UFES - DEPARTAMENTO DE INFORMÁTICA

3ª. Prova de Sistemas Operacionais - Período: 2016/1 - Profª. Roberta Lima Gomes

1) **(4,0)** Desenvolva um mecanismo baseado em troca de mensagens permitindo que processos *Clientes* possam realizar operações "down()" e "up()" em um *Mutex*, gerenciado por um processo *Servidor*. Para tando, descreva em código (ou pseudo-código) as rotinas "down()" e "up()" dos Clientes, e a rotina "simulaMutex()" do Servidor.

```
//Código dos Clientes
void down() {
     ...
}

void up(){
     ...
}
```

```
//Código do Servidor
void simulaMutex(){
    while(1){
           …
     }
}
```

2) **(3,5)** Sobre o código a seguir, responda:

```
#define SIZE 6
#define READ 0
#define WRITE 1
main()
{
    pid_t pid1, pid2, pid3, pid4; int status;
    int fd[2]; char buffer[SIZE+1];
    struct rusage usage;
    pipe(fd);
    if ((pid1=fork())==0) {          // child 1
      while(1) {
         read(fd[READ], buffer, SIZE);
         buffer[SIZE]='\0';
         write(fd[WRITE], "tomato", SIZE);
      }
    } else if ((pid2=fork())==0) {   // child 2
      while(1) {
         read(fd[READ], buffer, SIZE);
         buffer[6]='\0';
         write(fd[WRITE], "turnip", SIZE);
      }
    } else {  // parent
      write(fd[WRITE],"potato",SIZE);
      fprintf(stderr,"Parent: I wrote a potato!\n", buffer);
      sleep(10);
      read(fd[READ], buffer, SIZE);  buffer[SIZE]='\0';
      fprintf(stderr,"Parent: I got back a %s!\n", buffer);
      kill(pid1, SIGINT); pid3 = wait(&status, 0, &usage);
      kill(pid2, SIGINT); pid4 = wait(&status, 0, &usage);
    }
}
```

a) O que este código faz?
b) É possível prever qual será a saída do programa? Explique.
c) O que acontece se omitirmos o comando "write()" de um dos filhos?

3) **(2,5)** Quais as diferenças entre threads em nível de usuário e threads em nível de kernel? Quais as vantavens/desvantagens de cada um desses tipos?

Boa Prova !!!