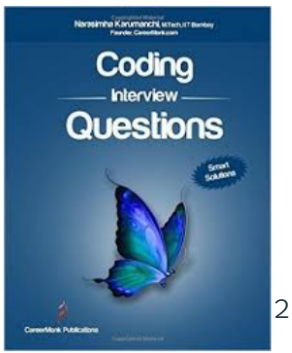
- Pourquoi le faire ?
  - Pour le fun !
  - Pour apprendre à mieux programmer !
  - Pour la gloire ! (si on gagne…)
  - Pour trouver du travail !











Cours INF280

- Préparation au concours de programmation ACM-ICPC

Cours INF280

- Préparation au concours de programmation ACM-ICPC
- Déroulement des séances :
  - Bref cours magistral sur un thème algorithmique
  - Trois problèmes d'annales ACM-ICPC donnés sur le site du cours :
    `https://a3nm.net/work/teaching/#y2016-inf280`
  - À tester sur un juge en ligne (UVa ou ICPC Live Archive)
  - À rendre sur Moodle au plus tard la veille de la séance suivante

Cours INF280

- Préparation au concours de programmation **ACM-ICPC**
- **Déroulement des séances :**
  - Bref **cours magistral** sur un thème algorithmique
  - Trois **problèmes** d'annales ACM-ICPC donnés sur le site du cours : `https://a3nm.net/work/teaching/#y2016-inf280`
  - À tester sur un **juge en ligne** (UVa ou ICPC Live Archive)
  - À rendre sur **Moodle** au plus tard **la veille** de la séance suivante
- **Évaluation :**
  - **Contrôle continu :** exercices à rendre
  - **Concours de programmation interne :** sélection pour ACM-ICPC

Cours INF280

- Préparation au concours de programmation **ACM-ICPC**
- **Déroulement des séances :**
  - Bref **cours magistral** sur un thème algorithmique
  - Trois **problèmes** d'annales ACM-ICPC donnés sur le site du cours : `https://a3nm.net/work/teaching/#y2016-inf280`
  - À tester sur un **juge en ligne** (UVa ou ICPC Live Archive)
  - À rendre sur **Moodle** au plus tard **la veille** de la séance suivante
- **Évaluation :**
  - **Contrôle continu :** exercices à rendre
  - **Concours de programmation interne :** sélection pour ACM-ICPC
- Langage : **C++** uniquement, i.e., C avec STL

Concours ACM-ICPC

- **ACM-ICPC :** ACM International Collegiate Programming Contest

Concours ACM-ICPC

- **ACM-ICPC :** ACM International Collegiate Programming Contest
- **ACM :** Association for Computing Machinery, principale **société savante** en informatique (conférences, prix, journaux, etc.)

Concours ACM-ICPC

- **ACM-ICPC :** ACM International Collegiate Programming Contest
- **ACM :** Association for Computing Machinery, principale **société savante** en informatique (conférences, prix, journaux, etc.)
- **Collegiate :** les candidats (étudiants) sont organisés en **équipe** (de 3 membres) qui viennent tous de la même université

Concours ACM-ICPC

- **ACM-ICPC :** ACM International Collegiate Programming Contest
- **ACM :** Association for Computing Machinery, principale **société savante** en informatique (conférences, prix, journaux, etc.)
- **Collegiate :** les candidats (étudiants) sont organisés en **équipe** (de 3 membres) qui viennent tous de la même université
- **International :** équipes du monde entier, compétition en **deux phases** (régionale et mondiale)

Concours ACM-ICPC

- **ACM-ICPC :** ACM International Collegiate Programming Contest
- **ACM :** Association for Computing Machinery, principale **société savante** en informatique (conférences, prix, journaux, etc.)
- **Collegiate :** les candidats (étudiants) sont organisés en **équipe** (de 3 membres) qui viennent tous de la même université
- **International :** équipes du monde entier, compétition en **deux phases** (régionale et mondiale)
- **SWERC :** Phase régionale de l'Europe du sud-ouest. Télécom ParisTech participe depuis 2013. À Télécom en **novembre 2017**.

