PSED

I. Question de cours

1. Non, il reinitialise tte les entree sauf 0, 1 et 2, donc les descripteur déjà ouvert
   - Pour ne pas etre utilisé par un programme, et pour liberer les descripteurs
   - On garde 0, 1 et 2 pour les redirections
2. On ne peut pas savoir, ça depent de l'ordonanceur, sauf en cas de synchronisation
3. Non car on aurait pas l'unicite de l'identifiant du processus
4. Un seul close ne ferme pas les 3 d'un coup, il agir sur un descripter, pas un fichier
5. Oui si ce n'est pas un lien symbolique (ou fichier inexistant)

II. Exercice sur les fichiers

```
Main(int argc, char * argv[]){
        If(argc != 4) exit(1) ;
        If ! access(argv[1], R_OK) exit(2) ;
        Int i, int j, int k, char * chaine, struct stat buf ;
        i = stat(argv[1], buf) ;
        if(i != 0) exit (3) ;
        if(S_ISREG(buf.st_mode)){
                if(buf.st_size < atoi(argv[3])) exit(4) ;
                j = open(argv[1], 0_RDWR) ;
                if(j>0){
                        k = lseek(j, atoi(argv[3]), SEEK_SET) ;
                        chaine = (char*) malloc(strlen(argv[2])
                        if(strlen(argv[2]) > (buf.st_size – atoi(argv[3]){
                                read(j, chaine, buf.st_size – atoi(argv[3]) ;
                                k = lseek(j, strlen(argv[2]) ;
                        else
                                read(j, chaine, strlen(argv[2])) ;
                                k = lseek(j, buf.st_size – atoi(argv[3]), SEEK_CUR) ;
                        write(1, chaine, strlen(chaine) ;
                        write(j, argv[2] , strlen(argv[2]) ;
                }
        }
        else
        {
                if(S_ISLNK(buf.st_mode) || S_ISDIR(buf.st_mode)){
                        write(2, argv[2], strlen(argv[2]) ;
                        write(2,argv[3] , strlen(argv[3]° ;
                        if(S_ISLNK(buf.st_mode) write(2, « LIEN », strlen(« LIEN ») ;
                        else write(2, « DIR », strlen(« DIR ») ;
                }
        }
}
```

III. Exercice sur les processus

```
Main(int argc, char *argv[]){
        Int i, j ;
        i= (argc – 1) div 3 ;
         for(j = 1, j <= i, j++){
                res = fork() ;
                if(res == 0){
                   execlp(« remplacer », « remplacer », argv[i], argv[i+1], argv[i+2], NULL) ;
                }
                else
                {
                   i += 3 ;
                   wait(&k) ;
                   if WIFEXITED(k){
                         if(WIFEXITSTATUS(k))
                   }
                   l = l+1 ;
                }
        }
        if(l < i)printf(« erreur\n ») ;
}
```

IV. Exercice sur les redirections

```
Main(int argc, char * argv[]){
        Int i, j, char * argv2[argc], k, l ;
        i = open(« /tmp/res », O_WRONLY | O_CREAT, 777) ;
        j = open(« /tmp/err », O_WRONLY | O_CREAT, 777) ;
        dup2(i, 1) ;
        dup2(j, 2) ;
        k = 1, l = 1 ;
        while(k < argc)
        {
                argv2[k] = argv2[l] ;
                l++ ;
                k++ ;
        }
        argv2[0] = « nremplacer » ;
        execvp(nremplace, argv2) ;
```