

# IFT209 – Programmation système

## Université de Sherbrooke

### Devoir 2

Enseignant: Michael Blondin  
 Date de remise: mercredi 1<sup>er</sup> février 2023 à 23h59  
 À réaliser: en équipe de deux  
 Modalités: remettre en ligne sur **Turnin**  
 Pointage: sur 10 points + 0,75 point bonus

Le but de ce devoir est d'écrire un programme complet en langage d'assemblage afin de mettre en pratique l'arithmétique et les branchements sur un problème plus complexe que celui du laboratoire 1.

**Problème.** Rappelons qu'un nombre entier  $n \geq 2$  est *premier* si ses seuls diviseurs sont 1 et  $n$ . Par exemple, les nombres premiers dans l'intervalle  $[2, 30]$  sont:

2, 3, 5, 7, 11, 13, 17, 19, 23 et 29.

Un nombre est *palindromique* s'il se lit de la même façon dans les deux sens (en base 10). Par exemple, 2, 11 et 454 sont palindromiques, mais 6767 ne l'est pas.

Vous devez écrire un programme, en langage d'assemblage de l'architecture ARMv8, qui:

- lit deux entiers positifs  $a$  et  $b$ , représentés sur 64 bits, tels que  $2 \leq a \leq b$ ;
- affiche la quantité de nombres *premiers* dans l'intervalle  $[a, b] = \{a, a + 1, \dots, b\}$ ;
- affiche la quantité de nombres *premiers palindromiques* dans ce même intervalle.

**Tests.** Par exemple, dans un terminal, vous devriez obtenir:

|    |    |       |
|----|----|-------|
| 2  | 10 | 5000  |
| 30 | 20 | 15000 |
| 10 | 4  | 1085  |
| 5  | 1  | 13    |

où les deux premières lignes correspondent aux nombres entrés au clavier, et les **deux dernières lignes** correspondent aux sorties affichées (avec sauts de ligne).

#### Directives.

- Votre programme doit être obtenu en complétant le code partiel de la page suivante;
- Votre programme doit être remis dans un seul fichier nommé `devoir2.s`;
- Ne modifiez pas le point d'entrée et respectez le format des entrées et sorties;
- Ne figez pas les sorties manuellement dans le code afin de passer les tests;
- Supposez que les valeurs en entrée sont valides;
- Votre programme doit terminer dans un délai raisonnable, donc au plus 30 secondes sur une entrée où  $a \leq b \leq 75\,000$ . Si ce n'est pas le cas, le code sera tout de même corrigé, mais vous n'obtiendrez pas tous les points. Le temps d'exécution correspond à la somme des temps `user` et `sys` obtenus avec la commande «`echo a b | (time ./devoir2)`» sur une machine du département.

**Pointage.** Vous pouvez obtenir jusqu'à 10 points répartis ainsi:

- 1 point pour la lecture et l'affichage de deux nombres;
- 4 points pour le calcul de la quantité de nombres *premiers* (vous aurez au moins 2 points si vous passez les trois tests de la page précédente);
- 2 points pour le calcul de la quantité de nombres *premiers palindromiques* (vous aurez au moins 1 point si vous passez les trois tests de la page précédente);
- 1 point si votre code est bien indenté (étiquettes, codes d'opération, opérandes et commentaires);
- 2 points pour la qualité et la lisibilité du code (commentaires significatifs, usage des registres, organisation du code, pas de « code spaghetti », etc.)

★ **Bonus.** Pour obtenir des points bonus, votre programme doit être en mesure de gérer des nombres jusqu'à  $b = 1\,000\,000$  (nous n'entrerons pas de valeurs plus grandes). Dans ce cas, vous obtiendrez:

- 0,25 point bonus si votre programme affiche les bonnes quantités *et* termine en au plus 2 secondes sur l'intervalle  $[2, 1\,000\,000]$ ;
- 0,5 point bonus additionnel si votre programme affiche les bonnes quantités *et* termine en au plus 100 millisecondes sur l'intervalle  $[10\,000, 1\,000\,000]$ .

**Code partiel.**

```
.global main

// ...
main:
    /*
        code ici
    */

    mov     x0, 0
    bl      exit

/*
    données ici au besoin
*/
```