## 1) Réseaux: Adresses IP(v4) avec masque identique ( 2,5 pts)

Soient 4 stations st1,st2, st3, st4 ayant respectivement les adresses suivantes:
@st1:172.30.135.45 @st2: 172.30.135.152 @st3: 172.30.135.65 @st4: 172.30.135.114
On configure les stations avec le masque 255.255.255.224

**1.1)** Déterminer les adresses de sous-réseau et de broadcast pour chaque station
**1.2)** Calculer le nombre d'adresses disponibles pour des hôtes dans un sous-réseau.

## 2) Réseaux VLSM ( 3 pts)**:** Découpage de réseaux à des tailles appropriées

On optimise l'adressage d'un réseau en utilisant un masque de sous-réseau de longueur variable: VLSM . Cela permet d'attribuer des nombres d'adresses différents selon les besoins de chaque sous-réseau. Le réseau d'adresse 192.68.0.0/24. est décomposé en 4 sous-réseaux.

☺ Déterminer et représenter dans un tableau, pour chacun des 4 sous-réseaux, son adresse, son masque, son adresse de diffusion, la plage d'adresses disponibles pour un hôte.
Nombre d'adresses d'hôtes nécessaires par sous-réseau:

| | |
|---|---|
| sous-réseau1: | 110 |
| sous-réseau2: | 48 |
| sous-réseau3: | 11 |
| sous-réseau4: | 2 |

## 3) Routage (3 pts)

Un routeur peut livrer directement des paquets aux interfaces 0 et 1 ou les retransmettre vers les routeurs R2, R3,R4 ou R5. Soit sa table de routage :

| **No de sous-réseau** | **Masque de sous-réseau** | **Saut suivant** |
|---|---|---|
| 168.96.39.0 | 255.255.255.192 | Interface 0 |
| 168.96.39.128 | 255.255.255.192 | Interface 1 |
| 168.96.40.0 | 255.255.255.192 | R2 |
| 192.4.153.96 | 255.255.255.224 | R3 |
| 192.4.153.64 | 255.255.255.240 | R5 |
| <par défaut> | <par défaut> | R4 |

☺ Donner la saut suivant pour les paquets à destination des adresses suivantes :
a) 168.96.39.187 b) 168.96.39.105 c) 168.96.40.68
d) 192.4.153.78 e)192.4.153.87 f) 168.96.40.58

## 4) Réseau:Dysfonctionnement (1.5 pts)

Sur le schéma fourni en **annexe 1** les stations st1 et st2 ne peuvent pas communiquer.
Expliquer le dysfonctionnement.et déterminer le masque de sous-réseau afin que les 2 stations puissent communiquer.

## 5) Requêtes SQL ( Base en **annexe 2**) (3 pts)

**5.1)** Afficher les noms des différents éditeurs de livres de l'auteur " KING " ,
**5.2)** Afficher les titres et le prix des livres de " KING " parus chez l'éditeur " J'ai Lu ".
**5.3)** Déterminer le titre du livre le moins cher de " KING " chez " Fusion Comics " .

## 6) Linux Annexe 3(3 pts)

**6.1)** Donner la commande donnant l'affichage des attributs des fichiers du dossier en cours.
exemple:

```
-rwxr--r-- 1 sergio sergio 383 2011-10-02 17:45 copie
-rw-r--r-- 1 sergio sergio 143 2011-10-02 17:44 efface
```

**6.2)** Proposer une commande qui donne le droit d'exécution au fichier *efface* à tout le monde.

**6.3)** Donner le code de ***efface.*** Ce script prend 1 argument: le nom du fichier à effacer.
Un message d'erreur est affiché au cas où le fichier n'existe pas.

**6.4)** Réaliser ***efface2.*** Ce script ne prend pas d'argument. Par contre lors de son exécution il invite à taper le nom du fichier à effacer.

## 7) Développement web (annexe 4) ( 4pts)

**7.1) Javascript:**
A partir du formulaire fourni, modifier la fonction javascript `Valide` afin d'envoyer le formulaire au serveur seulement si le champ est rempli sinon un message d'alerte est affiché.

