NF16
TP4

Vérification de programmes

L'objectif de ce TP est de vérifier l'exécution d'un programme correspondant au TP3. Chaque binôme recevra un module objet ***tp3.o*** regroupant les fonctions développées. Le fichier ***tp3.h*** est le fichier de spécifications des fonctions développées. Les fonctions seront testées en appliquant la méthodologie présentée dans ce TP. Nous présentons tout d'abord les critères de qualité d'un programme puis la méthodologie à employer pour effectuer les tests.

1. Critères de qualité d'un programme

Un programme est dit :

- correct s'il répond aux exigences du problème initial ;
- robuste s'il réagit correctement en présence de données erronées ;
- fiable s'il est correct et robuste ;
- commenté si les commentaires aident le lecteur à comprendre ce que fait le programme;
- clair s'il contient des commentaires et des identificateurs significatifs.

Remarque : vous ne disposez que du fichier objet, vous ne pourrez donc pas apprécier la qualité des commentaires et la clarté du programme dans ce TP.

2. Méthodologie et travail demandé

La méthodologie pour effectuer les tests se décline en trois étapes :

- analyse de la spécification et extraction de cas de tests ;
- écriture des programmes ;
- exécution et analyse des tests.

2.1. Analyse de la spécification et extraction de cas de tests

A partir de la spécification, vous allez extraire les cas de tests permettant de vérifier les fonctions. Pour cela vous allez réaliser pour chaque fonction *F* présente dans le fichier *tp3.h* un tableau dont le format est le suivant :

Fonction *F*	**Valeur des v.e.**	**Comportement attendu**	**Valeur des v.s.**	**Commentaires**
annule_tache	1) liste des tâches 2) identifiant de la tâche	Liste vide : sans action. Message d'avertissement. Liste non vide et nom correct : parcourir la liste jusqu'à la tâche spécifiée par son nom, et annuler l'exécution de cette tâche par le processeur avant que ce dernier ne l'ait traitée	Pointeur vers la tâche annulée	Variables d'entrée : Pointeur sur liste des tâches *liste_tache* Identifiant *caract* 1ère catégorie (valeurs courantes) : 1) *liste_tache* non vide. 2) *caract* existe dans la liste. 2ème catégorie (valeurs particulières) : 1) *liste_tache* vide. 2) *caract* n'existe pas dans la liste. 3ème catégorie (valeurs aberrantes) : 1) *liste_tache* est un pointeur nul. 2) *caract* nul. Compléter avec les résultats des tests

avec :

1 v.e. : variables d'entrée de la fonction ;
2 v.s. : variables de sortie de la fonction.

Fonctions à tester :

1. task * cree_tache(char caract[MAX_NOM+1], int duree);
2. task * cree_liste(task *tache);
3. void affiche_liste(task *list_task);
4. int ajoute_tache(task *list_task, task *ptache);
5. task * annule_tache(task *list_task, char caract[MAX_NOM+1]);
6. task * execute_tache_FIFO(task *list_task);
7. task * depile_tache(task *list_task);
8. task * execute_tache_LIFO(task *list_task);
9. task * load_data(char * nom_fichier);
10. task * load_data2(char * nom_fichier);
11. task * insere_tache(task *list_task, task *ptache);
12. task * insere_tache_priorite(task *list_task, task *ptache);
13. task * fusion_listes(task *list_task1, task *list_task2);
14. int MAJ_priorite(task *list_task);

Pour chaque fonction à tester, identifier trois catégories de variables d'entrées :

- la première catégorie correspond à des valeurs de variables d'entrée usuelles qui permettent à la fonction de s'exécuter correctement. Par exemple, pour la fonction *annule_tache* les variables d'entrée sont une liste de tâches non vide et un identifiant de tâche existant dans la liste;

- la deuxième catégorie correspond à des valeurs de variables d'entrée correspondant à des cas particuliers. Par exemple, pour la fonction *annule_tache* la première variable d'entrée est une liste vide ou la deuxième variable d'entrée est un identifiant non existant dans le liste ;

- la troisième catégorie correspond à des valeurs de variables d'entrée qui permettent de mettre en défaut la fonction. Par exemple, pour la fonction *annule_tache* la première variable d'entrée est un pointeur nul, ou la deuxième v.e. est un identifiant nul.

Les deux premières catégories permettent de vérifier le caractère « correct » de la fonction *F*.
La troisième catégorie permet de vérifier le caractère « robuste » de la fonction *F*.

2.2. Ecriture du programme

L'objectif du programme principal ***test.c*** que vous devez mettre au point est de tester chaque fonction *F* avec ses trois catégories de variables d'entrée. Le programmeur a donc besoin de coder pour chaque fonction les trois catégories de variables qu'il a auparavant déterminées.
Le choix de la fonction à tester s'effectuera à l'aide d'un menu. Si une fonction ne fonctionne pas avec une catégorie de variables d'entrée, un message d'erreur sera affiché à l'écran et remplacera l'exécution de la fonction.

2.3. Exécution et analyse des tests

Compiler et exécuter le programme. Compléter les tableaux de tests. Conclure sur l'état de fonctionnement des fonctions. Vous préciserez leurs caractères (cf. partie Critères de qualité d'un programme).