IFT209 – Programmation système

Université de Sherbrooke

Laboratoire 4

Enseignant:	Michael Blondin
Date de remise:	dimanche 18 février 2024 à 23h59
À réaliser:	en équipe de deux
Modalités:	remettre en ligne sur Turnin

Le but de ce laboratoire est de mettre en pratique la programmation structurée.

Problème. Rappelons qu'un *mode* d'un tableau d'entiers t est un élément x qui apparaît un nombre maximal de fois dans t. Par exemple, l'unique mode de $t = [42, 9000, 9000, 42, 9000]$ est $x = 9000$. En général, un tableau peut contenir plusieurs modes, mais dans ce laboratoire, les tableaux en contiendront toujours exactement un.

Vous devez écrire un programme, en langage d'assemblage de l'architecture ARMv8, qui calcule et affiche le mode d'un tableau d'entiers lu au clavier. La lecture et l'affichage sont déjà implémentés. De plus, un sous-programme `num_occ` est fourni. Celui-ci calcule le nombre de fois qu'un élément x apparaît dans un tableau t de taille n. Vous devez compléter le code du sous-programme `mode`.

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

```
5                       4               1
42 9000 9000 42 9000    1 2 1 3         8
9000                    1               8
```

où les deux premières lignes sont les entrées, et la troisième ligne est la sortie.

Rappels. Voici des rappels de la section 8.2 des notes de cours:

- Les sous-programmes émulent les fonctions;
- Les arguments d'un sous-programme sont passés dans l'ordre via x_0 à x_7;
- La valeur de retour d'un sous-programme (s'il y en a une) doit être stockée dans x_0;
- Les macros `SAVE` et `RESTORE` permettent respectivement de mettre de côté les valeurs des registres x_{19} à x_{30} de l'appelant, et de les restaurer.

Directives.

- Votre programme doit être obtenu en complétant le code partiel ci-dessous (disponible sur Turnin);
- Votre programme doit être remis dans un seul fichier nommé `labo4.s`;
- Ne modifiez pas le code existant; vous devez seulement implémenter « `mode:` »;
- Ne modifiez pas le point d'entrée ainsi que le format des entrées et sorties;
- Supposez que les valeurs en entrée sont valides; en particulier, le tableau ne sera pas vide.

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

- 5 points si votre programme passe les trois tests ci-dessus;
- 2,5 points si votre programme affiche la bonne sortie sur d'autres tests qui seront choisis à la correction;
- 2,5 points si vous utilisez le sous-programme `num_occ`.
- 0 point pour la lisibilité du code: indentation, commentaires et usage des registres (recommandé).

Code partiel.

```
.include "macros_save_restore.s"
.global main

.section ".rodata"                      //
fmtNum:     .asciz  "%ld"               // const char* fmtNum    = "%ld";
fmtSortie:  .asciz  "%ld\n"             // const char* fmtSortie = "%ld\n";
                                        //
.section ".bss"                         //
            .align  8                   //
temp:       .skip   8                   // long temp;
nombres:    .skip   100*8               // long nombres[100];
                                        //
.section ".text"                        //
/******************************************************************************
  Effet:  lit un tableau d'entiers de 64 bits,
          puis affiche un mode du tableau
  Usage:  x19 -- n    x20 -- x
******************************************************************************/
main:                                   // int main()
        adr     x0, nombres             // {
        bl      lire_tab                //
        mov     x19, x0                 //   long n = lire_tab(nombres);
                                        //
        adr     x0, nombres             //
        mov     x1, x19                 //
        bl      mode                    //
        mov     x20, x0                 //   long x = mode(nombres, n);
                                        //
        adr     x0, fmtSortie           //
        mov     x1, x20                 //
        bl      printf                  //   printf(fmtSortie, x);
                                        //
        mov     w0, 0                   //
        bl      exit                    //   return 0;
                                        // }
                                        //
/******************************************************************************
  Entrée: tableau d'entiers de 64 bits
  Effet:  lit une taille n, puis n entiers stockés dans le tableau
  Sortie: n
  Usage:  x19 -- tab    x20 -- n    x21 -- i
******************************************************************************/
lire_tab:                               // long lire_tab(long tab[])
        SAVE                            // {
        mov     x19, x0                 //
                                        //
        adr     x0, fmtNum              //
        adr     x1, temp                //
        bl      scanf                   //   scanf(fmtNum, &temp);
        ldr     x20, temp               //   long n = temp;
        mov     x21, x20                //   long i = n;
                                        //
lire_tab_boucle:                        //
        cbz     x21, lire_tab_ret       //   while (i != 0)
        adr     x0, fmtNum              //   {
        mov     x1, x19                 //
        bl      scanf                   //     scanf(fmtNum, tab);
                                        //
        add     x19, x19, 8             //     tab++;
        sub     x21, x21, 1             //     i--;
        b       lire_tab_boucle         //   }
                                        //
lire_tab_ret:                           //
        mov     x0, x20                 //
        RESTORE                         //
        ret                             //   return n;
                                        // }
                                        //
```

```
/*****************************************************************************
  Entrée: tableau d'entiers de 64 bits, taille n du tableau, entier x
  Sortie: nombre d'occurrences de x dans le tableau
  Usage:  x19 -- num_x    x20 -- *tab
*****************************************************************************/
num_occ:                                // long num_occ(long tab[], long n, long x)
        SAVE                            // {
        mov     x19, x0                 //
        mov     x0, 0                   //   long num_x = 0;
                                        //
num_occ_boucle:                         //
        cbz     x1, num_occ_ret         //   while (n != 0)
        ldr     x20, [x19], 8           //   {
        cmp     x20, x2                 //
        b.ne    num_occ_prochain        //     if (*tab == x)
        add     x0, x0, 1               //       num_x++;
num_occ_prochain:                       //
        sub     x1, x1, 1               //     n--;
        b       num_occ_boucle          //   }
                                        //
num_occ_ret:                            //
        RESTORE                         //
        ret                             //   return num_x;
                                        // }
                                        //
/*****************************************************************************
  Entrée: tableau d'entiers de 64 bits, taille n > 0 du tableau
  Sortie: un mode du tableau
  Usage:  ...
*****************************************************************************/
mode:                                   // long mode(long tab[], long n)
    /*                                  // {
        CODE À COMPLÉTER                //
                        */              // }
```