```
package Semafor_Bin is

   protected type Semafor_Bin(N: Natural := 1) is
      entry PB;
      procedure VB;
   private
      count: natural := N;
   end Semafor_Bin;
   type Semafor_Bin_Acc is access all Semafor_Bin;

end Semafor_Bin;
```

```
package body Semafor_Bin is

   protected body Semafor_Bin is
      entry PB when count > 0 is
      begin
         count := count - 1;
      end PB;

      procedure VB is
      begin
         count := count + 1;
      end VB;
   end Semafor_Bin;

begin
   null;
end Semafor_Bin;
```

```
-- Wersja synchronizacji oparta na typach chronionych
--
package Bufor is

   subtype T is Float; -- deklaracja typu danych bufora
   Rozmiar_max: constant Integer := 100;
   subtype Rozmiar is Integer range 1..Rozmiar_max;
   type Buff_Arr is array (Integer range <>) of T;

   protected type Bufor_Obj(N: Rozmiar := 10) is
      entry Pobierz(X: out T);
      entry Wstaw(X: in T);
   private
      Buf: Buff_Arr(0..N);
      Wej: Integer := 0;
      Wyj: Integer := 0;
      Licznik: Integer := 0;
   end Bufor_Obj;

   type Bufor_Obj_Acc is access all Bufor_Obj;

   buf_acc : Bufor_Obj_Acc;

end Bufor;
```

```
package body Bufor is

   protected body Bufor_Obj is
      entry Pobierz(X: out T) when Licznik > 0 is
      begin
         X := Buf(Wyj);
         Wyj := (Wyj + 1) mod N;
         Licznik := Licznik - 1;
      end Pobierz;

      entry Wstaw(X: in T) when Licznik < N is
      begin
         Buf(Wej) := X;
         Wej := (Wej + 1) mod N;
         Licznik := Licznik + 1;
      end Wstaw;
   end Bufor_Obj;

begin
   null;
end Bufor;
```

```
with Bufor, Semafor_Bin;
use Bufor, Semafor_Bin;

package Zadania is
   task type Producent(nr : Integer) is
      entry Start;
      entry Stop;
   end Producent;

   task type Konsument(nr : Integer) is
      entry Start;
      entry Stop;
   end Konsument;

   sem_bin_acc : Semafor_Bin_Acc;

   licznik_prod : Integer;
   licznik_kons : Integer;
end Zadania;
```

```
with Ada.Numerics.Float_Random, Ada.Calendar, Ada.Real_Time,Ada.Text_IO,
     Ada.Integer_Text_IO, Ada.Float_Text_IO, Bufor, Semafor_Bin;
use Ada.Numerics.Float_Random, Ada.Calendar, Ada.Real_Time,Ada.Text_IO,
    Ada.Integer_Text_IO, Ada.Float_Text_IO, Bufor, Semafor_Bin;

package body Zadania is

   gen: Generator;
   first: Boolean := true;

   procedure Produkuj(f: out T) is
      sc: Seconds_Count;
      ts: Time_Span;
   begin
      if (first) then
         first := false;
         Split (Clock, sc, ts);
         Reset(gen, Integer(sc) + 1000);
      end if;
      f := Random(gen) * 10.0;
      delay(Duration(0.1 * f));
   end;

   procedure Konsumuj(f: in T) is
      sc: Seconds_Count;
      ts: Time_Span;
   begin
      if (first) then
         first := false;
         Split (Clock, sc, ts);
         Reset(gen, Integer(sc) + 1000);
      end if;
      delay(Duration(0.05 * f));
   end;

   task body Producent is
      f : float;
   begin
      accept Start  do
         sem_bin_acc.all.PB;
         Put("[P-");
         Put(nr);
         Put("] :: Start - Produkcja");
         New_line;
         sem_bin_acc.all.VB;
      end Start;
      for i in Integer range 1..licznik_prod loop
         sem_bin_acc.all.PB;
         Put("[P-");
         Put(nr);
         Put("] :: Poczatek - Produkcja nr: ");
         Put(i);
         New_line;
         sem_bin_acc.all.VB;
         Produkuj(f);
         buf_acc.all.Wstaw(f);
         sem_bin_acc.all.PB;
         Put("[P-");
         Put(nr);
         Put("] :: Koniec - Produkcja nr: ");
         Put(i);
         Put(" - val= ");
         Put(f);
```

```
         New_line;
         sem_bin_acc.all.VB;
      end loop;
      accept Stop  do
         sem_bin_acc.all.PB;
         Put("[P-");
         Put(nr);
         Put("] :: !! Produkcja zakonczona");
         New_line;
         sem_bin_acc.all.VB;
      end Stop;
   end;

   task body Konsument is
      f : float;
   begin
      accept Start  do
         sem_bin_acc.all.PB;
         Put("[K-");
         Put(nr);
         Put("] :: Start - Knsumpcja");
         New_line;
         sem_bin_acc.all.VB;
      end Start;
      for i in Integer range 1..licznik_kons loop
         sem_bin_acc.all.PB;
         Put("[K-");
         Put(nr);
         Put("] :: Poczatek - Konsumpcja nr: ");
         Put(i);
         New_line;
         sem_bin_acc.all.VB;
         buf_acc.all.Pobierz(f);
