Rapport du projet de programmation 2

Il s'agit de mettre en valeur la qualité de votre travail à l'aide d'un rapport montrant aux enseignants que vous avez compris et appliqué les règles de base de la programmation afin de produire un code lisible, modulaire, peu redondant, réutilisable et efficace.

Pour cela le rapport doit explicitement faire le point sur les fonctionnalités du logiciel (lister les objectifs atteints, lister ce qui ne fonctionne pas et expliquer - autant que possible - pourquoi). À cet effet, on pourra proposer des jeux de tests permettant de mettre en valeur la correction du logiciel (est-ce qu'il fait bien ce qu'on attend de lui ?). On pourra aussi s'appuyer sur les résultats d'un utilitaire comme *gcov* pour mesurer l'exhaustivité des tests réalisés ou sur l'utilitaire *valgrind* pour vérifier la qualité de la gestion mémoire dynamique.

Ensuite le rapport doit mettre en valeur le travail réalisé sans paraphraser le code, bien au contraire : il s'agit de rendre explicite ce que ne montre pas le code, de démontrer que le code produit a fait l'objet d'un travail réfléchi et même parfois minutieux. Par exemple, on pourra évoquer comment vous avez su résoudre un bug, comment vous avez su éviter/éliminer des redondances dans votre code, comment vous avez su contourner une difficulté technique ou encore expliquer pourquoi vous avez choisi un algorithme plutôt qu'un autre, pourquoi certaines pistes examinées voire réalisées ont été abandonnées.

Enfin il s'agit de démontrer l'efficacité du logiciel (combien de temps de calcul prend-il ?). On pourra encadrer de façon approximative la complexité théorique des algorithmes proposés et confronter cette analyse à la réalité (utilisation des utilitaires *time*, *gcov* et *gprof*). On pourra observer le temps de calcul pour des problèmes de taille croissante. Il s'agira de réaliser des tests significatifs nécessitant par exemple plusieurs secondes de calcul.

Pour la PSE, on présentera les temps obtenus sur la machine aragog du CREMI pour les cas Burma14, Ulysses16 et Ulysse22 de la bibliothèque TSPLIB[1]. On compilera le programme de façon optimisée (option –O3, avec un O comme Optimisation) et on présentera les temps d'exécution (temps user de la commande time) sur le modèle suivant :

Méthode	Burma14	Ulysses16	Ulysse22
PSE distance de retour			
PSE plus lointain non visité			
PSE élimination croisements			

Performances des algorithmes de PSE implémentés

Pour les méthodes approchées on traitera les cas d198 à d15112 en indiquant le résultat obtenu en plus du temps d'exécution, on pourra montrer les résultats sous forme de graphique.

Pour conclure on pourra parler des limites et extensions possibles des logiciels proposés. Enfin on présentera une bibliographie (livres, articles, sites web) brièvement commentée. On précisera en annexe les modifications apportées au projet par rapport à la dernière version présentée.
Vous préciserez de façon explicite l'origine de tout texte[2] ou toute portion de code emprunté (sur internet, par exemple) ou réalisé en collaboration avec tout autre trinôme. Il est évident que tout manque de sincérité sera lourdement sanctionné.

---

1 http://www2.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/

2 Directement dans le texte, par une note en bas de page comme celle-ci ou par une référence bibliographique entre crochet [1].