# **Compte rendu – Devoir 2**

## Exercice 2

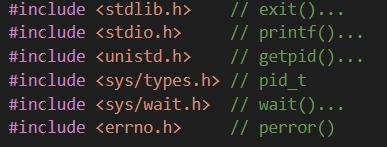
### Objectif

Nous souhaitons faire communiquer un processus fils avec son processus père. Le premier doit écrire un entier dans un fichier, puis son processus père doit récupérer cet entier en lisant le ficher.

Les codes se situent dans les fichiers ex2\_qt4.c et ex2\_qt5.c

Sachant que les headers files, les fonctions ecrire\_fils et lire\_pere sont identiques dans ces deux fichiers.

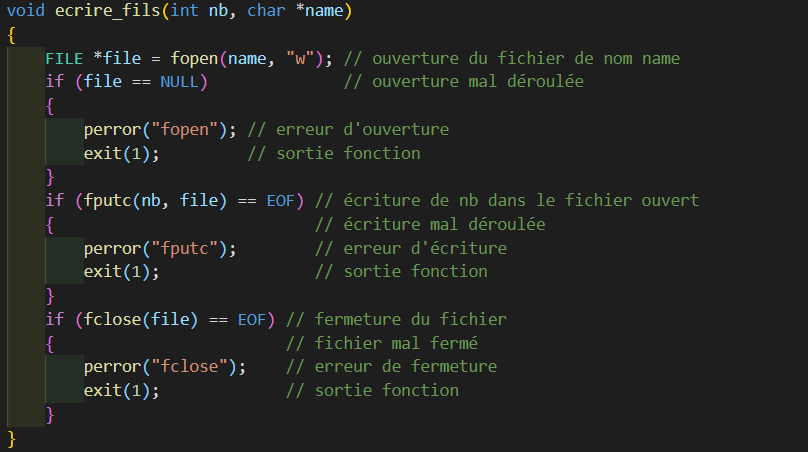
### Initialisation des headers files



On définit deux constantes :

* NOMFICHIER qui sera le nom du fichier dans lequel on va écrire et lire le nombre ;
* NB qui sera le nombre qu’on écrira et lira dans le fichier.

### Fonction ecrire\_fils

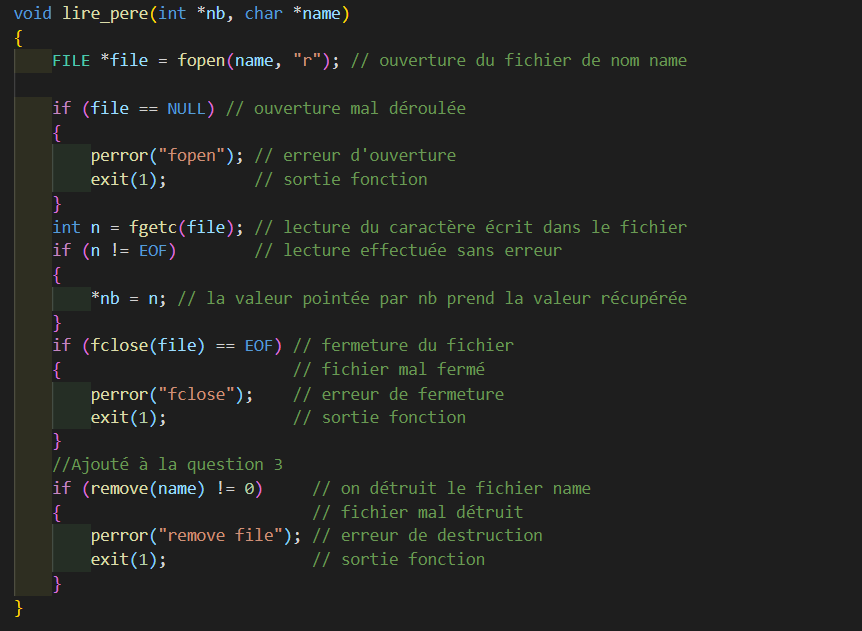


L’objectif de cette fonction est d’ouvrir un fichier de nom name donné en paramètres de la fonction, puis d’y écrire le nombre nb donné aussi en paramètres de la fonction. On veille à ce que chaque étape se déroule bien, si ce n’est pas le cas on affiche des erreurs et on quitte la fonction.

A savoir :

* La fonction fopen retourne un pointeur sur un fichier si l’ouverture s’est bien déroulée sinon elle retourne NULL ;
* La fonction fputc retourne le caractère qui vient d’être écrit si l’écriture s’est bien déroulée sinon elle retourne EOF (end of file) ;
* La fonction fclose retourne 0 si le flux a bien été fermé sinon elle retourne EOF (end of file).

