

Examen SINF1252 - 30 août 2019

Prénom et Nom :  
(majuscules)

NOMA (sans - ni .) :

Instructions [Durée : 1h15]

- Aucun document, aucune autre feuille, même de brouillon ne sont autorisés durant l'examen
- Aucun dispositif électronique n'est autorisé durant l'examen écrit
- **Répondez à l'encre avec une écriture lisible et uniquement dans les cadres prédéfinis.** Si une réponse doit être plus longue, utilisez le recto de la dernière feuille. Utilisez les versos comme feuilles de brouillon.

1 Système de fichiers [1 point]

Le système de fichiers Unix stocke différents types d'informations dans les **inodes** et les **répertoires**. Pour chaque type d'information, marquez avec une croix dans le tableau si elle est contenue dans l'inode ou le répertoire.

	Inode	Répertoire
Nom du fichier		X
Taille du fichier	X	
Liste des blocs du fichier	X	
Date de création du fichier	X	
Propriétaire du fichier	X	
Permissions du fichier	X	

NOMA (sans - ni .)

## 2 Algorithme de Peterson [3 points]

Proposez une implémentation correcte en pseudocode de l'algorithme de Peterson.

```
#define A 0
#define B 1
int flag[2];
int turn

void* thread A (void*) {
    flag[A] = true;
    turn = B;
    while (flag[B] == true && turn == B) {
        // waiting loop
    }
    section-critique();
    flag[A] = false;
}
```

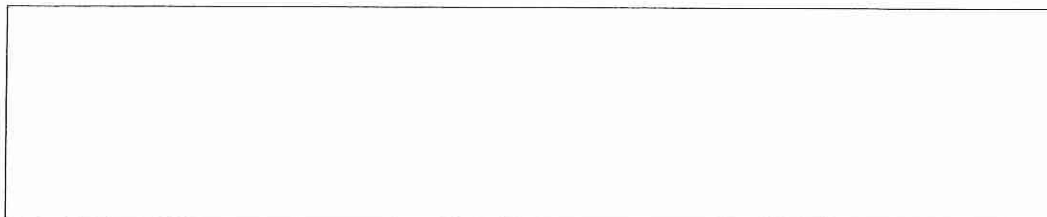
Voir feuille annexe

NOMA (sans - ni .) :

### 3 Mémoire virtuelle [2 points]

Expliquez, **en utilisant un dessin clair**, le fonctionnement de la table des pages et la façon dont celle-ci permet de traduire une adresse virtuelle en adresse réelle.

Si on considère un processeur utilisant des adresses virtuelles encodées sur 32 bits, avec une mémoire utilisant des adresses réelles de 28 bits, combien de lignes sa table des pages contiendra-t-elle? On supposera l'utilisation de pages de 8 KBytes. Justifiez votre réponse.



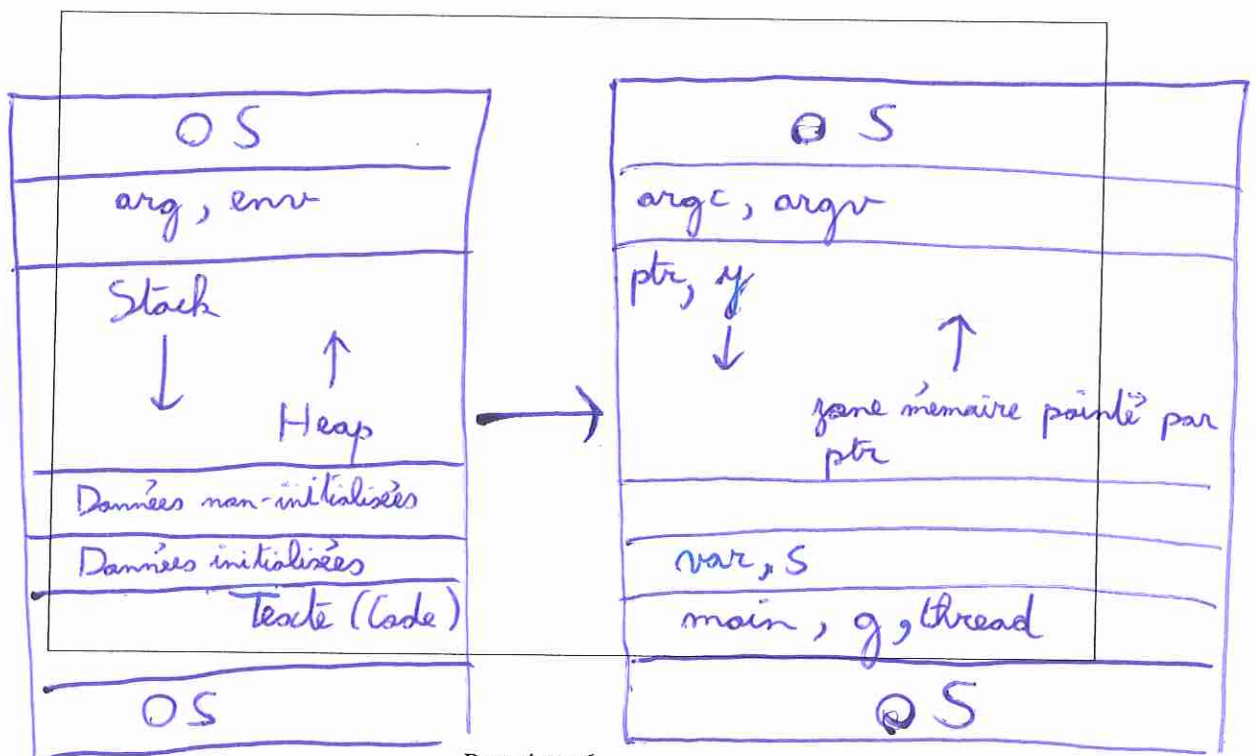
NOMA (sans - ni .) :

#### 4 Organisation d'un processus en mémoire [2 points]

Considérons le programme ci-dessous qui est en train de s'exécuter. Ce programme comprend un thread qui exécute la fonction `thread`.

```
int var=123;
void * thread(void *a) {
    // code non fourni
}
void g(int x) {
    static int s=789;
    // ...
}
int main(int argc, char **argv) {
    int y=456;
    g(y);
    int *ptr=(int *)malloc(12*sizeof(int));
    // ...
}
```

Représentez graphiquement l'organisation de la mémoire de ce programme en indiquant notamment l'endroit où se trouvent les variables `var`, `y`, `ptr`, `s`, et les fonctions `main`, `g` et `thread`.



Le thread a-t-il accès aux variables suivantes ? Justifiez votre réponse.

Variable	Oui/Non	Justification
var	Oui	Car un thread a accès aux mêmes données initialisées que le thread dant il provient
y	Non	Car des threads ne peuvent pas se partager des données provenant de leur stack
ptr	Non	Même chose que pour la variable y en revanche il a accès à la zone mémoire pointée par celui-ci.
zone mémoire pointée par ptr	Oui	Car un thread peut partager son heap avec un thread qu'il vient de créer
s	Oui	Même chose que pour la variable var, car 2 threads peuvent avoir le même segment de données initialisées

NOMA (sans - ni .) :

Brouillon ou complément de réponse

```
2) void* thread B(void*) {  
    flag[B] = true;  
    turn = A;  
    while (turn == A && flag[A] == true) {  
        // waiting loop  
    }  
    section-critique();  
    flag[B] = false; }  
}
```