Tube anonyme = fork forcément avant.

Communication bidirectionnelle : Deux pipes.

Exercice 1 du contrôle :

Processus 1 : Tube 1 qui communique à processus 2.

Processus 2 qui renvoie l'information au processus 1 par un tube 2. Et affiche l'information

Solutions possibles :

Solution A : Soit deux tubes nommées (Pas de lien entre P1 et P2)

Solution B : deux tubes anonymes et fork (si P1 et P2 sont apparentés).

P1 va ouvrir T1 en écrire, P2 va ouvrir T1 en lecture.

P2 va ouvrir T2 en écriture, P1 va ouvrir T2 en lecture.

Exercice 2 du contrôle :

Solution A ne marchera pas.

Solution B peut fonctionner si on est sous windows mais pas sous linux.

Explication : Ce n'est pas un vrai fichier sous linux donc la communication ne marchera pas, c'est dans une zone mémoire sauf que dans une zone mémoire ça ne fonctionne pas car elles ne sont pas partagées. Ça ne passera pas par un tube si on utilise réseau.

Exercice 3 :

```c
int main()
{
    while(1){
        int pid;
        printf("$");
        scanf("%s\n",cmd);
        switch(pid = fork()):
        case 0:
            exec(cmd);
            perror("exec");
        default:
            waitpid(pid);
    }

    return 0;
}
```

Exercice 4 :

```c
#define MAX = 255;
int main(int argc, char* argv[]){
    if(argc<3) exit(-1);
    int tube[2];
    char* buffer[MAX];
    pipe(tube);
    switch(fork()){
        case 0:
            fd=open(argv[1],O_RDONLY);
            close(tube[0]);
            while((n=read(fd,buffer,MAX))!=0){
                write(tube[1],buffer,n); // ou aussi for(int i=0;i<n;i++){ write(tube[1],buffer[i],1); }
             }
            close(fd);
            close(tube[1]);
            break;
        case default:
            close(tube[1]);
            fd = open(argv[2];O_WRONLY|O_CREAT|O_TRUNCT);
            while((n=read(tube[0],buffer,MAX))!=0){
                write(fd,buffer,n);
            }
            close(fd);
            close(tube[0]);
            exit(0);
    }  wait(0);
}
```

Exercice 5 :

Primitives :

Création processus : Fork

Recouvrement : exec

Destruction : suicide exit / meurtre : kill

Synchro directe :

signaux tube création pipe/lecture read/écriture write/fermeture close

Fichier :création open / lecture read / write / close