Cours d'administration Linux

M. Davy MOUSSAVOU- Consultant Linux, Certifié Linux International

Session 2: Les entrées sorties

Objectifs du cours

Cette session sera consacrée a:

- La manipulation des entrées sorties de base;
- La gestion des fichiers

Agenda

- Séquence 0: Synthèse de la session 1
- Séquence 1: Descripteur de fichier
- Séquence 2: Appels systèmes de gestion des fichiers
- Séquence 3: Opération d'entrée sortie sur les fichiers

Séquence 0 : Fondamentaux du C

Séquence 0 : Fondamentaux du C

❑ Objectifs

- Manipuler de pointeur;
- Manipuler des structures de données;
- Manipuler des fonctions.

Séquence 0 : Fondamentaux du C

Manipulation des pointeurs

Séquence 0 : Fondamentaux du C

Manipulation des pointeurs

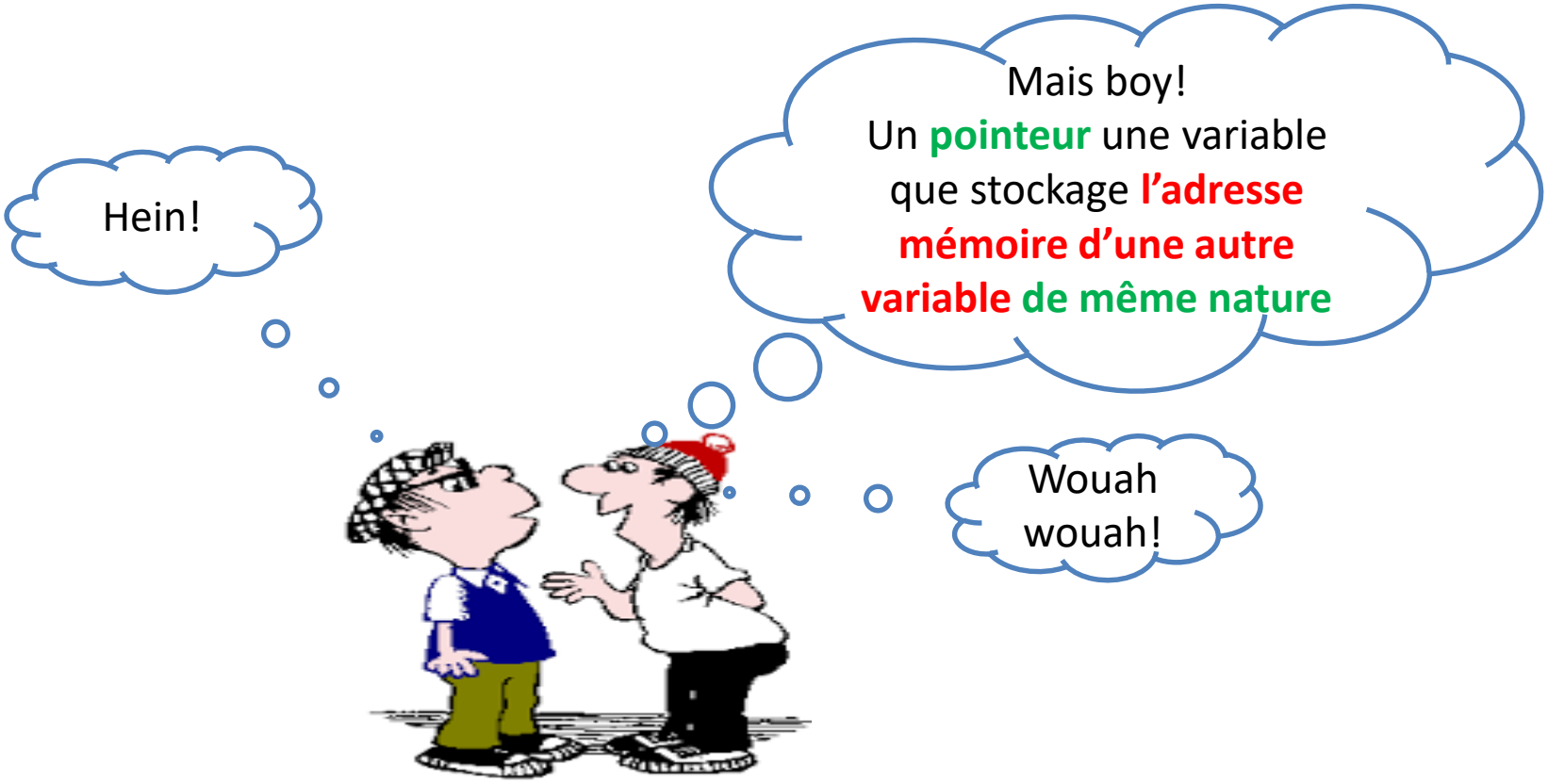
Séquence 0 : Fondamentaux du C

Manipulation des pointeurs

Hier au cours d'Unix, M. MOUSSAVOU nous a parlé des pointeurs, mais je ne pige rien à son **charabia!**

Séquence 0 : Fondamentaux du C

Manipulation des pointeurs



Séquence 2 : Fondamentaux du C

Manipulation des pointeurs

Séquence 0 : Fondamentaux du C

Manipulation des pointeurs

Séquence 0 : Fondamentaux du C

Manipulation des pointeurs

Pour un pointeur **K**:
***K désigne valeur de l'objet pointe**

Par exemple, si **A=7**
Et
K=&A
***K fait référence à 7**

Séquence 0 : Fondamentaux du C

Manipulation des pointeurs

❑ Ce qu'il faut retenir sur les pointeurs

- ❖ **Une pointeur stocke la l'adresse mémoire d'une autre variable de même nature;**

Séquence 0 : Fondamentaux du C

Manipulation des pointeurs

- **Ce qu'il faut faire**

```
int A;

A=7;

int *P;

P=&A  // P reçoit  l'adresse de A.
```

Manipulation des pointeurs

- **Ce qu'il ne pas faut faire**

```
int A;

A=7;

int *P;

P=A  // à ne pas faire, car P et A ne sont pas de même

type.
