FSO - Trabalho 3

Na fase 1 queriamos que todos os processos executassem as 5 iterações todas de uma vez, dando ao processo seguinte a possibilidade de fazer o mesmo, apenas depois de o primeiro acabar, ou seja produzir o resultado:

Resultado 1
<100> mensagem 1
<100> mensagem 2
<100> mensagem 3
<100> mensagem 4
<100> mensagem 5
<101> mensagem 1
<101> mensagem 2
<101> mensagem 3
<101> mensagem 4
<101> mensagem 5

Para isso criamos um semáforo na main

```
exmut=createSem(1);
for( i = 0; i < nfilhos; i++ )
        if( fork() == 0 )
        {       nim(i);
                exit(0);
        }
for( i = 0; i < nfilhos; i++ )
        wait(NULL);
printf( "fim!\n" );
deleteSem( exmut );
```

e na função nim :
chamada de P(exmut) antes do for e chamada de V(exmut) depois do for.

```
P(exmut);
        for( n = 1; n <= N_NIM; n++ )
        {
                printf("Filho %d - <%d> Mensagem %d\n", i, pid, n );
/*      para mono processadores acrescentar a instrução
                usleep(100);
*/
        }
V(exmut);
```

Na 2ª fase do trabalho queriamos que cada um dos processos escrevesse a mensagem i-1 e só depois cada um deles iria escrever a mensagem i.

Resultado 1
<100> mensagem 1
<101> mensagem 1
<101> mensagem 2
<100> mensagem 2
<100> mensagem 3
<101> mensagem 3
<100> mensagem 4
<101> mensagem 4
<101> mensagem 5
<100> mensagem 5

Utilizámos uma barreira.Começámos por criar uma zona de memória partilhada .
Depois fizemos a ligação da zona de sharemem ao processo e atribuição de um identificador ao apontador de countPtr.Utilizámos 2 semáforos, exmut e block, para controlo de entrada e bloqueio de processos na barreira .Com a combinação de tudo garantimos que os processos ficassem bloqueados na barreira até ao ultimo chegar os libertasse para irem para a proxima barreira.

Trabalho realizado por:
Pedro Miranda