TD de stème n° : arcours de ré ertoires – fork, ait, exit

Ex rcic 1 :

Écrire une commande qui prend en argument le chemin d'un répertoire et affiche la tructure de l'arbore cence de fichier à partir du répertoire donné. Exemple d'affichage :

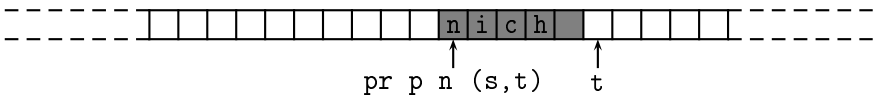
```
 Y TEME
| TD7
  | t 7.p f
  | t 7.t x
| TP7
  | tp7.c
  | tp7.p f
  | tp7.t x
```

Ex rcic 2 : « pw »

Le but de cet exercice e t de réécrire la commande « `pw` » qui affiche le chemin du répertoire courant.

Pour cela, on remontera dan l'arbore cence ju qu'à la racine, en recherchant à chaque étape le numéro d'inœud du répertoire courant dan on répertoire-père pour déterminer on nom. Par ouci de implification, on uppo era que l'arbore cence n'e t con tituée que d'un eul di que logique *e ce* , et que la racine de l'arbore cence e t donc le eul répertoire dont le numéro d'inœud e t 2.

n uppo e qu'on di po e d'une fonction `char *pr p n (const char *s, char *t)` in- érant une chaîne `s` dan un tableau de caractère , dan le ca e *p écé t* celle pointée par `t`, et renvoyant l'adre e de la première ca e de la copie de `s` ; la zone néce aire doit avoir été précédemment allouée ; par exemple, i `s "nich "` :



n pourra au i utili er le fonction uivante :

- `DIR *op n ir(const char *nam );`
- `struct ir nt *r a ir(DIR * ir);`
- `int clos ir(DIR * ir);`
- `int ch ir(const char *path);`

n rappelle de plu le champ uivant de tructure que vou pourrez utili er :

```
struct stat
  ino_t     st_ino;      * ino   numb r *
  ...
 ;
```

et

```
struct  ir nt
  ino_t      _ino;        * ino   numb r *
  char       _nam [2   ;  * fil nam  *
  ...
 ;
```

Dan la uite, on utili era le fonction uivante :

`pi _t fork(voi );`

- crée un clone du proce u appelant duplique tout l'e pace d'adre age ;
- renvoie : le PID du fil dan le proce u père, 0 dan le proce u fil ;
- renvoie -1 en ca d'erreur.

`pi _t wait(int *status);`

- attend la terminai on d'un proce u fil ;
- renvoie le PID du fil qui 'e t terminé ou -1 en ca d'erreur ;
- quand `wait` e débloque `status` contient la valeur du code de retour du fil : l'octet de poid faible e t un code indiquant pourquoi le proce u fil 'e t arrêté, et i le proce u fil a effectué un appel à `xit`, l'octet précédent contient le code de retour.

`voi xit(int status);`

- termine l'exécution d'un proce u et demande au y tème de le upprimer ;
- la valeur `status` pécifie un code de retour compri entre 0 et 255.
- la valeur `status` e t pa ée au père 'il e t en attente ur un `wait`.

Ex rcic 3 :

ue fait le programme uivant ?

```
#  fin  N 10
int main()
  int i   1;
  whil  (fork()    0 && i    N) i  ;
  printf("   n",i);
   xit(0);
```

Ex rcic 4 :

ombien de « meuh ! et de « coin ! affiche le programme uivant ?

```
int main()
  int i;
  for (i 0 ; i 4 ; i  )
    int pi    fork() ;
    if (pi  !  0)
      printf("m uh ! n") ;
    printf("coin ! n") ;

   xit(0);
```

Ex rcic : ch rch arall li an un ta l au

n ouhaite trouver toute le occurrence d'un entier dan un tableau de grande taille. Pour cela, on coupe le tableau en tableaux de 256 entier et on effectue la recherche indépendamment dan tou le ou -tableaux en le confiant à de proce u différent travaillant en parallèle.

Écrire un programme effectuant la recherche de cette manière ; le programme devra afficher le nombre d'occurrence de l'élément dan chaque ou -tableaux, pui le nombre total d'occurrence de l'élément dan le tableau.