```

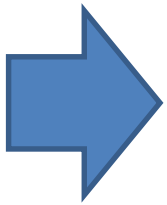
Séquence 0 : Fondamentaux du C

Manipulation des structures

❑ Syntaxe de la déclaration

```
struct nomstructure {
type membre1
type membre 2
.....
type membre n
};
```

Exemple



```
struct Appel {
int  Numppelant;
int  NumDestinatire;
int Duree;
};
```

Séquence 0 : Fondamentaux du C

Manipulation des structures

❑ Utilisation de la structure

```
struct nomstructure variable;

variable. membre1=valeur;

variable. membre1=valeur;

.....

variable. membre n=valeur;
```

```
struct Appel cdr;

cdr. NumAppelant=775736903;

cdr. NumDestinaire=765620323;

cdr. Duree=87;
```

Séquence 0 : Fondamentaux du C

Manipulation des structures

❑ Utilisation de la structure

```
struct nomstructure *variable;

Variable->membre1=valeur;

variable->membre1=valeur;

.....

variable->membre n=valeur;
```

```
struct Appel *cdr;

cdr->NumAppelant=775736903;

cdr->NumDestinaire=765620323;

cdr->Duree=87;
```

Séquence 0 : Fondamentaux du C

Manipulation des fonctions

❑ Syntaxe de la déclaration

```
type nomfonctoion(type arg1,type arg2,..., type arg N)

{/début

Bloc d'instructions

return valeurDeretour;

}//fin
void nomfonctoion(type arg1,type arg2,..., type arg N))

{/début

Bloc d'instruction

}//fin
```

Séquence 0 : Fondamentaux du C

Manipulation des fonctions

❑ Syntaxe de la déclaration

```
int addition(type a, type b)

{/début

return a+b;

}//fin


void Afficher(type s)

{/début

printf("%d",s);

}//fin
```

Séquence 0 : Fondamentaux du C

Manipulation des fonctions

❑ Appel d'une fonction(1/2)

```
int addition(int a, int b)

{/début

return a+b;

}//fin


void Afficher(type s)

{/début

printf("%d",s);

}//fin
```

```
Int main ()

{/début

int somme;

int x,y;

x=5;y=10;

somme=addition(x,y);

Afficher(somme);

return 0;

}//fin
```

