Université des sciences et de la technologie d'Oran – MB
Faculté des Mathématiques et de l'Informatique - Département d'informatique
2ème Année LMD - S3 - 2021/2022

Corrigé du TP N°2 – Architecture des ordinateurs

Exercice 1 :

```
#Solution TP2/Exercice 1
.data
Newln : .asciiz "\n"
.text
main :
li $t0,1
li $t1,15
loop : move $a0,$t0
li $v0,1
syscall
li $v0,4
la $a0,Newln
syscall
addi $t0,$t0,1
ble $t0,$t1,loop #if ($t0<=$t1 aller à loop)
li $v0,10
syscall
```

Exercice 2 :

```
#Solution TP2/Exercice 2
.data
Nombre: .byte 0
Message1: .asciiz "Donnez un nombre entier : "
reponse1: .asciiz "Ce nombre est positif. \n"
reponse2: .asciiz "Enfin un nombre négatif"
.text
main:
boucle:
li $v0,4
la $a0, Message1
syscall
li $v0,5
syscall
move $t0, $v0
bgez  $t0,SuperieurEgal
li $v0,4
la $a0, reponse2
syscall
j end
SuperieurEgal:
li $v0,4
la $a0, reponse1
syscall
j boucle
end:
li $v0,10
syscall
```

Exercice 3 :

```
#Solution TP2/Exercice 3
.data
Message1: .asciiz "Donnez un nombre entier : "
Message2: .asciiz "La table de multiplication du nombre donné est: \n"
Entree: .asciiz "\n"
produit: .asciiz " x "
egal: .asciiz " = "
.text
main:
li $v0,4
la $a0,Message1
syscall
li $v0,5
syscall
move $t1,$v0
li $t2,1
li $t3,10
li $t4,0
li $v0,4
la $a0,Message2
syscall
boucle:
bgt $t2,$t3,end
move $a0,$t1
li $v0,1
syscall
li $v0,4
la $a0,produit
syscall
move $a0,$t2
li $v0,1
syscall
li $v0,4
la $a0,egal
syscall
add $t4,$t4,$t1
move $a0,$t4
li $v0,1
syscall
li $v0,4
la $a0,Entree
syscall
addi $t2,$t2,1
j boucle
end:
li $v0,10
syscall
```

Exercice 4 :

```
#Solution TP2/Exercice 4
.data
Message1: .asciiz "Donnez un nombre entier : "
Message2: .asciiz "La factorielle du nombre donné est: "
.text
main:
li $v0,4
la $a0,Message1
syscall
li $v0,5
syscall
li $t1,1
boucle:
blt $v0,1, sortie
mul $t1,$v0,$t1
sub $v0,$v0,1
j boucle
sortie:
li $v0,4
la $a0,Message2
syscall
li $v0,1
move $a0,$t1
syscall
li $v0,10
syscall
```

Exercice 5 :

```
#Solution TP2/Exercice 5
.data # déclaration de données
entree1: .asciiz "Entrez un entier X: " ;
entree2: .asciiz "Entrez un entier positif N: " # déclaration de 3
sortie: .asciiz "Résultat = " # chaînes de caractères
.text # début du programme
main:
li $v0, 4 # v0 ← 4 , choisir le service
la $a0, entree1 # a0 ← @entree1 , le paramètre du service
syscall # appel de service d'affichage d'une chaînes de caractères
li $v0, 5 # v0 ← 5 , choisir le service
syscall # appel de service de lecture d'un nombre entier
addu $t0, $0, $v0 # $t0 ← $v0 équivalent de move $t0, $v0
li $v0, 4 # v0 ← 4 , choisir le service
la $a0, entree2 # a0 ← @entree2 , le paramètre du service
syscall # appel de service d'affichage d'une chaînes de caractères
li $v0, 5 # v0 ← 5 , choisir le service
syscall # appel de service de lecture d'un nombre entier
addu $t1, $0, $v0 # $t1 ← $v0 équivalent de move $t1, $v0
li $t2, 1 # $t2 ← 1
boucle:
beq $t1, $0, fin # si ($t1 == 0) on sort de la boucle, c'est la condition d'une boucle tanque
multu $t0, $t2 # $lo ← $t2 x $t0
mflo $t2 # $t2 ← $lo
sub $t1,$t1, 1 # $t1 ← $t1 - 1
j boucle # sauter au début de la boucle
fin:
li $v0, 4 # v0 ← 4 , choisir le service
la $a0, sortie # a0 ← @sortie , le paramètre du service
syscall # appel de service d'affichage d'une chaînes de caractères
addu $a0, $0, $t2 # $t1 ← $v0 équivalent de move $a0, $t2
li $v0, 1 # v0 ← 1 , choisir le service
syscall # appel du service d'affichage d'un nombre entier
li $v0, 10 # v0 ← 10 , choisir le service
syscall # appel de service d'arrêt du programme
```

Exercice 6 :

```
#Solution TP2/Exercice 6
.data
msg1: .asciiz "Donnez S.V.P. un nombre positif : "
msg2: .asciiz "La suite de Fibonacci est comme suit : "
separateur: .asciiz " - "
.text
main:
li $v0 , 4
la $a0 , msg1
syscall # affichage du msg1
li $v0 , 5
syscall
move $s0 , $v0 # lecture de n dans s0, donc n est s0
li $v0 , 4
la $a0 , msg2
syscall # affichage du msg2
li $t0 , 1 # t0 est T0, il est initialisé à 1
li $t1 , 1 # t1 est T1, il est initialisé à 1
li $v0 , 1
move $a0 , $t0
syscall # affichage de T0
li $v0 , 4
la $a0 , separateur
syscall # affichage du separateur
li $v0 , 1
move $a0 , $t1
syscall # affichage de T1
 li $v0 , 4
 la $a0 , separateur
 syscall # affichage du separateur
 loop: add $t2 , $t1 , $t0 # T2 ← T1 + T0 ;
 li $v0 , 1
 move $a0 , $t2
 syscall # affichage de T2
 li $v0 , 4
 la $a0 , separateur
 syscall # affichage du separateur
 move $t0 , $t1 # T0 ← T1 ;
 move $t1 , $t2 # T1 ← T2 ;
 sub $s0 , $s0 , 1 # n-- ;
 bgt $s0 , $0 , loop # condition do-while, si c'est true ça boucle au début du bloque
 li $v0 , 10
 syscall # arrêter le programmee
```