         Konsumuj(f);
         sem_bin_acc.all.PB;
         Put("[K-");
         Put(nr);
         Put("] :: Koniec - Konsumpcja nr: ");
         Put(i);
         Put(" - val= ");
         Put(f);
         New_line;
         sem_bin_acc.all.VB;
      end loop;
      accept Stop  do
         sem_bin_acc.all.PB;
         Put("[K-");
         Put(nr);
         Put("] :: !! Konsumpcja zakonczona");
         New_line;
         sem_bin_acc.all.VB;
      end Stop;
   end;

begin
   null;
end Zadania;
```

```
with Ada.Command_Line,Ada.Text_IO, Ada.Integer_Text_IO, Ada.Float_Text_IO,
     Ada.Exceptions, Zadania, ada.Unchecked_Deallocation, Bufor, Semafor_Bin;
use Ada.Command_Line, Ada.Text_IO, Ada.Integer_Text_IO, Ada.Float_Text_IO,
    Ada.Exceptions, Zadania, Bufor, Semafor_Bin;

-- Wersja synchronizacji oparta na typach chronionych
--
procedure Main is

procedure Free is new Ada.Unchecked_Deallocation(Bufor_Obj, Bufor_Obj_Acc);
procedure Free is new Ada.Unchecked_Deallocation
                       (Semafor_Bin.Semafor_Bin, Semafor_Bin.Semafor_Bin_Acc);

   prod1 : Producent(0);
   prod2 : Producent(1);
   kons1 : Konsument(0);
   kons2 : Konsument(1);
   kons3 : Konsument(2);
begin
   licznik_prod := 60;
   licznik_kons := 40;
   buf_acc := new Bufor_Obj(5);
   sem_bin_acc := new Semafor_Bin.Semafor_Bin(1);
   prod1.Start;
   prod2.Start;
   kons1.Start;
   kons2.Start;
   kons3.Start;
   prod1.Stop;
   prod2.Stop;
   kons1.Stop;
   kons2.Stop;
   kons3.Stop;
   Free(buf_acc);
   Free(sem_bin_acc);
   Put("  Koniec programu");
   New_line;
end Main;
```

```
-- Wersja synchronizacji oparta na spotkaniach
--
package Bufor is

   subtype T is Float; -- deklaracja typu danych bufora
   Rozmiar_max: constant Integer := 100;
   subtype Rozmiar is Integer range 1..Rozmiar_max;
   type Buff_Arr is array (Integer range <>) of T;

   task type Bufor_Obj(N: Integer) is
      entry wstaw(x: in T);
      entry pobierz(x: out T);
      entry start;
      entry stop;
   end Bufor_Obj;

   type Bufor_Obj_Acc is access all Bufor_Obj;

   buf_acc : Bufor_Obj_Acc;

end Bufor;
```

```
with Ada.Text_IO, Semafor_Bin, Zadania;
use Ada.Text_IO, Semafor_Bin, Zadania;

package body Bufor is

   task body Bufor_Obj is
      tab: Buff_Arr(0 .. N - 1);
      we, wy: Integer := 0;
      licznik: Integer := 0;
      koniec : Boolean := false;
   begin
      accept Start  do
         sem_bin_acc.all.PB;
         Put("START task - Bufor");
         New_line;
         sem_bin_acc.all.VB;
      end Start;
      loop
         exit when koniec;
         select
            when licznik < N =>
               accept wstaw(x: in T) do
                  tab(we) := x;
               end wstaw;
               licznik := licznik + 1;
               we := (we + 1) mod N;
         or
            when licznik > 0 =>
               accept pobierz(x: out T) do
                  x := tab(wy);
               end pobierz;
               licznik := licznik - 1;
               wy := (wy + 1) mod N;
         or
            accept Stop do
               sem_bin_acc.all.PB;
               Put("STOP task - Bufor");
               New_line;
               sem_bin_acc.all.VB;
               koniec := true;
            end Stop;
         end select;
      end loop;
   end Bufor_Obj;
begin
   null;
end Bufor;
```

```
with Ada.Command_Line,Ada.Text_IO, Ada.Integer_Text_IO, Ada.Float_Text_IO,
     Ada.Exceptions, Zadania, ada.Unchecked_Deallocation, Bufor, Semafor_Bin;
use Ada.Command_Line, Ada.Text_IO, Ada.Integer_Text_IO, Ada.Float_Text_IO,
    Ada.Exceptions, Zadania, Bufor, Semafor_Bin;

-- Wersja synchronizacji oparta na spotkaniach
--
procedure Main is

procedure Free is new Ada.Unchecked_Deallocation(Bufor_Obj, Bufor_Obj_Acc);
procedure Free is new Ada.Unchecked_Deallocation
                       (Semafor_Bin.Semafor_Bin, Semafor_Bin.Semafor_Bin_Acc);

   prod1 : Producent(1);
   prod2 : Producent(2);
   prod3 : Producent(3);
   kons1 : Konsument(1);
   kons2 : Konsument(2);
begin
   licznik_prod := 20;
   licznik_kons := 30;
   buf_acc := new Bufor_Obj(5);
   sem_bin_acc := new Semafor_Bin.Semafor_Bin(1);
   prod1.Start;
   prod2.Start;
   prod3.Start;
   kons1.Start;
   kons2.Start;
   prod1.Stop;
   prod2.Stop;
   prod3.stop;
   kons1.Stop;
   kons2.Stop;
   buf_acc.Stop;
   Free(buf_acc);
   Free(sem_bin_acc);
   Put("  Koniec programu");
   New_line;
end Main;
```