Nom: Prénom:

Prog. Sys. et Réseau 2016—2017

QCM 1 (5pt)

Chaque question sur 1 point, une seule réponse par question, toute rature sera considérée comme une mauvaise réponse.

Comment sont indexés les «fichiers» Unix en interne ?
- ☐ lien (link)
- ☐ lien symbolique (symbolic link)
- ☐ chemin (pathname)
- ☐ i-nœud (inode)

Qu'est ce qui constitue l'espace de nommage des «fichiers» Unix ?
- ☐ les liens (link)
- ☐ les répertoires (directory)
- ☐ les processus (process)
- ☐ les périphériques (devices)

En C, toute manipulation du contenu d'un fichier nécessite :
- ☐ un tampon (buffer)
- ☐ un pointeur sur `FILE`
- ☐ un lien
- ☐ une socket

Du point de vue du système Unix, en C la manipulation du contenu d'un fichier nécessite :
- ☐ un descripteur de type `int`
- ☐ un pointeur de type `int`
- ☐ un `HANDLE`
- ☐ un `InputOutputStream`

En C, il n'existe pas dans la bibliothèque :
- ☐ de fonctions de manipulations de chaînes
- ☐ de types pour représenter les collections
- ☐ de fonctions mathématiques
- ☐ de fonctions d'entrées/sorties

QCM 2 (5pt)

Chaque bonne réponse sur 1 point, plusieurs réponses possibles, toute mauvaise réponse ou rature compte 1 point en moins.

Un tube (pipe) est :
- ☐ un moyen de communication entre processus
- ☐ un i-nœud qui représente en périphérique
- ☐ un i-nœud qui ne conserve pas les informations qu'il contient de façon permanente
- ☐ un moyen de communication entre machines

Parmi les propositions suivantes lesquelles sont des références relatives :
- ☐ `/etc/../usr/bin/ls`
- ☐ `./ps`
- ☐ `.././.././bidule`
- ☐ `root/`

Nom: Prénom:

QCM 2 (suite)

Soit le résultat d'une commande `ls`

```
3756475 -rw-r--r-- 1 root  wheel  5253 31 oct  2013 /etc/passwd
```

quelles propositions sont vraies :

- le fichier est de taille 3756475 octets
- tout utilisateur peut le lire
- ce n'est pas un exécutable
- il possède 1 et 1 seul nom

EXERCICE 1 (5pt)

Écrire en C et à l'aide des fonctions systèmes une commande permettant de copier le contenu d'un fichier dans un autre (les fichiers d'inclusions nécessaires peuvent être omis). Le nom de ces fichiers étant passés en arguments de la ligne de commande, comme par exemple :

macopie *fichier_source fichier_copie*

On oubliera pas de traiter quelques cas d'erreurs (bon nombre d'arguments, existence des fichiers ou non, etc). Aide: `int open(char *pathname,int mode, mode_t access_rights); ssize_t read(int descriptor, void *bufffer, size_t length); ssize_t read(int descriptor, void *bufffer, size_t length); int close(int descriptor);`

EXERCICE 2 (5pt)

Écrire en C et à l'aide des fonctions systèmes une commande permettant de chiffrer le contenu d'un fichier. La commande sera invoquée de la façon suivante :

chiffre *clé fichier*

et produira en sortie un fichier de nom fichier.chiffre qui sera le résultat du chiffrement. Le chiffrement (très élémentaire mais efficace et connu sous le nom de One Time Pad) consiste simplement à appliquer la fonction suivante :

La clé est une chaîne de caractère quelconque, ex.: 4abc3z24, chacun d'eux est noté k_i dans la suite (c'est-à-dire i-ème caractère de la clé—key).

Le chiffrement est obtenu de la façon suivante (le caractère ^ représente le XOR bit-à-bit du C) :

Fichier	c_1	c_2	…	c_n	c_{n+1}	c_{n+2}	…	c_{2n}	…
Clé	k_1	k_2	…	k_n	k_1	k_2	…	k_n	…
Fichier chiffré	c_1^k_1	c_2^k_2	…	c_n^k_n	c_{n+1}^k_1	c_{n+2}^k_2	…	c_{2n}^k_n	…

On fera en sortie que le fichier résultat possède les mêmes droits d'accès que le fichier d'origine. Aide: même fonctions que pour l'exercice précédent et `int stat(char *path_name, struct stat *stat_buf);`. On rappelle aussi que la structure `stat` contient de nombreux champs parmi lesquels :

```
struct stat {
    dev_t     st_dev;     /* device inode resides on */
    ino_t     st_ino;     /* inode's number */
    mode_t    st_mode;    /* inode protection mode */
    nlink_t   st_nlink;   /* number of hard links to the file */
    uid_t     st_uid;     /* user-id of owner */
    gid_t     st_gid;     /* group-id of owner */
```