

IFT 1227 – Architecture des ordinateurs

Devoir 3

- À faire en groupe de **deux** étudiants.
- Remise : Le 16 mars 2017 avant minuit.
- Il y aura une pénalité de **10%** par jour de retard.

Programmation en assembleur MIPS

1. a) Écrire une fonction **initialiser** qui initialise un tableau de **n** éléments. Les valeurs des éléments et un nombre d'éléments **n** doivent être saisis par un utilisateur. Votre fonction doit enregistrer un tableau à partir de l'adresse 0x10040000 (région de **heap**).

Pour saisir les valeurs manuellement il faudra utiliser un appel système. Voici la procédure :

- Charger la valeur 5, le numéro de service de l'OS qui signifie « lire une valeur entière de l'entrée standard »
- Exécuter la commande **syscall**
- Comme résultat, on aura dans **\$v0** la valeur lue :

```
li    $v0, 5      # charger le numéro de service
syscall          # faire appel de ce service
# récupérer le résultat de $v0 dans $s0
add   $s0, $v0, $0
```

L'information concernant les appels système se trouve dans le simulateur Mars, bouton Help => onglet MIPS => onglet Syscalls

- b) Écrire une fonction **main** qui fait l'appel à votre fonction.
2. Un tas (monceau) est une structure de données représentant un arbre binaire presque complet (seulement la dernière couche du côté droite pourrait être incomplète) qui permet de retrouver la plus grande valeur ou la plus petite valeur de tous les éléments du tas de manière efficace.
Considérons un tas MIN. Dans ce type de tas les clés sont ordonnées selon la propriété de tas : la clé d'un nœud parent doit être plus petit que les clés de ses enfants. Comme un tas possède une structure très régulière, on l'implémente avec un tableau. Dans les prochaines 2 numéros vous allez implémenter un trie «Heap sort» qui permettra de trier un tableau des éléments en ordre décroissant.

- a) Concevoir les fonctions **swap**, **getLeftChildIndex** et **getRightChildIndex** ainsi que la fonction **main** qui fera les appels à ces trois fonctions en les testant.

La fonction **swap** échange le contenu de deux éléments du tableau avec les indices **i** et **j** respectivement. Les fonctions **getLeftChildIndex** et **getRightChildIndex** retournent les indices des enfants gauche et droite dans la structure de données TAS.

```
void swap(int i, int j)
{
    int temp = a[i];
    a[i] = a[j];
    a[j] = temp;
}
int getLeftChildIndex(int index)
{
    return 2 * index + 1;
}
int getRightChildIndex(int index)
{
    return 2 * index + 2;
}
```

- b) Rajouter les implémentations de deux fonctions

```
void fixHeap(int rootIndex, int lastIndex);
void sort(int [], int par_taille);
```

qui compléteront l'implémentation de l'algorithme **HeapSort**. Écrire une fonction **main** qui fait l'appel au **HeapSort** pour trier un tableau de **n** éléments. Incorporer votre code d'initialisation d'un tableau du numéro 1.a. Le code Java de **HeapSort**, de la fonction **main** en Java ainsi que les notes de cours Programmation II qui traitent ce sujet sont fournis sur le site du cours.

3. Coder une fonction récursive suivante pour générer une suite de nombres. Utiliser les appels **syscall** pour pouvoir saisir le paramètre **n** et imprimer les valeurs générées (fonction **main**).

```
int suite(n)
{
    if (n == 0) return 1;
    else
        return (suite(n-1) + 2*n) % 7;
}
```

Directives.

Dans toutes les questions, il ne faut pas utiliser ni la multiplication, ni la division, ni la virgule flottante. Il faut obligatoirement respecter toutes les conventions MIPS.

Imprimer une valeur stockée dans le registre \$t0 :

```
        li    $v0, 1    # service 1 imprime un entier
# charger la valeur à imprimer dans le registre $a0.
# Notre valeur se trouve dans $t0

        add   $a0, $t0, $zero
        syscall      # faire appel du service
```

Imprimer une chaîne se trouvant à l'adresse msg

```
la    $a0, msg
li    $v0, 4    # Print string (Code 4 de syscall)
syscall      # faire appel du service
```

À remettre : 4 fichiers *.asm pour les numéros 1, 2.a, 2.b et 3

Barème de correction :

Bonne fonctionnement : 15 pts chaque numéro => $15 * 4 = 60$

Utilisation des conventions : $5 * 4 = 20$

Utilisation correcte de la pile : 5

Respect des directives, propreté, présentation, commentaires significatifs dans les programmes, etc. : 15

Total : 100