  `http://swerc.up.pt/2014/reports/ranking.html` (13e et 33e sur 49)

  `http://swerc.up.pt/2015/reports/ranking.html` (14e et 45e sur 52)

  `http://swerc.up.pt/2016/reports/ranking.html` (20e et 36e sur 60)

  `https://swerc.eu/theme/results/official/public/` (30e et 42e sur 75)

Concours SWERC

- **Équipes de 3 candidats** pour chaque école ou université
- À Télécom : le **concours interne** pour INF280 (en fin de P4) sélectionne **deux équipes** formées des meilleurs candidats
- Les participants doivent être **inscrits pédagogiquement** à Télécom en novembre au moment du SWERC. (La plupart des 3e année au moment du concours interne donc non éligibles…)
- Les deux meilleurs équipes du SWERC participent à la **finale mondiale** en avril–juin de l'année suivante

Règles du concours SWERC

- Temps limité, généralement à **5 heures**
- Une **dizaine** de problèmes de programmation à résoudre
- **Un seul ordinateur** par équipe !
- Langages de programmation : **C, C++ ou Java**
- **Pas d'accès à Internet !**
- Accès aux **documentations des langages** (JavaDoc, doc STL)
- Seul matériel autorisé : **antisèche de 25 pages** imprimées A4

Détails :

- `https://swerc.eu/pages/regulations.html`
- `http://icpc.baylor.edu/worldfinals/rules`
- `http://icpc.baylor.edu/worldfinals/programming-environment`

Règles du concours interne

- Temps limité, généralement à 3 heures
- Environ six problèmes de programmation à résoudre
- Concours individuel
- Langage de programmation : C, C++ ou Java
- Pas d'accès à Internet, excepté le juge en ligne, la documentation des langages, et parfois Wikipédia.
- Antisèche de 25 pages autorisée

Date : le jeudi 28 juin 2018 de 13h30 à 16h30.

Format des problèmes

- Description en anglais d'un problème concret à résoudre
- Description du format des entrées et sorties
- Exemple d'entrée et de sortie correspondante fournie
- Le programme à implémenter prend sur son entrée standard (`stdin`, `cin`) une instance du problème et doit produire sur sa sortie standard (`stdout`, `cout`) la sortie correspondante

Soumettre un programme

- Soumission du code source sur une interface Web

Soumettre un programme

- Soumission du code source sur une interface Web
- Évalutation automatique sur des entrées secrètes.

Soumettre un programme

- Soumission du code source sur une interface Web
- Évalutation automatique sur des entrées secrètes.
- Compilation, édition de liens, exécution sur le serveur de test

Soumettre un programme

- Soumission du code source sur une **interface Web**
- Évalutation **automatique** sur des entrées secrètes.
- Compilation, édition de liens, exécution **sur le serveur de test**
- Temps d'exécution et mémoire disponible **limités** (e.g., quelques secondes, quelques méga-octets)

Soumettre un programme

- Soumission du code source sur une **interface Web**
- Évalutation **automatique** sur des entrées secrètes.
- Compilation, édition de liens, exécution **sur le serveur de test**
- Temps d'exécution et mémoire disponible **limités** (e.g., quelques secondes, quelques méga-octets)
- **Verdict** sur la soumission, les plus courants sont :
  - **Accepted :** La soumission passe les tests
  - **Time limit exceeded :** Trop lent (ou boucle infinie, bug...)
  - **Runtime error :** Erreur d'exécution (segfault, etc.)
  - **Wrong answer :** Résultats faux
  - **Presentation error :** (parfois) Mauvais formatage des résultats
  - **Memory limit exceeded**, **Compilation error**, etc.

