TD2. Où l'on rame un peu

Exercice 1.

Rappel : Une RAM sur l'alphabet Σ est un ensemble infini de registres $R_1, R2, \ldots$ pouvant chacun contenir un mot dans Σ^*, un ensemble de noms de lignes $N_1, N2, \ldots$ et une suite finie d'instructions du type :
1_j) X add j Y (ajoute a_j à la droite du registre)
2) X del Y (efface la lettre la plus à gauche du registre si elle existe)
3) X clr Y (met le mot vide dans le registre Y)
4) X Y←Z (copie le mot de Z dans Y en le laissant aussi dans Z)
5) X jmp X' (va à la ligne X' la plus proche, a avant ou b après)
6_j) X Y jmp j X'(si la première lettre du mot dans Y est a_j, va à la ligne X')
7) X stop (Arrète le programme)
où X est un nom de ligne ou rien, Y et Z des noms de registres et X' un nom de ligne. En fait, les règles 1_j, 2, 6_j et 7 forment un ensemble minimal suffisant pour les fonctions calculables par RAM.
Nous allons montrer que toute fonction calculable par une RAM est calculable par une machine de Turing.
Une SRM (machine à un seul registre) sur Σ est une machine qui n'a qu'un seul registre qui peut contenir n'importe quel mot de Σ^*. Ses instructions sont de trois sortes :
1_j) X add j (ajoute a_j à la droite du registre)
2) X del (efface la lettre la plus à gauche du registre si elle existe)
3_j) X jmp j X' (va à X' si le registre commence par a_j)
4) X stop (Arrète le programme)
où X est un nom de ligne ou rien et X' un nom de ligne.
Si Φ est une fonction partielle de Σ^* alors un programme P qui est une SRM sur $\Sigma' = \Sigma \cup \{,\}$ calcule Φ si quand le registre contient $x_1, \ldots, x_n$ alors P s'arrête et le registre contient $\Phi(x_1, \ldots, x_n)$.

1. Donner une SRM qui sur l'entrée $x_1, \ldots, x_n$ calcule $x_2, \ldots, x_n, x_1$.

2. Montrer que toute fonction calculable par une RAM est calculable par une SRM.

3. Montrer que toute fonction calculable par une SRM est calculable par une machine de Turing.

Exercice 2.
Une RAM sur l'ensemble des entiers est un ensemble infini de registres $R_1, R2, \ldots$ pouvant chacun contenir un entier, un ensemble de noms de lignes $N_1, N2, \ldots$ et une suite finie d'instructions du type :
1) X $R_i := 1$
2) X $R_i := R_j + R_k$
3) X $R_i := R_j - R_k$
4) X $R_i := \lfloor R_i/2 \rfloor$
5) X $R_i := R_{R_j}$
6) X $R_{R_i} := R_j$
7) X aller à la ligne m si $R_i > 0$.
8) arrêt
où X est un nom de ligne.

1. Donner une RAM qui, si on lui donne en entrée les entiers x et i, calcule le i^{e}bit de l'écriture binaire de x.

2. Donner une RAM qui, si on lui donne en entrée les entiers x et y, calcule le produit xy en un temps proportionnel à la longueur des entrées.

3. Donner une RAM qui sur l'entrée n calcule $n!$.

4. Donner une RAM qui, si on lui donne en entrée les entiers x et y, calcule le quotient de la division euclidienne de x par y en un temps proportionnel à la longueur des entrées.

5. Donner une RAM qui accepte les entrées de la forme $1^n 2^{n^2} 0$.