

Test 2

Module : Architecture des ordinateurs L2

Exercice :

Ecrire un programme en assembleur qui permet de lire 10 valeurs entières entrées au clavier, de vérifier si la constante 23 existe dans cette suite ou non ; il affiche le message suivant : "la valeur 23 existe dans cette suite" si la valeur 23 existe dans les valeurs entrées ou bien le message suivant : "la valeur 23 n'existe pas dans cette suite" dans le cas contraire. L'exécution du programme ressemble à ça :

```
insérer 1'élément SVP:20
insérer 1'élément SVP:5
insérer 1'élément SVP:13
insérer 1'élément SVP:3
insérer 1'élément SVP:9
insérer 1'élément SVP:23
insérer 1'élément SVP:18
insérer 1'élément SVP:22
insérer 1'élément SVP:31
insérer 1'élément SVP:2
la valeur 23 existe pas dans cette suite
```

```
insérer 1'élément SVP:20
insérer 1'élément SVP:5
insérer 1'élément SVP:13
insérer 1'élément SVP:3
insérer 1'élément SVP:9
insérer 1'élément SVP:18
insérer 1'élément SVP:22
insérer 1'élément SVP:31
insérer 1'élément SVP:2
insérer 1'élément SVP:64
la valeur 23 n'existe pas dans cette s
```

Indication :

Vous pouvez utiliser 4 registres :

\$t0 : pour la variable compteur.

\$t2 : pour les valeurs entrées au clavier.

\$t3 : pour stocker la constante 23.

\$t4 : valeur booléenne qui prend la valeur 0 si 23 n'existe pas dans le suite et 1 si elle existe.

seq \$t1, \$t2, \$t3 : si \$t2 est égal à \$t3 alors mettre 1 dans \$t1 sinon 0.

beqz \$t1, label : se brancher à label si \$t1 est égal à 0.

Correction Test 2

+ 2 pts

Solution de l'exercice :

.data

entree : .asciiz "insérer l'élément SVP:"

sortie1 : .asciiz "la valeur 23 n'existe pas dans cette suite"

sortie2 : .asciiz "la valeur 23 existe pas dans cette suite"

.text

addi \$t0, \$zero, 0

addi \$t3, \$zero, 23

boucle:

beq \$t0, 10, fin

li \$v0, 4

la \$a0, entree

syscall

li \$v0, 5

syscall

move \$t2, \$v0

beq \$t2, 23, poserbool

addi \$t0, \$t0, 1

b boucle

poserbool:

seq \$t4, \$t2, \$t3

addi \$t0, \$t0, 1

b boucle

fin:

beqz \$t4, afficher1

li \$v0, 4

la \$a0, sortie2

syscall

li \$v0, 10

syscall

afficher1:

li \$v0, 4

la \$a0, sortie1

syscall