Séquence 0 : Fondamentaux du C

Manipulation des fonctions

❑ Appel d'une fonction(2/2)

```
int addition(int *a, int *b)
{/début
return *a+*b;
}//fin

void Afficher(type s)
{/début
printf("%d",s);
}//fin
```

```
Int main ()
{/début
int somme;
int x,y;
x=5;y=10;
somme=addition(&x, &y);
Afficher(somme);
return 0;
}//fin
```

Séquence 1 : Descripteur de fichier

Séquence 1 : Descripteur de fichier

❑ Objectifs

- Comprendre le rôle d'un descripteur;
- Manipuler des descripteurs;

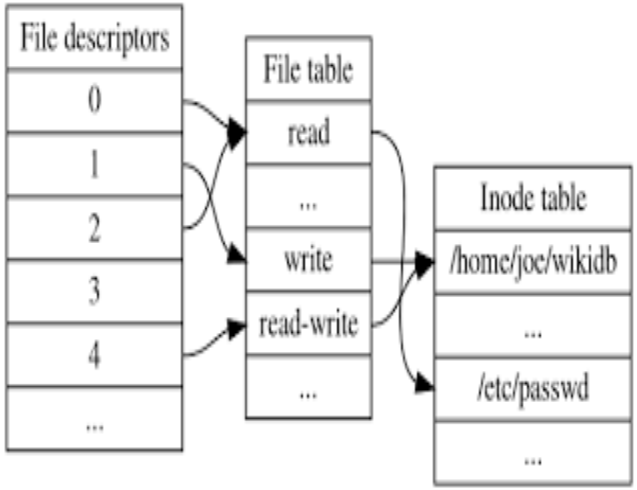
Séquence 1 : descripteur de fichier

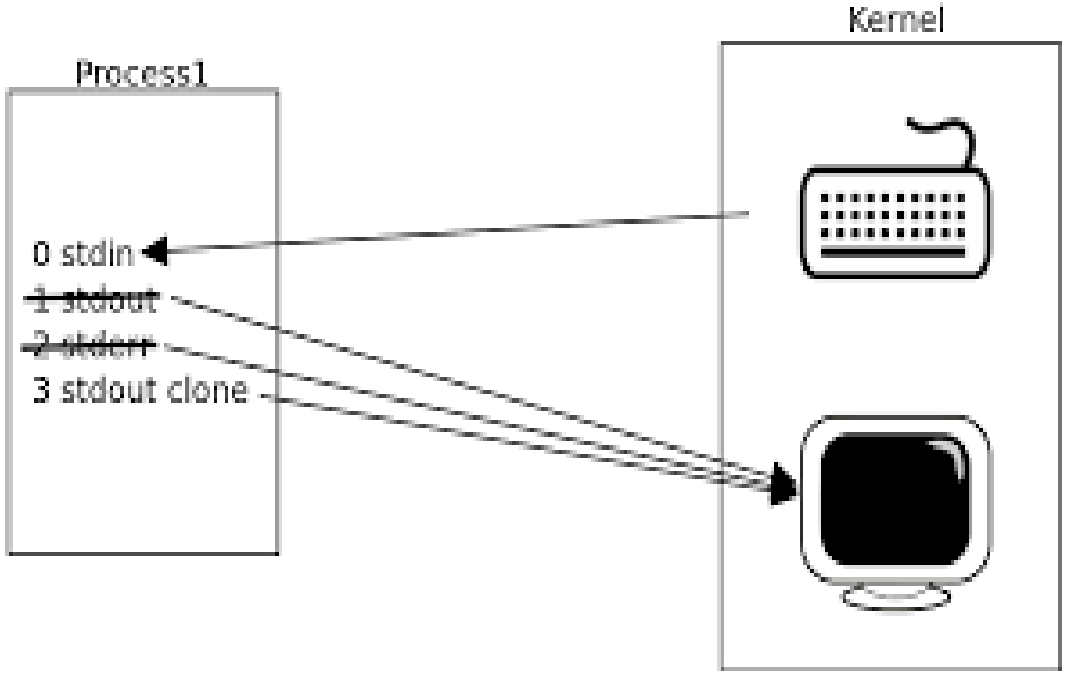
Définition

- **Numéro permettant de référencer un fichier ouvert dans un programme**

Séquence 1 : descripteur de fichier

- **Les descripteurs réservés**





Séquence 2: Appels systèmes de gestion de fichier

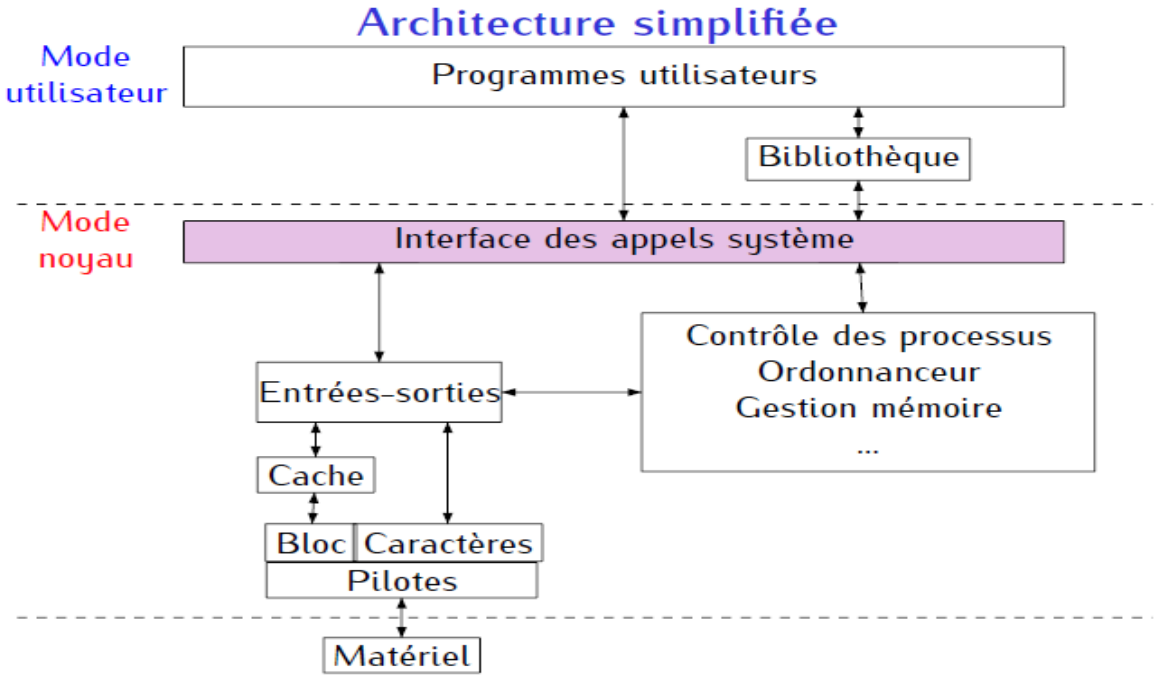
Séquence 2 : Appels systèmes de gestion de fichier

❑ objectifs

- ➢ Créer un descripteur
- ➢ Effectuer des opérations sur les descripteurs

Séquence 2 : Appels systèmes de gestion de fichier

❑ Un appel système



Séquence 2 : Appels systèmes de gestion de fichier

❑ mkdir(2)

```
#include <sys/stat.h>
#include <sys/types.h>

int mkdir(const char *pathname, mode_t mode);
```

Chemin du répertoire

Permission du répertoire

Exemple:

```
#include <sys/stat.h>

#include <sys/types.h>

int main(int argc, char *argv[])

{

mkdir(argv[1],0750);

return 0;

}
```

```
$cc my_mkdir –o my_mkdir

$./my_mkdir rep1
```

Séquence 2 : Appels systèmes de gestion de fichier

❑ rmdir(2)

```
#include <unistd.h>
int rmdir(const char *pathname);
```

Chemin du fichier

Exemple:

```
#include <unisted>
int main(int argc, char *argv[])
{
rmdir(argv[1]);
return 0;
}
```

```
$cc my_rmdir –o my_rmdir
$./my_rmdir rep1
```

