# Oinarrizko Programazioa (2012/06/19)

*(Azterketa honek notaren %70 balio du)*

1. **(2,5 puntu)** [0,100] tarteko zenbaki arruntezko multzo bat balio boolearrez osatutako taula baten bidez errepresenta dezakegu. Horrela, N zenbakia M multzoan baldin badago taularen N-garren osagaia `True` izango da, bestela bere balioa `False` izango da. **`Multzo`** datu motaren definizioa hau da:

```
Max : constant Positive := 100;
type Multzo is array (0..Max) of boolean;
```

Adibidez, {0, 3, 4, ..., 99} multzoa honela errepresentatuko da:

| 0 | 1 | 2 | 3 | 4 | ... | 99 | 100 |
|---|---|---|---|---|---|---|---|
| True | False | False | True | True | ... | True | False |

Ondoko azpiprogramak espezifikatu eta inplementa itzazu:

- Horrelako multzo bat eta [0,100] tarteko zenbaki arrunt bat emanda, ea zenbakia multzoko elementua den aztertzen duen **`Multzokoa`** azpiprograma.
- Multzo bat emanda ea multzo hutsa den aztertzen duen **`Hutsa_Da`** azpiprograma.
- Bi multzo emanda bien arteko ebakidura itzuliko duen **`Ebakidura`** azpiprograma.

```
function  Multzokoa (M: Multzo; N: Integer) return Boolean is
   --Aurre: 0<=N<=100
   --Post: Emaitza = True N zenbakia M multzoan badago,
   --      Emaitza= False bestela.
   Emaitza: Boolean;
begin
   Emaitza := M(N);
   return Emaitza;
end Multzokoa;

function Hutsa_Da (M: Multzo) return Boolean is
   --Aurre:
   --Post: Emaitza = True M multzoa multzo hutsa bada,
   --      Emaitza= False bestela.
   Emaitza: Boolean;
   I: Integer;
begin
   Emaitza:= True;
   I:=0;
   while I<=100 and Emaitza=True loop
      if M(I) then
         Emaitza:= False;
      end if;
   end loop;
   return Emaitza;
end Hutsa_Da;

function Ebakidura (M1, M2: in Multzo) return Multzo is
   --Aurre:
   --Post: emaitza = M1 eta M2 multzoen bilduraebakidura
   M:Multzo;
begin
   for I in M'First..M'Last loop
      M(I) := M1(I) and M2(I);
   end loop;
   return M;
end Ebakidura;
```

2. **(3,5 puntu)** `Kirolari` datu motaren eragiketa hauek emanda:

```
function Izena (K: Kirolari) return string;
-- Post: Emaitza = K kirolariaren izena.

procedure NAN_Portzentajea (K: Kirolari; Kodea: out integer; P: out natural);
-- Post: Kodea = K kirolariaren NAN zenbakia;
--       P =  K kirolariaren balorazioaren portzentajea
```

Eta `Kirolari_Lista` datu motaren definizio hau emanda:

```
Max: constant positive:= 50;
subtype Osoko0_Max is Integer range 0 .. Max;
type Taula is array (1..Max) of Kirolari;
type Kirolari_Lista is record
    Info: Taula;
    Zenbat: Osoko0_Max;
end record;
```

Espezifikatu, Ada lengoaiaz inplementatu azpiprograma bat hau egingo duena: Kirolari-lista bat emanda, izenaz alfabetikoki ordenatuta egongo den `Kirolari_Lista` motako beste lista bat sortuko du balorazioan gutxienez %70eko portzentajea duten kirolariekin.

```
procedure  Balorazio_handikoak_70 (L: Kirolari_Lista; L70: out Kirolari_Lista) is
  --Aurre:
  --Post: L70 listako elementuak L listako elementuak dira balorazioan 70
          edo gehiago dutenak.
  --      L70 listako eelementuak  ordenatuta daude izenaren arabera alfabetikoki
  Emaitza: Boolean;
  Kodea : Integer;
  Balorazioa: Natural;
begin
  L70.Zenbat := 0;
  for I in 1..L.Zenbat loop
   NAN_Portzentajea (L.Info(I), Kodea, Balorazioa);
   if Balorazioa >=70 then
     Txertatu_Ordenatuan(L70, L.Info(I));
   end if;
  end loop;
end Balorazio_handikoak_70;
```

```
procedure Txertatu_Ordenatuan
          (KL:       in out Kirolari_Lista;
           Kirolaria: in     Kirolari) is
 --AURRE: KL Listako kirolariak ordenatuta daude izenaren arabera alfabetikoki
 --POST:  Kirolaria  KL listan dago.
 --       KL ordenatuta dago.
   Posizioa: Integer ;
 begin
   Bilatu_Posizioa_Ordenatuan (KL, Kirolaria, Posizioa);
   Desplazatu_Eskuinaldera (KL, Posizioa, KL.Zenbat);
   KL.Info(Posizioa) := Kirolaria;
   KL.Zenbat := KL.Zenbat +1;
end Txertatu_Ordenatuan;
```

```
procedure Bilatu_Posizioa_Ordenatuan
            (L        : in     Kirolari_Lista;
             Kirolaria: in     Kirolari;
             Pos      :    out Integer) is
  --Post: K kirolaria L lista ordenatuan
  --      Pos posizioan sartu beharko litzateke
  --      L lista ordenatuta jarraitzeko
  I : Integer;
  Aurkitua: Boolean;
  Ize, Ize1 : Izen;
  Kodea, Balorazioa: Integer;
begin
  I := 1;
  Aurkitua := False;
  NAN_Portzentajea (L.Info(I), Kodea, Balorazioa);
  Ize := Izena( Kirolaria);
  while I<= L.Zenbat and not Aurkitua loop
   Ize1 := Izena( L.Info(I));
     if Ize < Ize1 then
        Aurkitua := True ;
     else
        I := I + 1;
     end if;
  end loop;
  --(Aurkitua =True
  --   eta listako Igarrenaren izena kontaktu berriarena
  --         baino handiago den lehengoa da lista ordenatuan)
  --edo
  --(Aurkitua = False
  --      eta listako izen guztiak kontaktuarena baino txikiagoak dira)
  if Aurkitua then
     Pos := I;
  else
     Pos := L.Zenbat + 1;
  end if;
end Bilatu_Posizioa_Ordenatuan;



procedure Desplazatu_Eskuinaldera (L     : in out Kirolari_Lista;
                                   I1,I2 : in     integer) is
  --Aurre:
  --Post: L listako [I1..I2] indizetarteko osagaiak posizio bat
  --      desplazatu dira eskuinaldera [I1+1..I2+1] tarteko osagaietara.
begin
  for I in reverse I1..I2 loop
     L.Info(I+1) := L.Info(I);
  end loop;
end Desplazatu_Eskuinaldera;
```

