Voici un programme de démonstration d'un tube

.

Vous devez en faire l'essai et remettre les programmes source réalisé et les programmes exécutables dans un fichier compressé.

```
/* version triviale d'utilisation d'un tube */

main()
 {
  int pfd[2], nread;
  char s[100];
  /* si création du tube pas réussi, retourne -1 */
  if (pipe(pfd) ==-1)
    { perror("pipe") ; exit(1) ;  }
  /* on écrit dans [1] */
  if (write(pfd[1],"Allo toi !",11) ==-1)
    { perror("write"); exit(1) ; }
  /* on lit dans [2] */
  switch(nread=read(pfd[0],s,sizeof(s)))
    {
     case -1:
         perror("write");
         exit(1);
     case 0:
         printf("EOF\n");
         exit(1);
     default:
         printf("%d octets lus dans le tube: %s\n",nread,s);
    }
 }
```

```
/* démo utilisation de pipe */
#include <unistd.h>
#include <stdio.h>
main()
{
    int tube[2];
    int retexec;
    int status;
    int ret;
    pipe(tube);
    /* vfork est une version allégée de fork... */
    if (vfork() == 0) {
        printf("enfant\n");
        close(tube[1]);    /*Ferme le tube en écriture. */
        close(0);          /*Ferme l'entrée standard. */
        dup(tube[0]);      /* Accepte l'entrée standard à partir du tube. */
        retexec = execl("/bin/sort", "sort", "-r",  (char *) 0);
        /* execl("chemin pgm à exécuter", "nom pgm", ["arg 1"], ["arg..."], (char *) 0) */
        printf("erreur exec\n");
        /* si exec fonctionne, le programme enfant se termine ici !  */
        /* si exec ne fonctionne pas, exec retourne -1 dans retexec  */
        printf("retour erreur exec %d\n", retexec);
        exit(5);
    }
    printf("Retour au parent\n");
    close(tube[0]) ;      /* Ferme le tube en lecture. */
    close(1);             /* Ferme la sortie standard. */
    dup(tube[1]); /* Dirige la sortie standard vers le tube. */
    retexec = execl("/bin/ps", "ps", "-l", (char *) 0);
    printf("retour exec parent = %x\n", retexec);
}
```