Université Gustave Eiffel

Travaux Dirigés de Compilation n°3

Licence d'informatique

———

Assembleur : pile ; orientation little-endian ; multiplication et division

Le but de ce TD est de pratiquer des opérations dont le résultat dépend de l'orientation little-endian ou big-endian, d'allouer de la mémoire dans la pile, et d'utiliser les instructions de multiplication et de division.

▸ Exercice 1. *Orientation little-endian*

Dans le programme ***little_endian.asm****, à chaque commentaire dans le code, déduisez les valeurs des registres* ***rbx*** *et* ***r14****.*

```
; little_endian.asm
section .data
seq_numbers: dw 1, 2, 6, 3, 4, 22, 10, 0
section .text
global _start
extern show_registers
_start:
    mov rbx, 0
    mov r12, seq_numbers
    mov r13, 5
    mov r14, 0
    mov bl, byte [r12+r13*2]
    ; après copie sur 1 octet
    mov r14d, dword [r12+2]
    ; après copie sur 4 octets
    mov rax, 60
    mov rdi, 0
    syscall
```

▸ Exercice 2. *Allocation dans la pile*

1. *Écrivez un programme qui alloue de la mémoire statique pour une liste d'entiers de votre choix, tous strictement positifs sauf le dernier qui sera 0. Votre programme doit*

ensuite allouer de la mémoire dans la pile pour 4 variables (8 octets chacune), puis y mettre le plus grand entier de la liste, le plus petit entier non nul, le nombre d'entiers non nuls, et leur somme, et enfin afficher les résultats.

Vérifiez aussi que vous pouvez appeler la fonction `show_stack` *(définie dans* `utils.asm`*) au début du programme et avant de dépiler les résultats, et que cela n'écrase pas les résultats. (Dans l'affichage de* `show_stack`*, la "base du bloc" n'a pas d'importance pour l'instant, elle servira quand on écrira des fonctions.)*

2. *Écrivez un programme qui alloue de la mémoire dans la pile pour 30 entiers, qui calcule les 30 premières valeurs de la suite de Fibonacci et qui les place dans les emplacements alloués. Trois octets suffisent largement pour contenir la valeur de fib_{30}, mais réservez quand même 8 octets pour chaque variable.*

▸ **Exercice 3.** ***Multiplication et division***

1. *Écrivez un programme qui met dans* ***r12*** *la factorielle de l'entier contenu dans* ***rbx****. Indication :* 10! *tient à l'aise dans 64 bits.*
2. *Modifiez votre code de l'exercice 2.1 pour qu'il compte les entiers pairs.*
3. *Écrivez un programme qui met 1 dans* ***r12*** *si l'entier dans* ***rbx*** *est premier, et 0 sinon.*
4. *Écrivez un programme qui calcule par l'algorithme d'Euclide le PGCD de deux entiers non nuls contenus dans* ***rbx*** *et* ***r12****.*