Séquence 2 : Appels systèmes de gestion de fichier

❑ creat(2)

```
int creat(const char *pathname, mode_t mode);
```

Chemin du fichier

Permission du fichier

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int main(int argc, char *argv[])
{

creat(argv[1],0644);

return 0;
}
```

$cc my_touch.c –o my_touch

$./my_touch fic1

Séquence 2 : Appels systèmes de gestion de fichier

❑ opendir(3)

```
#include <sys/types.h>
#include <dirent.h>

DIR *opendir(const char *nom);
```

Exemple:

```
#include <sys/types.h>

#include <dirent.h>

int main(int argc, char *argv[])

{

DIR *rep;

rep=opendir(argv[1]);

return 0;

}
```

Séquence 2 : Appels systèmes de gestion de fichier

❑ readdir(2)

```
#include <dirent.h>

struct dirent *readdir(DIR *dirp);
```

La fonction **readdir**() renvoie un pointeur sur une structure dirent représentant l'entrée suivante du flux répertoire pointé par dirp. Elle renvoie NULL a la fin du répertoire, ou en cas d'erreur.

Exemple:

```
#include <sys/types.h>

#include <dirent.h>

int main(int argc, char *argv[])

{

DIR *rep;

struct dirent *contenu;

rep=opendir(argv[1]);

contenu=readdir(rep);

return 0;

}
```

Séquence 2 : Appels systèmes de gestion de fichier

❑ readdir(2)

```
#include <dirent.h>

struct dirent *readdir(DIR *dirp);
```

```
struct dirent {
    ino_t          d_ino;       /* numéro de l'inode */
    off_t          d_off;       /* décalage vers le prochain dirent */
    unsigned short d_reclen;    /* longueur de cet enregistrement */
    unsigned char  d_type;      /* type du fichier ; pas pris en
                                   charge par tous les types de
                                   système de fichiers */
    char           d_name[256]; /* nom du fichier */
};
```

Exemple:

```
#include <sys/types.h>

#include <dirent.h>

int main(int argc, char *argv[])

{

DIR *rep;

struct dirent *contenu;

rep=opendir(argv[1]);

contenu=readdir(rep);

printf("%s", contenu->d_name);

return 0;