### Fonction lire\_pere



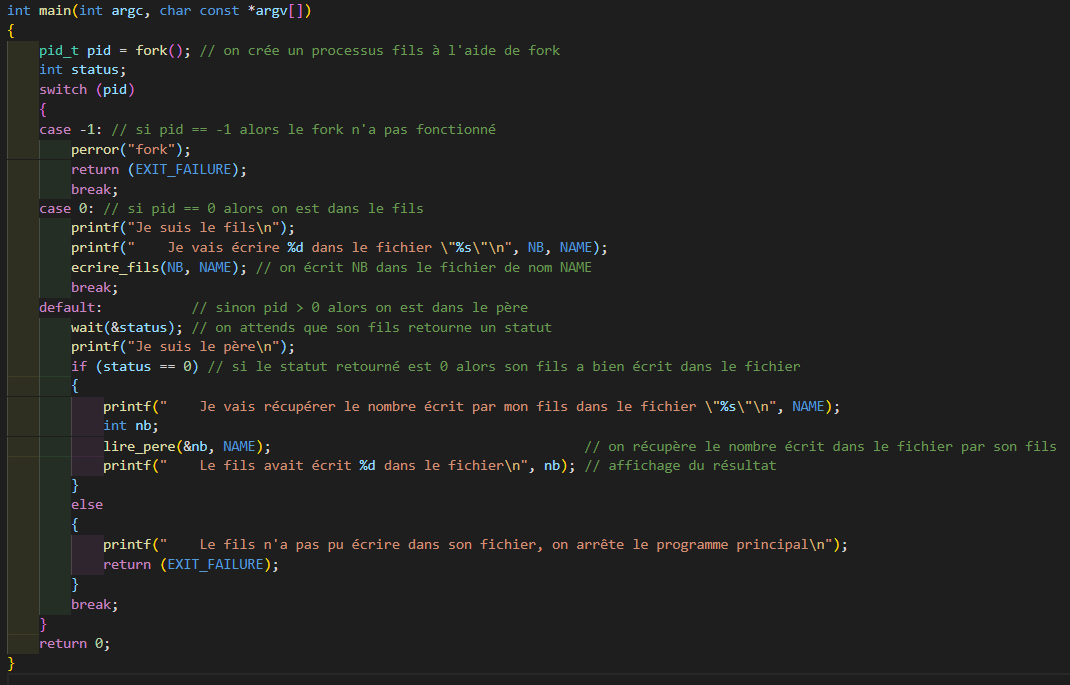
L’objectif de cette fonction est de récupérer dans un fichier name un caractère, si l’ouverture et la lecture du fichier s’est bien déroulée, ce caractère sera mis dans la variable nb passée en paramètres. On ferme ensuite le fichier, puis on détruit le fichier name qui ne nous servira plus (partie ajoutée à la suite de la question 3).

A savoir :

* La fonction fgetc retourne le caractère qu’il vient de récupérer si la récupération s’est bien déroulée sinon elle renvoie EOF (end of file) ;
* La fonction remove retourne 0 si le fichier a bien été détruit sinon elle retourne -1

### Programme principal 1



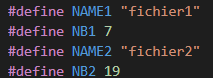


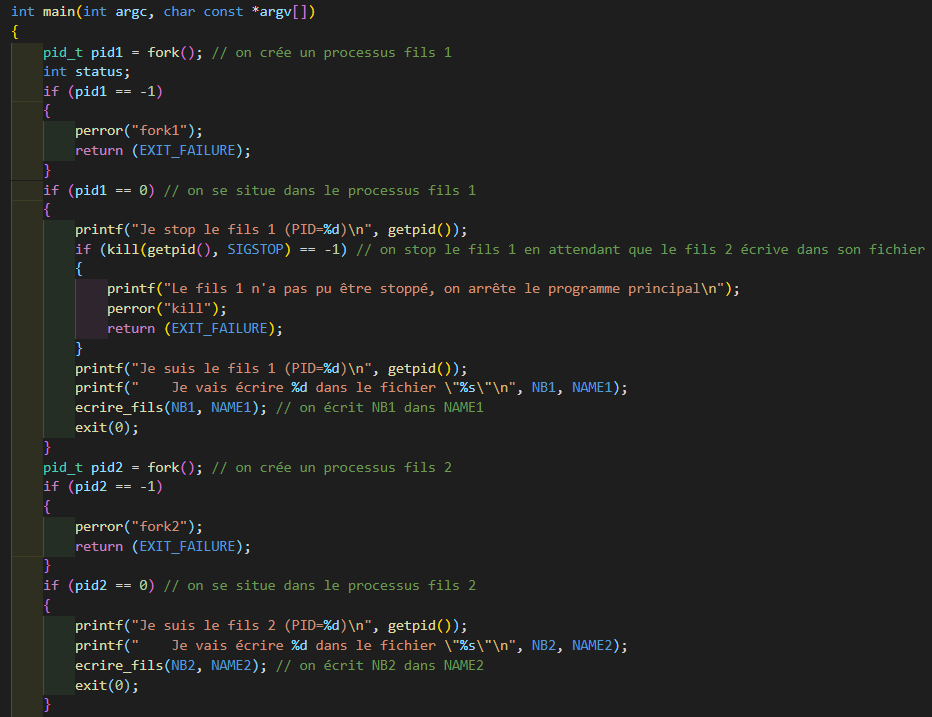
Le premier programme principal demandé à la question 4 est codé dans le fichier ex2\_qt4.c

