A. Pourquoi faire un compte rendu ?

Un compte rendu de TP est un document destiné :

- à faciliter la compréhension du code pour le correcteur
- à expliquer et justifier la démarche suivie lors de l'implémentation
- à mettre en valeur votre travail

B. Le fond

Ce document sera lu avant de parcourir le code, il doit donc comporter une analyse du problème, une conception détaillée de la solution, puis une explication de la réalisation, une description des tests effectués et enfin un retour sur expérience.

1. L'analyse du problème.

Le rapport comportera une introduction rapide au problème mais il est inutile de rappeler longuement le sujet. En effet, une ou deux phrases suffisent en général. Dans l'analyse du problème, vous précisez votre interprétation de certains points qui vous ont semblé flous et/ou importants dans le sujet. Vous complétez le cahier des charges de manière à ce que la solution que vous recherchez soit complètement spécifiée.

2. La conception de la solution.

Il faut amener progressivement le lecteur à comprendre la solution que vous proposez. Pour cela vous êtes amenés à présenter un découpage du problème en sous-problèmes et à donner des solutions pour résoudre ces sous-problèmes. Ces solutions peuvent être des algorithmes mais aussi des structures de données ou des choix pour restreindre l'espace des conditions d'utilisation. Pensez aussi à relier les sous-problèmes entre eux : comment les résultats d'un sous-problème sont utilisés par un autre sous-problème. Un schéma est souvent utile à ce niveau (il existe des approches plus formelles, MCD, UML par exemple, lorsque l'on aborde des problèmes plus gros).

Chaque algorithme, doit être décrit précisément et justifié : quels auraient été les autres choix possibles ? Quels sont les avantages et les inconvénients d'un tel choix ? Quel est la complexité (en temps, en mémoire)? Quels sont les impacts sur la fonctionnalité finale?

Cette partie ne doit pas contenir de code, elle doit pouvoir être utilisée pour une réalisation dans un autre langage impératif. Vous aurez ici à mentionner les structures de données mais en restant au niveau logique : les listes, les piles, les tableaux, tables de hachage etc... Sans détailler la manière dont seront implémentées ces structures de données.

Le plus important est que le texte se lise bien, l'enchaînement logique des idées amenant naturellement à la compréhension de la solution. C'est une décomposition top-down, c'est à dire que l'on part du problème global que l'on décompose et on décrit ensuite les solutions pour chacun des sous-problèmes.

3. Réalisation

Le rapport doit comporter les listings complets du code, mais en annexe (ils ne comptent pas dans le nombre de pages). Cette partie doit expliquer l'implémentation de la solution que vous avez proposée, structures de données détaillées, implémentation des algorithmes, éventuellement rôle de certaines variables clés, etc.

La décomposition en fonctions doit se rapprocher de la décomposition proposée dans la partie conception. Là encore les choix doivent être justifiés, éventuellement en relatant des problèmes apparus avec d'autre choix.

4. Test

C'est une phase importante du développement. Plus vous testerez votre implémentation, plus le lecteur sera convaincu de sa validité. Indiquez précisément quelles fonctionnalités sont testées, quelle est l'infrastructure de test utilisée (mise en place de test automatique, ou tests à la main). Il est, en général, assez précieux de pouvoir ré-exécuter tous les tests de manière à vérifier la compatibilité du code au cours de l'évolution du logiciel. Plus cette phase de test est prévue tôt dans le processus, plus elle sera efficace.

5. Retour d'expérience

Évitez les généralités du style « nous avons beaucoup appris lors de la réalisation de ce TP » ... Mentionnez éventuellement les problèmes rencontrés. Un bilan des efforts fournis (en terme de temps passé à la conception, à la réalisation et à la rédaction ou même en terme de recherche d'informations) doit également vous permettre de situer le rapport effort/retour et d'évaluer le coût de ce projet. C'est une étude critique de votre travail, plus elle est perspicace, plus le lecteur est convaincu que vous avez compris les problèmes.

C. La forme

La taille des différentes sections mentionnées ci-dessus dépendra évidemment du nombre de pages du document. En général un nombre de pages maximum est indiqué, il ne faut pas le dépasser. Il n'y a pas obligation de faire autant de page, mais lorsqu'on demande au plus 5 pages, il est assez mal vu de rendre un compte-rendu de 2 pages. Pour cela il est important de prévoir précisément le plan détaillé de votre document afin que vous puissiez inclure toutes les informations importantes, les informations secondaires n'étant rajoutées qu'après, selon la place disponible.

1. Page de garde

N'oubliez rien : titre TP, nom de l'étudiant (bien visible), groupe, date, puis le plan du document (table des matières).

2. Quelques remarques sur la forme

Pensez à faire des phrases complètes qui sont plus faciles à lire que les listes d'item. Organisez votre texte en paragraphes de plusieurs lignes. Évidemment, une grammaire et une orthographe correctes sont fondamentales, sans elles le lecteur ne peut pas se concentrer sur le fond.

L'utilisation de schéma facilite souvent la compréhension et aère le document. Un schéma doit comporter une légende qui permet au lecteur de le comprendre sans qu'il ait obligatoirement lu le texte associé (de manière générale, gardez à l'esprit que beaucoup de documents que vous rédigerez seront seulement parcourus rapidement). Un schéma doit normalement être référencé depuis le texte.

Utiliser une fonte différente (courrier par exemple) pour le code voire un screenshot localisé de votre IDE. Éviter l'utilisation du gras, préférez l'italique pour mettre en avant un mot. En ce qui concerne le code, une bonne indentation est indispensable. Les commentaires sont bons s'ils aident effectivement à mieux comprendre le code (et ne se contentent pas de paraphraser le code).

D. Conclusion

Les conseils ci-dessus n'ont aucun caractère obligatoire, mais ils sont en général respectés dans les bons rapports. Ils ne suffisent pas non plus à faire un bon rapport. La qualité essentielle d'un bon rapport est qu'il se lise naturellement et qu'il explique de manière claire les difficultés de compréhension du code.