}
```

Séquence 3: Opérations d'entrées sorties sur les fichiers

Séquence 3 : Opérations d'entrée sortie sur les fichiers

❑ objectifs

- ➢ Comprendre les descripteurs de fichiers;
- ➢ Lire dans un fichier
- ➢ Écrire dans un fichier

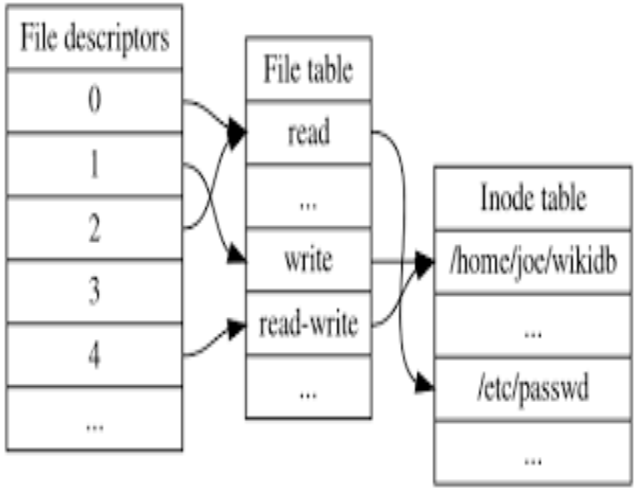
Séquence 3 : Opérations d'entrée sortie sur les fichiers

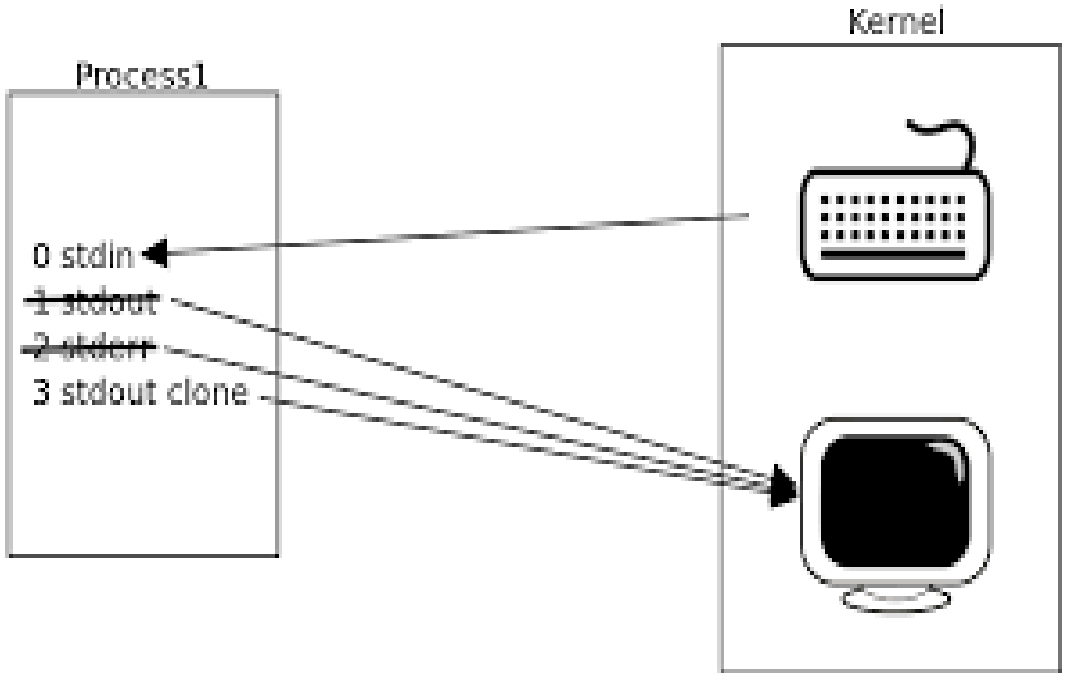
❑ Descripteur de fichier

- C'est un numéro qui sert de référence à un fichier ouvert dans un programme.

Séquence 3 : Opérations d'entrée sortie sur les fichiers

- **Les descripteurs réservés**





Séquence 3 : Opérations d'entrée sortie sur les fichiers

❑ Open(2)

Chemin du fichier

Mode d'ouverture du fichier

Valeur possible:
O_RDONLY
O_WRONLY
O_RDWR

```
SYNOPSIS
       #include <sys/types.h>
       #include <sys/stat.h>
       #include <fcntl.h>

       int open(const char *pathname, int flags);
       int open(const char *pathname, int flags, mode_t mode);
```

Un appel à **open**() crée une nouvelle description de fichier ouvert, une entrée dans la table de fichiers ouverts du système. Cette entrée enregistre la position dans le fichier et les attributs d'état du fichier (modifiables par l'opération **F_SETFL** de **fcntl**(2)). Un descripteur de fichier est une référence à l'une de ces entrées ; cette référence n'est pas modifiée si pathname est ensuite supprimé ou modifié pour correspondre à un autre fichier. La nouvelle description de fichier ouvert n'est initialement partagée avec aucun autre processus, mais ce partage peut apparaître après un **fork**(2).

Séquence 3 : Opérations d'entrée sortie sur les fichiers

❑ Open(2)

Chemin du fichier

Mode d'ouverture du fichier

Valeur possible:
- O_RDONLY
- O_WRONLY
- O_RDWR

```
SYNOPSIS
       #include <sys/types.h>
       #include <sys/stat.h>
       #include <fcntl.h>

       int open(const char *pathname, int flags);
       int open(const char *pathname, int flags, mode_t mode);
```

Descripteur de fichier

Exemple 1:

```
int main(int argc, char *argv[])
{
int fd;
fd= open(argv[1], O_RDONLY);
return 0;
}
$cc my_open –o open_rd
$./open_rd  fic1
```

Exemple 2:

```
int main(int argc, char *argv[])
{
int fd;
fd= open(argv[1], O_WRONLY);
return 0;
}
$cc my_open –o open_wr
$./open_wr fic1
```

Séquence 3 : Opérations d'entrée sortie sur les fichiers

❑ creat(2)