Scores et stratégies

- **Objectif :** résoudre le plus de problèmes le plus vite possible

Scores et stratégies

- **Objectif :** résoudre le plus de problèmes le plus vite possible
- **Aucun point** si la soumission n'est pas parfaite (**Accepted**)
  (sauf dans le contrôle continu et concours interne)

Scores et stratégies

- **Objectif :** résoudre le plus de problèmes le plus vite possible
- **Aucun point** si la soumission n'est pas parfaite (**Accepted**)
  (sauf dans le contrôle continu et concours interne)
- **Temps de pénalité** pour les soumissions fausses
  (sauf dans le contrôle continu)

Scores et stratégies

- **Objectif :** résoudre le plus de problèmes le plus vite possible
- **Aucun point** si la soumission n'est pas parfaite (**Accepted**)
  (sauf dans le contrôle continu et concours interne)
- **Temps de pénalité** pour les soumissions fausses
  (sauf dans le contrôle continu)
- **Stratégie :**
  - attaquer les **problèmes faciles** d'abord
    (problèmes dans un ordre aléatoire, sauf dans le contrôle continu)
  - ne pas **rester bloqué** en cas de bug

Style de développement

- Le but est de produire vite du code qui marche

Style de développement

- Le but est de produire **vite** du code qui **marche**
- Inutile d'être **réutilisable**, et nécessité d'être **concis** :
  - Pas besoin de **commentaires**
  - **Un seul fichier** par problème
  - Pas de programmation **orientée objet** (sauf STL)
  - Pas de **vérification des entrées**, sécurité mémoire, etc.

Style de développement

- Le but est de produire **vite** du code qui **marche**
- Inutile d'être **réutilisable**, et nécessité d'être **concis** :
  - Pas besoin de **commentaires**
  - **Un seul fichier** par problème
  - Pas de programmation **orientée objet** (sauf STL)
  - Pas de **vérification des entrées**, sécurité mémoire, etc.
- Mais il ne faut pas faire d'**erreur** :
  - Noms de variables **brefs** mais **logiques**
  - Conserver l'**indentation**

Traiter un problème

- Souvent un **algorithme classique** caché derrière le programme (tri, graphe, géométrie, file de priorité, etc. ; cf le cours)
- Réfléchir **sur papier** à l'algorithme et au pseudo-code :
  Ne pas se précipiter pour coder !
- Tester sur l'**exemple fourni** et autres exemples simples
- Faire attention au **format de sortie** (retours à la ligne, etc.)
- Tester sur le juge en ligne, trouver les bugs s'il y en a...

Pour le contrôle continu...

- uDebug : AUTORISÉ
  - Vous fournissez un **fichier d'entrée**
  - Il fournit la **sortie correcte**
  - Permet de chercher les **bugs** dans votre programme

Pour le contrôle continu...

- uDebug : AUTORISÉ
  - Vous fournissez un fichier d'entrée
  - Il fournit la sortie correcte
  - Permet de chercher les bugs dans votre programme
- Forums, blogs, etc. : TOLÉRÉ
  - Si vous êtes bloqué, il y a souvent des éléments de solutions, idées (parfois fausses...) sur Internet
  - C'est OK aussi de discuter entre vous à propos du problème

Pour le contrôle continu...

- uDebug : AUTORISÉ
  - Vous fournissez un **fichier d'entrée**
  - Il fournit la **sortie correcte**
  - Permet de chercher les **bugs** dans votre programme
- Forums, blogs, etc. : TOLÉRÉ
  - Si vous êtes bloqué, il y a souvent des éléments de solutions, idées (parfois fausses...) sur Internet
  - C'est OK aussi de **discuter entre vous** à propos du problème
- Plagiat, copie de solutions en ligne : INTERDIT
  - **Interdit** de recopier le code d'une solution en ligne !
  - **Interdit** de recopier le code de vos camarades !
  - Tout **emprunt** d'un algorithme classique (Dijkstra, etc.) doit être **clairement indiqué** avec l'URL de sa source en commentaire

Exemple de résolution de problème

Traiter le problème “Identifying tea” :

- UVA 13012 `https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=866&page=show_problem&problem=4900`
- ACM 7211 `https://icpcarchive.ecs.baylor.edu/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=5223`

Crédits

- Version initiale de ces transparents par Pierre Senellart.
- Transparent 2 : Google Images, droit de citation.