3. **(4 puntu)** 30 herritako biztanle-dentsitateei buruzko datuak errepresentatzeko `DentsitateT` datu-mota hau definitu dugu:

```
type DentsitateT is array (1..30) of Float;
```

`Gertuena_Batezbestekotik` azpiprograma espezifikatu, inplementatu eta proba-kasuak definitu itzazu. Azpiprograma horrek `DentsitateT` motako taula bat emanda, taulako dentsitate guztien batezbestekotik gertuen dagoen dentsitatearen herria (bere posizioa taulan) itzuliko du. Herriren baten dentsitatea eta batezbestekoa berdinak badira herri horretxen posizioa da nahi duguna, eta horrelako bat baino gehiago balego horietako lehenengoaren posizioa lortzea aski zaigu. Oharrak:

- Gertutasuna distantziaren balio absolutuan neurtzen denez, gertuenaren balioa batezbestekoa baino handiagoa edo txikiagoa izan daiteke.
- Bi zenbaki erreal emanda berdinak direla esango dugu beren arteko kenduraren balio absolutua 0.1 baino txikiagoa bada.

```
function Gertuena_Batezbestekotik (D: in DentsitateT) return Natural is
-- Aurre:
-- Post: Gertuenekoa (Natural);
-- Gertuenekoa = dentsitateen batezbestekotik gertuen dagoen herriaren posizioa
--       (bere posizioa taulan)
--       Herriren baten dentsitatea eta batezbestekoa berdinak badira
--         herri horretxen posizioa da nahi duguna,
--         eta horrelako bat baino gehiago balego
--           horietako lehenengoaren posizioa lortzea aski zaigu.
    Gertuenekoa : Natural;
    I : Natural;
    Aurkitua : Boolean;
    Batez, Distantzia_Minimoa : Float;
begin
    Batez := Dentsitateen_Batezbestekoa (D);
    Gertuenekoa := D'First;
    Distantzia_Minimoa := abs(Batez-D(1));
    Aurkitua := False;
    I := D'First;
    while I <= D'Last and not Aurkitua loop
       if Berdinak(D(I),Batez) then
     Aurkitua := True;
          Gertuenekoa := I;
       elsif  abs(Batez-D(I)) <Distantzia_Minimoa then
     Distantzia_Minimoa := abs(Batez-D(I));
     Gertuenekoa := I;
     I := I + 1;
       end if;
    end loop;
    return Gertuenekoa;
end Gertuena_Batezbestekotik;
```

```
function Dentsitateen_Batezbestekoa (D : DentsitateT) return Float is
 -- Aurre:
 -- Post: Emaitza; Emaitza = Herri guztien dentsitateen batezbestekoa
 Batura, Emaitza : Float;
 begin
    Batura := 0.0;
    for I in D'First .. D'Last loop
       Batura := Batura + D(I);
    end loop;
    Emaitza := Batura / Float(D'Length);
    return  Emaitza;
 end Dentsitateen_Batezbestekoa;


 function Berdinak (X,Y: Float) return Boolean is
-- Aurre:
-- Post: Emaitza = True X eta Y zenbakien arteko distantzia 0.1 baino txikagoa bada
--       Emaitza = False bestela
    Emaitza: Boolean;
 begin
    if  abs(X-Y) < 0.1 then
  Emaitza := True;
    else
  Emaitza := False;
    end if;
    return Emaitza;
 end Berdinak;
```

**Proba-kasuak**

1. Batezbestekoaren berdina den osagai bat dago. Listako lehena da.
2. Batezbestekoaren berdina den osagai bat dago. Listako azkena da.
3. Batezbestekoaren berdina den osagai bat dago. Ez da lehena edo azkena. Dentsitateak
4. Batezbestekoaren berdina den osagai bat dago. Ez da lehena edo azkena. “Berdina” da baina distantzia 0,1 da.
5. Ez dago batezbestekoaren berdina den osagairik. Gertuenekoa listako lehena da.
6. Ez dago batezbestekoaren berdina den osagairik. Gertuenekoa listako azkena da.
7. Ez dago batezbestekoaren berdina den osagairik. Gertuenekoa ez da lehena edo azkena.