```
int creat(const char *pathname, mode_t mode);
```

Chemin du fichier

Permission du fichier

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int main(int argc, char *argv[])
{

creat(argv[1],0644);

return 0;
}
```

$cc my_touch.c –o my_touch

$./my_touch fic1

Séquence 3 : Opérations d'entrée sortie sur les fichiers

- **read(2)**

```
SYNOPSIS
       #include <unistd.h>

       ssize_t read(int fd, void *buf, size_t count);
```

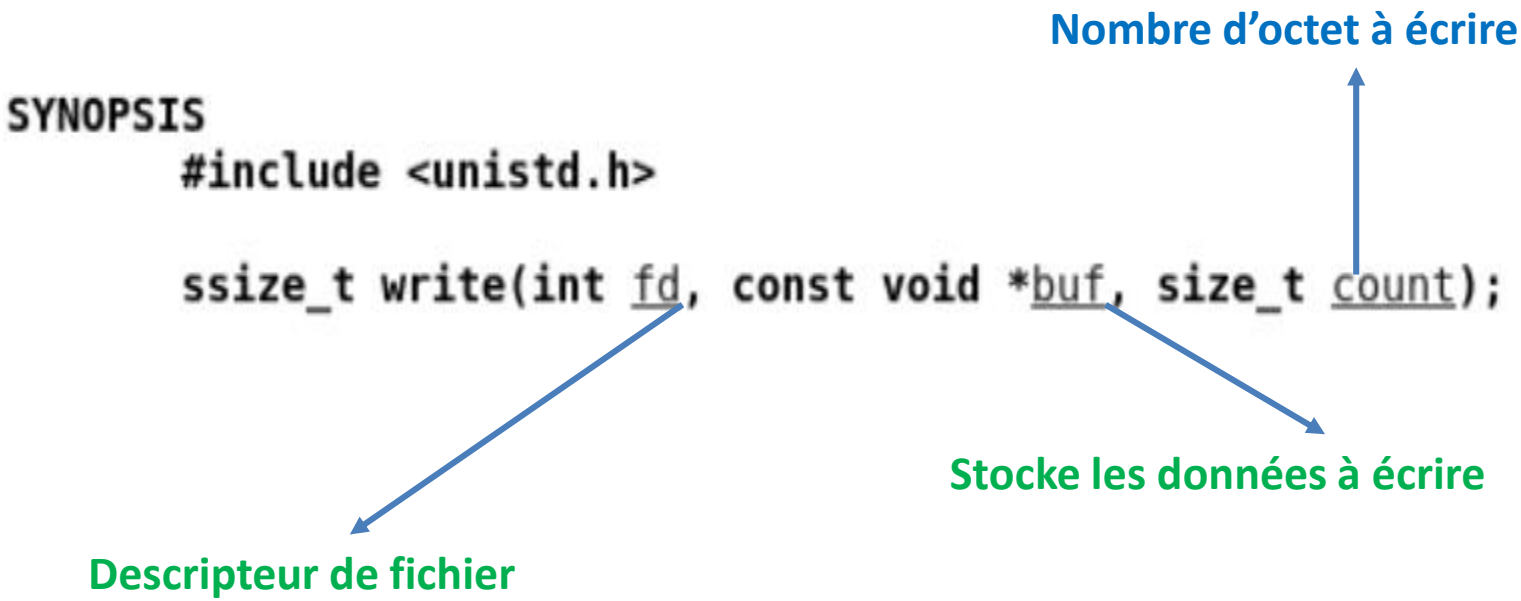
Nombre d'octet à lire

Stocke les données lues

Descripteur de fichier

Séquence 3 : Opérations d'entrée sortie sur les fichiers

- **write(2)**



Séquence 3 : Opérations d'entrée sortie sur les fichiers

read(2)

```
SYNOPSIS
       #include <unistd.h>

       ssize_t read(int fd, void *buf, size_t count);
```

Exemple 1:

```
#include <unisted>
int main(int argc, char *argv[])
{
int fd;
char buf[16];
fd= open(argv[1], O_RDONLY);
read(fd, buf,16);
write(1, buf,16);
return 0;
}
```

```
$cc my_open –o open_rd
$./open_rd  fic1
```

Atelier