Esercizio di Sincronizzazione tra Processi: il Deposito Bagagli
Esercizio d'Esame del 7 Novembre 1996

Si supponga di avere un **deposito bagagli** composto da V vani ognuno dei quali in grado di contenere N valigie. Gli utenti arrivano con un numero variabile (ma minore di N) di valigie, le depositano e, dopo un certo tempo, le ritirano. Tutte le valigie di uno stesso utente devono essere depositate all'interno di un unico vano, ma uno stesso vano può contenere le valigie di più utenti. Gli utenti che non riescono a depositare le valigie per problemi di capacità si pongono in attesa che si liberi dello spazio.
Si discutano le possibili politiche di gestione del deposito bagagli e se ne implementi una usando il costrutto **monitor**. Si discuta se nella politica implementata sono possibili casi di starvation e, in caso affermativo, si propongano soluzioni per eliminarli.

1° Soluzione

La struttura dei processi “utenti” vedrà una fase di acquisizione e una di rilascio, che si tradurranno nell’invocazione di due procedure entry di un monitor “deposito_bagagli” che gestisce la sincronizzazione.

I processi utenti dovranno prima accedere al deposito per verificare la disponibilità di spazio rispetto al numero di bagagli che essi portano. Se c’è spazio per il deposito (in uno qualsiasi dei V vani) dovranno aggiornare le variabili del monitor, altrimenti dovranno sospendersi in attesa che si liberi dello spazio. Poiché esistono più vani, i processi che riescono a depositare i bagagli dovranno anche sapere in quale dei V vani sono stati messi i loro bagagli.
Quando un processo bagaglio si sospende perché non c’è spazio, non sa ancora in quale vano verranno prima o poi depositati i suoi bagagli, perciò dovrà ogni volta che viene risvegliato fare un ciclo sui vani per vedere se e in quale andare a mettere i bagagli.

Quando vogliono ritirare i bagagli i processi dovranno anche rendere noto in quale dei vani era stato depositato il loro bagaglio, per permettere di aggiornare in modo corretto le variabili del monitor.

Stabilito ciò, è chiaro che il monitor “deposito_bagagli” dovrà avere come sue varabili di stato:

- un array di V elementi per tenere lo stato di occupazione di ogni vano;
- ameno una varabile condition su cui sospendere i processi;

La soluzione sicuramente più semplice è quella che va sospendere un processo se non c’è posto da nessuna parte, e fa risvegliare tutti i processi sospesi ogni qual volta un processo libera delle risorse: notiamo che serve solamente una coda di sospensione! Per realizzare questa soluzione evitando problemi di “livelock” (il solito problema dei cicli while di risveglio e risospensione infinita) bisogna però tenere conto di quanti processi sono sospesi ad un certo istante sulla coda e risvegliarne quel numero e basta.

```
type Bagagli        1..N;
type Vano           1..V

type process utente (Bagagli  n_bagagli)
var quale_vano : Vano;
begin
        deposito_bagagli.LASCIA(quale_vano, n_bagagli);
        <fa un po' quello che ti pare....>
        deposito_bagali.PRENDI(quale_vano, n_bagagli);
end

type monitor deposito_bagagli
var     cap_vano : array[Vano] of Bagagli;
        coda : condition;
        contatore_sospesi: integer;

procedure entry LASCIA (var quale_vano : Vano , n_bagagli : Bagagli)
var i : integer;
begin

quale_vano := 0;

while quale_vano = 0 do
        begin
                for i:=1 to V do
                        if cap_vano[i] + n_bagagli  <= N do
                                                quale_vano := i;
                if quale_vano = 0 do
                        begin
                                contatore_sospesi = contatore_sospesi + 1;
                                coda.wait;
                                contatore_sospesi = contatore_sospesi - 1;
                        end;
        end;

cap_vano[quale_vano] = cap_vano[quale_vano] + n_bagagli;

end;
```

```
procedure entry PRENDI (quale_vano : Vano , n_bagagli : Bagagli)
var i, s : integer;
begin

cap_vano[quale_vano] := cap_vano[quale_vano] - n_bagagli;
s := contatore_sospesi;
for i:=1 to s do
            coda.signal;

end

begin {inizializzazione del monitor }
        for i:=1 to V do
                cap_vano[i] := 0;
end;
```

2° Soluzione

La soluzione è quasi uguale a quella di prima ma con una piccola modifica che rende più giusta la gestione dell'assegnamento dei vani, provvedendo alla analisi dei posti nei vani in modo ciclico. Bisogna introdurre un'altra variabile.

```
type monitor deposito_bagagli
var     cap_vano : array Bagagli of Vano;
        coda : condition;
        contatore_sospesi: integer;
        alterna_vano : Vano;
procedure entry LASCIA (var quale_vano : Vano , n_bagagli : Bagagli)
var i : integer;
begin

quale_vano := 0;

while quale_vano = 0 do
        begin
                for i:= alterna_vano to (alterna_vano +V) do
                        if cap_vano[i MOD V] + n_bagagli  <= N do
                                                quale_vano := i;
.....
alterna_vano := (alterna_vano + 1) MOD V;
end;
```