Assembleur : Exercices

Exercice 1

Expliquez brièvement (sur le même modèle que les exemples ci-dessus), les instructions suivantes :

```
ADD R0, R1, #42
```

```
LDR R5,98
```

```
1. CMP R4, #18
2. BGT 77
```

```
STR R0,15
```

```
B 100
```

Exercice 2

2.1 Additionne la valeur stockée dans le registre R0 et la valeur stockée dans le registre R1, le résultat est stocké dans le registre R5.

2.2 Place la valeur stockée à l'adresse mémoire 878 dans le registre R0.

2.3 Place le contenu du registre R0 en mémoire vive à l'adresse 124.

2.4 La prochaine instruction à exécuter se situe en mémoire vive à l'adresse 478.

2.5 Si la valeur stockée dans le registre R0 n'est pas égale à 42 alors la prochaine instruction à exécuter se situe à l'adresse mémoire 85.

Exercice 3

Voici un programme Python simple :

```
x = 4
y = 8
if x == 10:
  y = 9
else :
  x=x+1
z=6
```

Voici maintenant son équivalent en assembleur.

```
 MOV R0, #4
STR R0,30
MOV R0, #8
STR R0,75
LDR R0,30
CMP R0, #10
BNE else
MOV R0, #9
STR R0,75
B endif
else:
 LDR R0,30
ADD R0, R0, #1
STR R0,30
endif:
MOV R0, #6
STR R0,23
HALT
```

3.1 Analyser le programme.

3.2 Quelles sont les numéros des cases mémoires qui stockent les valeurs de x,y et z ?

3.3 Simulation

A partir du simulateur suivant : http://www.peterhigginson.co.uk/AQA/

→ Étape 1 : Copier dans un fichier texte le programme

→ Étape 2 : A l'aide de la commande LOAD charger votre programme en mémoire

→ Étape 3 : A l'aide de la commande STEP executer le programme en mode pas à pas. Observer les registres et le placement dans la mémoire