



Examen n°2 de Programmation en C - IR1 2008-2009

Polycopiés de cours et notes de TD autorisés Durée: 2 heures

Il était une fois une gentille princesse qui devait apprendre la programmation réseau, car rien en ce temps-là n'était plus doux à l'oreille des princes qui étaient tous d'indécrottables geeks. Comme elle était un peu fainéante, elle voulut le faire en Java. Malheureusement, sa marâtre qui était une affreuse sorcière la surprit et voulut la punir: "puisque c'est ainsi, tu coderas tout en assembleur dans vi, gnark, gnark!". Heureusement, sa marraine la bonne fée réussit à conjurer partiellement le sort en la laissant coder en C. Cependant, la princesse devait toujours être capable de surmonter 4 terribles épreuves.

Épreuve 1: IP et opérations binaires (3 points)

Sachant que les règles sur les classes d'adresses IP sont les suivantes:

premier octet commençant par: 0XXX => classe A
 10XX => classe B
 110X => classe C
 1110 => classe D
 1111 => classe E

écrire une fonction prenant un tableau d'**unsigned char** représentant une adresse IP et retournant une valeur décrivant sa classe.

Épreuve 2: IP et E/S (6 points)

1) Écrire une fonction **get_IP** prenant une chaîne de caractères représentant une adresse IP (ex: "**128.34.77.0**") capable de la convertir en un tableau de 4 **unsigned char**. La princesse est libre d'utiliser le prototype de son choix, mais elle devra le justifier très soigneusement. De plus, son miroir magique lui suggère de se rappeler de **sscanf**.

2) En réutilisant la fonction de l'épreuve 1, écrire un programme prenant une chaîne de caractères représentant une adresse IP en argument et affichant sa classe sur la sortie standard, ou un message d'erreur pertinent sur la sortie d'erreur en cas d'adresse malformée.

Épreuve 3: ARP et allocation dynamique (6 points)

- 1) Sachant qu'une table ARP est composée de paires (adresse IP, adresse MAC), la princesse doit proposer une structure **ARP_entry** permettant de stocker une telle paire.
- 2) Sachant qu'une table ARP a une taille qui peut varier au cours du temps, la princesse doit proposer une structure **ARP_table** capable de stocker un tableau de **ARP_entry**, qui puisse s'agrandir dynamiquement si nécessaire.
- 3) Écrire la fonction **get_MAC**, prenant en paramètres un tableau d'**unsigned char** représentant une IP et une table ARP, et retournant l'adresse MAC correspondant si elle est présente dans la table. Une fois encore, la princesse devra faire bien attention à justifier son prototype.
- 4) Écrire la fonction **add_ARP_entry**, capable d'ajouter une entrée dans une table donnée, en la redimensionnant si nécessaire.

Épreuve 4: fichiers (6 points)

Pour son ultime épreuve, la princesse doit décrire un format de fichier de sauvegarde pour ses tables ARP. Le format est libre (texte ou binaire), mais la princesse doit fournir les deux fonctions de lecture et écriture, dont les prototypes sont les suivants:

```
ARP_table* load_ARP_table(char* filename)
void save_ARP_table(char* filename, ARP_table* table)
```

Note: bien que tout ceci soit à la charge de la princesse, vous serez noté proportionnellement à l'aide que vous lui apporterez.