

## Remarque

- Si vous avez une mauvaise note du partiel (moins de 6) ne faites pas ce projet. Concentrez-vous sur le cours, les TDs, les annales, pour bien réussir l'examen.

## Déroulement du projet

Ce projet s'effectue par binôme. Chaque binôme doit

- Choisir un algorithme dans la liste proposée et s'inscrire via la page web <http://www.doodle.com/b6wgr794a2fxbz97>. Les premiers venus auront bien sûr un choix plus large.
- Étudier l'algorithme en utilisant la littérature et l'Internet.
- Préparer un exposé de 25' sur l'algorithme.
- S'inscrire sur le web pour une soutenance orale. Les inscriptions seront ouvertes en décembre.
- Présenter l'exposé lors d'une soutenance orale de 30' en janvier 2010 (après le 15). Vous aurez à votre disposition un vidéo-projecteur, et si besoin est, un PC portable. Il n'y a pas de rapport à rendre.

## Contenu technique

Vous devez identifier et présenter clairement les éléments suivants :

- Le problème algorithmique
- L'algorithme qui résout ce problème
- Les grands principes utilisés dans cet algorithme
- Un exemple
- La preuve de correction
- Détails d'implémentation efficace, choix de structures de données
- Analyse de complexité
- Vos sources d'informations.

## Quelques conseils

- Avant de choisir définitivement un algorithme trouvez-le sur le web pour évaluer sa difficulté et son intérêt.
- Après avoir compris et étudié l'algorithme, prenez votre temps pour préparer la présentation.
- Soyez très pédagogiques. Il est possible qu'un membre du jury ne connaît pas l'algorithme, il doit le comprendre en suivant votre exposé.
- Il **n'est pas demandé** de programmer l'algorithme, vous pouvez le faire seulement si vous en avez envie.
- Si vous souhaitez, vous pouvez donner dans votre exposé autre information pertinente : histoire du problème et de l'algorithme, applications etc. C'est conseillé en cas où votre algorithme est facile.
- Si vous avez *vraiment* besoin d'un papier de recherche disponible sur un site payant d'éditeur, contactez E. Asarin.

## Les algorithmes

1. Composantes fortement connexes : algorithme de Tarjan. *strongly connected components*
2. Composantes fortement connexes : algorithme de Kosaraju. *strongly connected components*
3. Chemin le plus court pour tous les couples : algorithme de Johnson. *all pairs shortest path*
4. Chemin le plus court pour tous les couples : algorithme par carré de matrice en algèbre min-plus. *all pairs shortest path*
5. Algorithme de tri Radix-sort.
6. Problème 2SAT et un algorithme de solution.
7. Arbres 2-3-4 et algorithmes associés.
8. Arbres rouges-noirs et algorithmes associés.
9. Arbres Patricia et algorithmes associés. *Patricia trees*
10. Recherche de sous-chaine : algorithme de Rabin-Karp. *string-matching*
11. Distance de Levenshtein
12. Alignement de séquences : algorithme de Needleman-Wunsch *alignment*
13. Alignement de séquences : algorithme de Hirschberg *alignment*
14. Recherche de PGCD : algorithme binaire *binary GCD*.
15. Tableaux de suffixes et algorithmes associés. *suffix array*
16. Arbre de suffixes et algorithmes associés *suffix tree*.
17. Réduction transitive.
18. Tas binomial et son application à l'algorithme de Dijkstra.
19. Enveloppe convexe : algorithme de Graham. *convex hull*
20. Enveloppe convexe : algorithme de Jarvis. *convex hull*
21. Arbre couvrant minimum : algorithme de Borůvka *spanning tree*.
22. Union-find et son application à l'algorithme de Kruskal.
23. Code préfixé optimal : algorithme de Huffman.
24. Voyageur de commerce : méthode *branch and bound*.