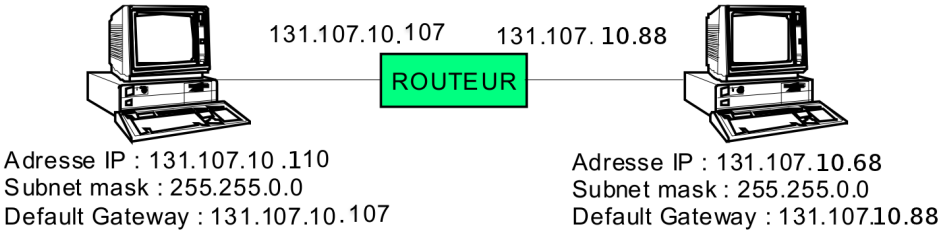
**7.2)PHP :**
☺Donner le code PHP permettant de visualiser le contenu du tableau `$_POST`.

☺Ecrire le code permettant d'affecter la variable `$n` avec le message venant du formulaire.

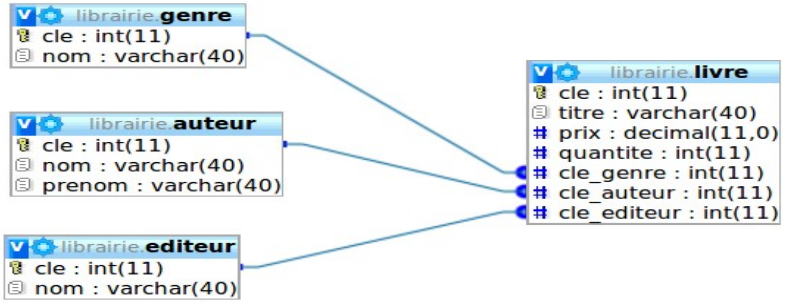
**7.3) PHP :** Ecrire la fonction `compare($n1,$n2)` qui retourne
0 si les 2 variables sont égales, -1 si $n1>$n2, 1 si $n1<$n2

```
function  compare($n1, $n2){}
```

## Annexe 1 :




## Annexe 2 :




## Annexe 3 :

**Efface:**
Exemple d'utilisation: `sergio@laptop:~$ efface  fichier1`

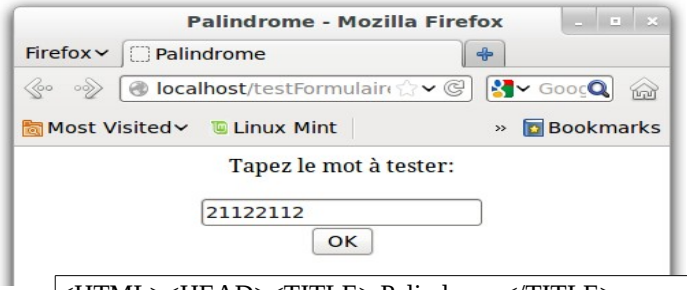
Ce script doit vérifier la présence du paramètre. Si le paramètre existe, il faut vérifier que cela corresponde à un fichier et dans ce cas, effacer ce fichier sinon indiquer un message d'erreur.

**Rappel**: test -e $1 //vérifie si le paramètre correspondant à $1 est un fichier.
a=$? // a reçoit le résultat de l'opération précédente
$a = 0 si $1 correspond à un fichier .

**Rappel2:**

**read param1** permet de lire le paramètre param1 tapé au clavier
$param1 permet d'exploiter la valeur affectée au paramètre.

## Annexe 4 :




Code du formulaire actuel

```
<HTML><HEAD><TITLE> Palindrome</TITLE>
<SCRIPT type=text/javascript>
function Valide( n)
{
alert(n.value);
document.formulaire.submit();
}
</SCRIPT></HEAD><BODY>
<CENTER>
Tapez le mot &agrave; tester:
<BR><FORM name="formulaire" action="traitement.php"
method="POST">
<BR><input type="text" name="n"><input type="hidden"
name="alea" value=21>
<input type="button" value="OK" onclick="Valide(n)">
</FORM></CENTER></BODY></HTML>
```

Dans cet exemple la fonction
Valide affiche la valeur de n et envoie le formulaire au serveur

Affichage attendu à la première partie de la question 8.2
Array ( [n] => 21122112 )