

INE 5410 - Laboratório

AULA 03 – SEMÁFOROS NO PASCAL-FC

1. INTRODUÇÃO

Nas aulas anteriores, foi mostrada a necessidade de se ter mecanismos para controlar a concorrência entre os processos. O não determinismo e a competição por recursos da plataforma de execução são fonte de conflito entre processos ou threads e, à medida que o sistema vai crescendo em complexidade, as dificuldades em controlar a concorrência aumentam, podendo gerar problemas diversos na operação do sistema e na gestão do acesso a recursos.

2. OBJETIVO DA AULA

Nesta aula prática, serão realizados alguns exercícios cujo objetivo será experimentar o uso de um dos mecanismos mais clássicos de controle da concorrência – o semáforo. A implementação deste mecanismo na linguagem Pascal-FC é clara e simples, podendo ser aplicada a diversos cenários de controle da concorrência, como alguns vistos na parte teórica da matéria.

3. EXEMPLOS

3.1 Gerenciamento de uma piscina

Neste primeiro programa, a preocupação é a alocação de recursos em exclusão mútua numa piscina onde um banhista só pode entrar para nadar após ter conseguido acesso a uma cabine e, antes disso, a um cesto para guardar suas roupas. Na área da piscina, estão disponíveis apenas 8 cestos e três cabines de modo que haverá competição pelos mesmos. A seqüência de execução de um banhista é chegar na piscina, acessar um cesto, entrar na cabine para trocar de roupa e entrar na piscina pra nadar; ao terminar de nadar, o banhista acessa uma cabine para se trocar e libera a cabine e o cesto de roupa.

A listagem do programa é apresentada a seguir, o qual será comentado a partir de agora. Cada banhista será representado por um processo que vai competir com os demais pelos diversos recursos (principalmente, cestos e cabines). O acesso a estes dois tipos de recursos será controlado por semáforos (no programa, representados pelas variáveis *Cesto* e *Cabine*). Um terceiro semáforo é definido para controlar o acesso exclusivo à tela (impressão de informações), de modo que, a cada vez que um processo deseja imprimir algo na tela, terá de obter acesso controlado por este semáforo, denominado *Tela*.

A execução dos processos é bastante simples, sendo que a cada etapa da execução, este imprime seu status para que se possa verificar a dinâmica do processamento concorrente e a eficiência do controle de acesso aos diversos recursos.

```

PROGRAM Piscina;
VAR
    Cesto,Cabine,Tela : semaphore;
PROCESS Type TPbanhista(NM : integer);
BEGIN
    WHILE (TRUE) DO
        BEGIN
            WAIT(Tela);
            WRITELN(NM:3,'CHEGANDO');
            SIGNAL(Tela);
            SLEEP(RANDOM(15)+5);
            WAIT(Cesto);
            WAIT(Cabine);
            WAIT(Tela);
            WRITELN(NM:3,'DESPINDO');
            SIGNAL(Tela);
            SLEEP(RANDOM(10)+3);
            SIGNAL(Cabine);
            WAIT(Tela);
            WRITELN(NM:3,'NADANDO');
            SIGNAL(Tela);
            SLEEP(RANDOM(10)+3);
            WAIT(Cabine);
            WAIT(Tela);
            WRITELN(NM:3,'VESTINDO');
            SIGNAL(Tela);
            SLEEP(RANDOM(10)+3);
            SIGNAL(Cabine);
            SIGNAL(Cesto);
        END;
    END;

VAR
    Banhista : ARRAY[1..20] OF TPbanhista;
    I : integer;
BEGIN
    INITIAL(Cabine,3);
    INITIAL(Cesto,8);
    INITIAL(Tela,1);
    COBEGIN
        FOR I := 1 TO 20
            DO Banhista[I](I);
        COEND;
    END.

```

3.2 Produtores-Consumidores

Neste programa será feito o uso de semáforos para controlar o acesso a um buffer de mensagens (com capacidade de apenas uma mensagem) que é acessado por processos *produtores* (inserir mensagens no buffer) e processos *consumidores* (consomem mensagens do buffer). Em termos de execução, os processos *produtores* podem acessar o buffer para salvar mensagens no buffer, sinalizando para os processos

consumidores a presença de mensagens. Os processos *consumidores* devem acessar o buffer para recuperar as mensagens e imprimir seu identificador, o identificador do processo *produtor* que gerou a mensagem e o conteúdo da mesma (um número, entre 1 e 1000, gerado aleatoriamente pelo produtor no momento de armazenar a mensagem no buffer).

```

PROGRAM ProdCons;
TYPE tpmsg = RECORD
    nprod: INTEGER
    cod  : INTEGER
END;
(*=====*)
VAR
    buffer          : tpmsg;
    Svaga, Smsg, tela : semaphore;

(*=====*)

PROCEDURE colocaMsg (VAR mens : tpmsg);
BEGIN
    wait(Svaga);
    buffer := mens;
    signal(Smsg);
END;

(*=====*)

PROCEDURE retiraMsg (VAR mens : tpmsg);
BEGIN
    wait (Smsg);
    mens := buffer;
    signal(Svaga);
END;

(*=====*)

PROCESS TYPE tpProdutor ( pid: integer );
VAR
    msg : tpmsg;
BEGIN
    REPEAT
        sleep ( random ( 10 ) + 5 );
        msg.nprod := pid;
        msg.cod   := random ( 1000 )+ 10;
        colocaMsg ( msg );
    FOREVER;
END;

(*=====*)

PROCESS TYPE tpConsumidor ( pid: integer );
VAR
    msg : tpmsg;
BEGIN

```

```

    REPEAT
        retiraMsg ( msg );
        sleep ( random ( 10 ) + 2 );
        wait ( tela );
        writeln('- C: ',pid:2, '- P: ', msg.nprod:2,' MSG:
', msg.cod:4 );
        signal ( tela );

    FOREVER;
END;

( *=====*)

VAR
    prod1, prod2, prod3      : tpProdutor;
    cons1, cons2, cons3, cons4 : tpConsumidor;

( *=====*)

BEGIN
    initial ( tela , 1 );
    initial ( Smsg , 0 );
    initial ( Svaga, 1 );
    COBEGIN
        prod1( 1 ); prod2( 2 ); prod3 ( 3 );
        cons1( 1 ); cons2( 2 ); cons3 ( 3 ); cons4 ( 4 );
    COEND;
END.

```

4. EXERCÍCIOS

Desenvolver e executar as soluções dos seguintes problemas.

4.1 Acesso a mesas num restaurante

Desenvolver um programa que represente o controle de acesso a mesas de um restaurante. Os clientes devem ser representados por processos que disputam as 12 mesas do estabelecimento. Utilize comandos de impressão para representar o estado dos clientes, os quais poderão estar caminhando ou comendo no restaurante. Ao fim da refeição, o cliente deverá liberar a mesa para permitir que outros clientes iniciem suas refeições.

4.2 Controle de Acesso a um Arquivo por vários processos

Escrever e executar um programa que faça o uso de semáforos para controlar o acesso a um arquivo da parte de diversos processos. A limitação para o acesso a este arquivo é que a soma dos identificadores dos processos não deve ultrapassar o valor 40.