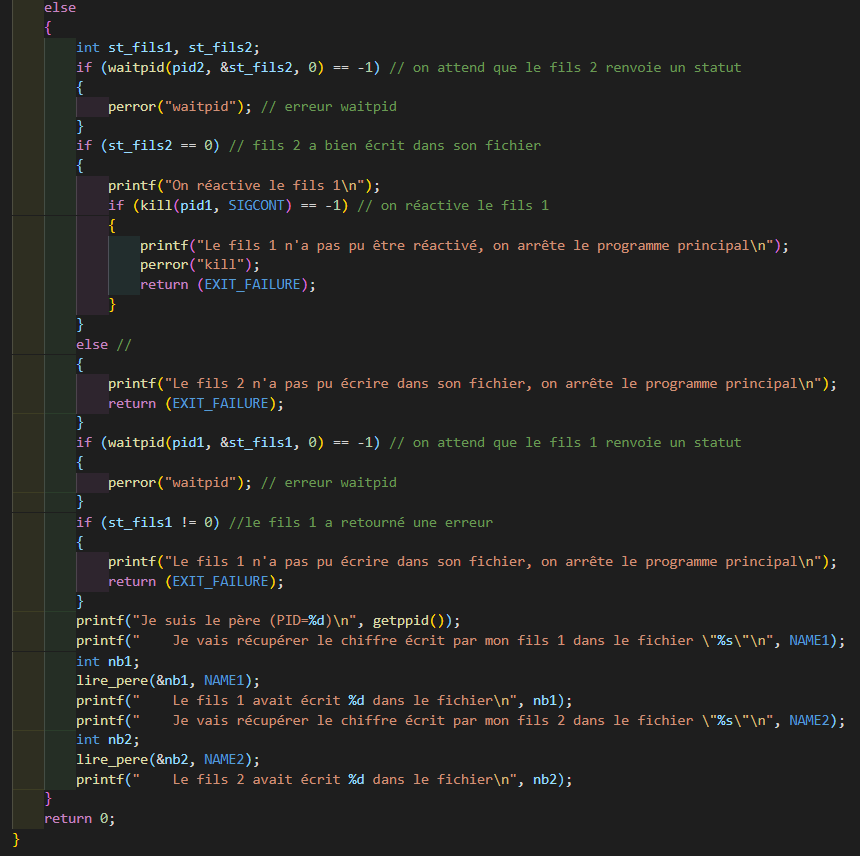
Ce programme nous permet de créer un processus fils qui écrira dans le fichier NAME un nombre NB, définis au début comme constantes, ce nombre sera récupéré par son processus père.

On crée un processus fils à l’aide de la fonction fork, si on se situe dans le processus fils alors on écrira le nombre NB dans le fichier NAME à l’aide de la fonction ecrire\_fils précédente. Si on se situe dans le processus père alors on attends la fin du processus fils pour récupérer la valeur qu’il a écrite dans le même fichier à l’aide de la fonction lire\_pere précédente.

### Programme principal 2







Le second programme principal demandé à la question 5 est codé dans le fichier ex2\_qt5.c

Ce programme nous permet, à l’aide des signaux SIGSTOP et SIGCONT, de laisser le deuxième fils écrire NB2 dans le fichier NAME2, puis le premier fils écrit NB1 dans le fichier NAME1, le père enfin récupérera les deux nombres.

On crée un premier processus pid1 à l’aide de fork, quand on se situe pour la première fois dans le fils 1, on envoie le signal SIGSTOP pour le mettre en pause, quand il sera réactivé il écrira NB1 dans le fichier NAME1 à l’aide de la fonction ecrire\_fils précédente. On crée ensuite un processus fils pid2 à l’aide de fork, il écrit NB2 dans le fichier NAME2 à l’aide de la fonction ecrire\_fils précédente. Quand on rentre dans le processus père, il attend le signal du fils 2 pour réactiver le fils 1. Quand tout s’est bien déroulé, il ira récupérer les valeurs écrites par ses fils dans les fichiers à l’aide la fonction lire\_pere précédente.