PPS

TD ① du 12/11

Exo 7:

```
void waitfg ( pid_t pid ) {
    sigset_t mask, mask_normal;
    struct job_t * j = job_get_pid(pid);

    if (j == null) return;

    while ( j -> state == FG && (j -> pid = pid) ) {

        sigsuspend( &mask ); // attente active
                             // code moche
    }

    sigprocmask( SIG_SETMASK, &mask_normal, &mask )

}

void do_fg( char ** argv ) {
    struct job_t *j = treat_argv( argv );
    if ( j == null) return ;

    kill ( j -> pid, SIGCONT );
    j -> state = FG;
    waitfg( j -> pid );

}
```

sigemptyset(&mask)
sigaddset (&mask, SIGCHILD)
{
sigprocmask(SIG_BLOCK, &mask, &mask_normal)

Exo 8

```
void do_exit(void) {

  if ( jobs_getstopped() ) {
      printf( "there is suspend jobs \n");
      return;

  }else{
    exit (EXITSUCCESS);
}
```

Exo 10 : #define SIZE

```
int nboctetlus = 0;
char buff[SIZE];


int nbocteok = 0;
int fd[2];
assert ( pipe (fd) == 0);
if ( fork() == 0) {
      close (fd[0]);
      parle (fd[1]);
      exit ( EXIT_SUCCESS);
else {
    close ( fd[1]);
    while( 1 == 1){
        nb_octet lus = read( fd[0], buff, SIZE)
        if( nb octet lus <= 0) break;
        nb_octet += nb_octetlus;
        Write ( STDOUT_FILENO, buf, nb_octetlus);
}
```

PPS TD ② du 12/11/18

Ps | grep toto

```
int main(void){
    int fd[2];
    assert(pipe(fd)==0);
    switch(fork()){
        case -1: return -1;
        case 0:  close(fd[1]);
                 dup2(fd[0],1);
                 close(fd[0]);
                 execl("Ps","Ps",NULL);
                 break;
        default: close(fd[0]);
                 dup2(fd[1],0);
                 close(fd[1]);
                 execl("grep","grep","toto",NULL);
    }
```

Ps