INF1600 — TP3
Programmation en assembleur et débogage

Giovanni Beltrame giovanni.beltrame@polymtl.ca
Luca Gianoli luca-giovanni.gianoli@polymtl.ca

Polytechnique Montréal
Année académique 2016/2017

Remise

Voici les détails concernant la remise de ce travail pratique :

— **Méthode :** sur Moodle (une seule remise par groupe).

— **Échéance :** avant 12h00 ; le 5 novembre 2015 pour la section B1, le 12 novembre 2015 pour la section B2 ;

— **Format :** un seul fichier zip, dont le nom sera `<matricule1>-<matricule2>.zip`. Exemple : `0123456-9876543.zip`. L'archive doit contenir les fichiers `equals.s`, `transpose.s`, `diagonal.s`, `average.s` et `multiply.s`.

— **Langue écrite :** français.

— **Distribution :** les deux membres de l'équipe recevront la même note.

Barème

Contenu	Points du cours
`equals.s`	1
`transpose.s`	2
`diagonal.s`	2
`average.s`	2
`multiply.s`	3
Illisibilité du code (peu de commentaires, mauvaise structure...)	jusqu'à -1
Format de remise erroné (irrespect des noms de fichiers demandés, fichiers superflus, etc.)	jusqu'à -1
Retard	-0,025 par heure

Travail demandé

Vous êtes en charge d'implementer en assembleur cinq fonctions pour le calcul matriciel. Une implementation en C est vous donné comme référence.

Fichiers fournis

Les fichiers nécessaires à la réalisation du TP sont dans l'archive `inf1600_tp3.zip`, disponible sur Moodle.

Voici la description des fichiers :

— `Makefile` : le *makefile* utilisé pour compiler et nettoyer le projet ;

— `tp3.c` : programme de test qui utilise le fonctions de référence et celles en assembleur ;

— `equals.s` : fichier à compléter qui contient la fonction `matrix_equals_asm()` ;

— `transpose.s` : fichier à compléter qui contient la fonction `matrix_transpose_asm()` ;

— `diagonal.s` : fichier à compléter qui contient la fonction `matrix_diagonal_asm()` ;

— `average.s` : fichier à compléter qui contient la fonction `matrix_column_aver_asm()` ;

— `multiply.s` : fichier à compléter qui contient la fonction `matrix_multiply_asm()` ;

— `testmatrix2.dat` : fichier de donnés de test (matrice 2×2) ;

— `testmatrix3.dat` : fichier de donnés de test (matrice 3×3) ;

— `testmatrix4.dat` : fichier de donnés de test (matrice 4×4).

Vous devez compléter les fichiers `*.s` et les remettre dans un archive zip. Votre code doit passer chaque tests dans `tp3.c`.

Compilation et testing

Pour compiler le programme de test (`tp3`), il est suffisant de taper :

```
$ make
```

et pour l'executer avec le fichier de test `testmatrix2.dat` :

```
$ ./tp3 testmatrix2.dat
```

Débogage

Vous pouvez déboguer votre programme avec `gdb`. Vous pouvez aller voir votre source avec `gdb` et insérer des points d'arrêt dans le code assembleur.

Si vous avez une erreur de segmentation, `gdb` vous indiquera à quelle ligne se produit celle-ci et vous pourrez alors observer le contexte (valeurs des registres, des variables, de la mémoire, de la pile) et déterminer plus facilement la cause de cette erreur. Une erreur de segmentation arrive toujours lorsque le programme tente d'accéder à un emplacement mémoire invalide.

Une référence avec les commandes de `gdb` les plus utilisés se trouve ici :
`http://web.cecs.pdx.edu/~apt/cs577_2008/gdb.pdf`.

Pour executer `gdb`, tapez :

```
$ gdb tp3
```

et puis cliquez sur *File*, *Target settings* et écrivez le nom du fichier de test à utiliser dans *Arguments*.