```
  entry P;
  entry V;

end Semaphore;
```

Tarkoitettu käyttö on siis sellainen, että aluksi kutsutaan semaforin porttia Start, jolla annetaan semaforille alkuarvo. Tämän jälkeen sovellukset kutsuvat semaforin portteja P ja V normaaliin tapaan. V-operaatio voi vapauttaa minkä tahansa odottajista, eli jonotusta ei tarvitse tehdä minkään tietyn algoritmin mukaan.

Tehtäväsi on kirjoittaa Semaphore-tehtävätyypin runko-osa.

4. Ohessa on yksi ratkaisu tuottaja-kuluttaja-ongelmaan. Tuottajaprosessi kutsuu aliohjelmaa Produced, kun se on tuottanyt yhden alkion. Vastaavasti kuluttajaprosessi kutsuu aliohjelmaa Consumed, kun se haluaa uuden alkion kulutettavaksi.

```
type Item_type is private; -- Sisältö ei siis kuulu meille
Buffer_size : constant := 100;
buffer : array 1..Buffer_size of Item_type;
in_index : integer := 1;
out_index : integer := 1;

-- Tätä kutsutaan ennen kuin aliohjelmia
-- Produced ja Consumed aletaan käyttää
procedure Initially is
begin
  null;  -- Tässä versiossa ei tehdä mitään.
end;

procedure Produced (item : Item_type) is
begin
  buffer (in_index) := item;
  in_index := (in_index mod Buffer_size) + 1;
end;

procedure Consumed (item : out Item_type) is
begin
  item = buffer (out_index);
  out_index = (out_index mod Buffer_size) + 1;
end;
```

Oleta, että käytettävissäsi on edellisessä tehtävässä esitelty semafori. Saat siis uuden semaforin käyttöösi kirjoittamalla

```
uusi_semafori : semaphore;
```

Lisää välttämättömät (turhat katsotaan virheeksi) semaforit ja semaforikutsut, jos

a. Puskuri ei voi koskaan täyttyä; tuottajia ja kuluttajia on vain yksi.
b. Puskuri ei voi koskaan täyttyä: tuottajia ja kuluttajia on monta.
c. Puskuri voi täyttyä; tuottajia on kaksi ja kuluttajia on yksi.