Programmation Impérative

Licence 1^{ere} année

Lundi 04 Mai 2015

On souhaite écrire un programme qui modélise un réseau de type Token Ring. Sur un réseau de type token ring, les machines sont toutes disposées sur un même cercle (ring). Un jeton (token) circule de machine en machine sur le cercle. Une machine n'a le droit de recevoir ou d'émettre un message que lorsqu'elle possède le jeton.

Exercice 1. Machine (5 points)

(1 point) *Définir la structure* `machine_t` *qui contient :*
- *une adresse* **mac** *: il s'agit d'un tableau de 6 nombres tous compris entre 0 et 255,*
- *un* **nom**,
- *un pointeur vers la machine* **suivante**.

(1.5 point) *Écrire la fonction* `creer_machine` *qui prend en entrée une adresse mac et une chaîne de caractère correspondant au nom de la machine à créer. La fonction renvoie un* **pointeur** *vers une structure* `machine_t` *dont les champs auront été initialisés grâce aux paramètres de la fonction. Ne pas oublier d'initialiser le pointeur vers la machine suivante.*

(0.5 point) *Écrire la fonction* `detruire_machine` *qui prend en entrée* **un pointeur** *vers une structure* `machine_t` *et libère l'espace mémoire qu'elle occupe.*

(1 point) *Écrire la fonction* `afficher_machine` *qui prend en entrée* **une structure** `machine_t` *et affiche son nom et son adresse (on l'affichera sous la forme* 10.10.10.10.10.1*). l'espace mémoire qu'elle occupe. (Bonus, faire un affichage de l'adresse en Hexadecimal :* $0A.0A.0A.0A.0A.01$*)*

(1 point) *Écrire la fonction* `est_destinataire` *qui prend en entrée un pointeur m vers une* `machine_t` *et un tableau d'entier correspondant à une adresse mac. La fonction renvoie* 1 *si l'adresse mac passée en entrée est la même que celle de la machine. La fonction renvoie* 0 *sinon. On*

accordera une attention particulière à la boucle utilisée pour écrire cette fonction.

Exercice 2. Listes chaînées circulaire : le token ring (6.5 points)
Dans cet exercice, on va créer des listes chaînées dont les maillons sont les `machine_t`*. On utilisera donc les fonctions définies dans l'exercice précédent. Toutefois, il n'est pas nécessaire d'avoir réussi ces fonctions pour faire cet exercice.*

(0.5 point) *Définir la structure* `ring_t` *qui contient un pointeur* `debut` *et un pointeur* `fin` *sur une* `machine_t`*.*

(1 point) *Écrire la fonction* `ring_creer` *qui ne prend rien en entrée et renvoit un pointeur sur une* `ring_t` *ne contenant aucune machine. On pensera à initialiser les champs.*

(1 point) *Écrire la fonction* `ajouter_machine_fin` *qui prend en entrée un pointeur sur un* `ring_t` *et un pointeur sur une* `machine_t` *et ajoute la machine à la fin du ring. Attention au cas où le ring est vide. De plus, n'oubliez pas qu'il s'agit d'un cercle : la fin pointe vers le début.*

(2 point) *Écrire la fonction* `extraire_machine_debut` *qui prend en entrée un pointeur sur un* `ring_t`*. La fonction renvoie* $NULL$ *si le* `ring_t` *est vide. Sinon, la fonction extraie l'élément au début du ring et renvoie son adresse. Tout comme pour l'ajout, on fera attention aux différents cas. Il est fortement conseillé de faire des dessins aux brouillons pour vérifier ces deux fonctions.*

(1 point) *Écrire la fonction* `ring_detruire` *qui prend en entrée un pointeur sur une* `ring_t` *et libère complètement l'espace qu'elle occupe. Cette fonction devra impérativement utiliser deux fonctions définies précédement.*

(1 point) *Donnez la complexité en temps et en espace de la fonction* `ring_detruire`*.*

Exercice 3. Les messages (5.5 points)
Dans cet exercice, on utilise une structure `message_t` *définie par un autre développeur. On ne sait pas ce que contient cette structure mais on sait que l'on peut la manipuler avec les fonctions suivantes :*

- *la fonction* `message_t * emettre_message(machine_t * m);` *prends en entrée un pointeur sur une machine. Si celle ci désire émettre un message, la fonction alloue dynamiquement de la mémoire et renvoie un pointeur sur le message crée. Sinon, la fonction renvoie* $NULL$*.*
- *la fonction* `detruire_message(message_t * m);` *libère l'espace mémoire occupé par un message.*
- *la fonction* `char * lire_contenu(message_t * m);` *renvoie le contenu du message sous forme d'un chaîne de caractère.*

— *les fonctions* `int * adresse_emetteur(message_t * m);` *et* `int * adresse_destinataire(message_t * m);` *renvoie respectivement les adresses mac du l'émetteur et du destinataire.*

Dans les questions qui suivent, on utilisera les fonctions permettant de manipuler un message en considérant qu'elles ont déjà été définies : vous ne devez pas écrire ces fonctions.

(2.5 points) *Écrire la foncion* `sauvegarder_message` *qui prend un pointeur sur un message en entrée, ouvre le fichier* `message.log` *situé dans le répertoire* `/tmp/` *et écrit* **à la fin** *du fichier, sur une seule ligne, l'adresse mac de l'émetteur puis du destinataire.*

(3 points) *Écrire la foncion* `circulation_message` *qui prend en entrée un pointeur sur un* `ring_t`*. La fonction parcourt les machines sur le cercle et tente d'amener un message de l'emetteur vers son destinataire. Un seul message à la fois peut-être acheminé sur le réseau. Si aucun message n'a été transmis, la fonction teste si une machine veut émettre un message et passe à la machine suivante. Lorsqu'une machine émet un message, la fonction affiche ses informations et précise qu'elle vient d'émettre un message. Si un message est en cours de transmission, la fonction teste pour chaque machine s'il s'agit du destinataire. Si c'est le cas, la fonction affiche le contenu du message, appelle la fonction* `sauvegarde_message` *et détruit le message. La fonction s'arrete lorsqu'un tour complet du cercle a été effectué et qu'il n'y a pas de message à transmettre.*

Exercice 4. Le programme (3 points)

(2 points) *Écrire la foncion* `creer_ring_simple` *qui ne prend rien en entrée et renvoie un pointeur sur un ring, contenant 10 machines dont l'adresse mac est de la forme :* $10.10.10.10.10.x$ *(où* x *varie entre* 0 *et* 9*) et le nom de la machine est "Machine x" (idem pour le* x*). Afin de modifier le nom de la machine, on utilisera la fonction* `sprintf` *(permettant de faire un* `printf` *dans une chaîne de caractères). Le principe est le même que pour* `fprintf` *: le premier argument de la fonction est la chaîne dans laquelle on souhaite écrire, le reste fonctionne comme* `printf`*. On pensera bien sûr au préalable à créer un tableau de caractères de taille suffisante pour stocker la chaîne de caractères.*

(1 points) *Écrire la fonction* `main` *qui crée un ring simple, appelle la fonction* `circulation_message` *et détruit le ring.*