With Ada.Integer\_Text\_IO, Ada.Text\_IO, Ada.Calendar;

Use Ada.Integer\_Text\_IO, Ada.Text\_IO, Ada.Calendar;

procedure sincronizacao is

protected type Resource is

entry Seize;

procedure Release;

private

Busy : Boolean := False;

end Resource;

protected body Resource is

entry Seize when not Busy is

begin

Busy := True;

end Seize;

procedure Release is

begin

Busy := False;

end Release;

end Resource;

canal : array(1..6) of Integer;

vetorResource : array(1..6) of Resource;

function send(buf : in Integer; indice : in Integer) return integer;

function receive(indice : in Integer) return Integer;

function send(buf : in Integer; indice : in Integer) return integer is

begin

canal(indice) := buf;

vetorResource(indice).Release;

return 0;

end send;

function receive(indice : in Integer) return Integer is

begin

vetorResource(indice).Seize;

return canal(indice);

end receive;

task type Mostra1 is

--entry mostra(Item: in Integer);

end Mostra1;

task body Mostra1 is

valor : Integer := 10;

rec : Integer;

begin

delay 2.0;

rec := send(valor, 1);

delay 2.0;

rec := send(valor, 1);

end Mostra1;

task type Mostra2 is

-- entry mostra(Item: in Integer);

end Mostra2;

task body Mostra2 is

buf1 : Integer;

buf2 : Integer;

I : Integer;

begin

delay 1.0;

buf1 := receive(1);

Put(buf1);

delay 1.0;

buf2 := receive(1);

Put(buf1+buf2);

end Mostra2;

A : Mostra1;

B : Mostra2;

begin

vetorResource(1).Seize;

vetorResource(2).Seize;

vetorResource(3).Seize;

vetorResource(4).Seize;

vetorResource(5).Seize;

vetorResource(6).Seize;

end sincronizacao;