PB– 6. Laborategiko Informea

Izena: Eneko, Gontzal eta Markel **Data**:2014-11-04

Ikasitako kontzeptuak eta lortutako gaitasunak

Behin laborategia bukatu dudala, eskatu dizkidaten ariketetan erabiltzen diren kontzeptuak ondo ulertu ditudala kontsideratzen dut, eta beraz ariketak errepikatzeko gai izango nintzatekela esan dezaket. Erantzun hurrengo galdera-sorta esanez ea gai zaren (ala ez) hor galdetzen dena egiteko.

Eneko	**BAI**	**EZ**
1 Badakit taula bateko elementu bat topatzen modu eragikorrean	Bai	
2 Badakit array bati dagokion datu mota berri bat sortzen	Bai	
3 a) eta b) bertsioen arteko desberdintasuna ondo ulertzen dut eta bakoitza noiz erabili behar den ere ondo ulertzen dut a) type T_Osoak is array (1..100) of Integer; b) type T_Osoak is array (1..100) of Integer; type Info_Osoak is record Kopurua: integer; Zenbakiak: T_Osoak; end record; Ezezko erantzunen bat eman baduzu mesedez jarraian azaldu zergaitia, eta ze nolako neurria hartuko duzun sortu zaizun arazoa konpontzeko.	Bai	

Gontzal	BAI	EZ
1 Badakit taula bateko elementu bat topatzen modu eragikorrean	Bai	
2 Badakit array bati dagokion datu mota berri bat sortzen	Bai	
3 a) eta b) bertsioen arteko desberdintasuna ondo ulertzen dut eta bakoitza noiz erabili behar den ere ondo ulertzen dut a) type T_Osoak is array (1..100) of Integer; b) type T_Osoak is array (1..100) of Integer; type Info_Osoak is record Kopurua: integer; Zenbakiak: T_Osoak; end record; Ezezko erantzunen bat eman baduzu mesedez jarraian azaldu zergaitia, eta ze nolako neurria hartuko duzun sortu zaizun arazoa konpontzeko.	Bai	

Markel	BAI	EZ
1 Badakit taula bateko elementu bat topatzen modu eragikorrean	Bai	
2 Badakit array bati dagokion datu mota berri bat sortzen	Bai	
3 a) eta b) bertsioen arteko desberdintasuna ondo ulertzen dut eta bakoitza noiz erabili behar den ere ondo ulertzen dut a) type T_Osoak is array (1..100) of Integer; b) type T_Osoak is array (1..100) of Integer; type Info_Osoak is record Kopurua: integer; Zenbakiak: T_Osoak; end record; Ezezko erantzunen bat eman baduzu mesedez jarraian azaldu zergaitia, eta ze nolako neurria hartuko duzun sortu zaizun arazoa konpontzeko.	Bai	

6. Laborategiko soluzioa

1.Zenbakia bilatu osoko-bektore desordenatuan: N elementu dituen osoko-bektore desordenatu batean zenbaki bat bilatzeko algoritmoa espezifikatu eta egin. Irteera, zenbakia zerrendan dagoen ala ez izango da. Horretarako erabili proba_bektore_desordenatuan_bai_ez.adb eta bektore_desordenatuan_bai_ez.adb.

bektore_desordenatuan_bai_ez.adb

```
with bektore_eta_matrizeak;
use bektore_eta_matrizeak;

function bektore_desordenatuan_bai_ez (elem: in integer; taula: in
osokoenBektorea) return boolean is

    rdo: boolean;
    i: integer;

begin

    i := 1;
    rdo:= False;

    loop exit when ( (rdo = True) or (i > 10) );

        if (taula(i) = elem) then
            rdo:= True;
        end if;

        i := i + 1;
    end loop;

    return (rdo);
end bektore_desordenatuan_bai_ez;
```

proba_bektore_desordenatuan_bai_ez.adb

```
with Ada.Text_Io;
use Ada.Text_Io;

with bektore_eta_matrizeak;
use bektore_eta_matrizeak;
with bektore_desordenatuan_bai_ez;

procedure proba_bektore_desordenatuan_bai_ez is

    package Boolean_E_S is new Enumeration_Io(Boolean);
    use Boolean_E_S;

    Bektore1: osokoenBektorea(1..10);
    rdo:boolean;

begin

   Bektore1 := (1, 13, 55, 27, 99, 111, 133, 150, 17, 6);
   Put_line("Proba1: bilatu nahi dugun balioa erdian balego");
```

```
   Put_line(" bektore_ordenatuan_bai_ez(111, (1, 13, 55, 27, 99, 111, 133, 150,
17, 6))");
   Put_Line(" TRUE bueltatu beharko luke eta emaitza ");
   rdo:=bektore_desordenatuan_bai_ez(111,Bektore1);
   if rdo=true then
    Put("TRUE");
   else
    Put("FALSE");
   end if;
   New_Line(3);
   Put_Line("Return sakatu jarraitzeko");
   Skip_Line;
   New_Line(3);
   ----------------
   Bektore1 := (16, 33, 567, 73, 9, 111, 153, 15, 197, 1);
   Put_line("Proba2: bilatu nahi dugun balioa bukaeran balego");
   Put_line(" bektore_desordenatuan_bai_ez(1, (16, 33, 567, 73, 9, 111, 153, 15,
197, 1))");
   Put_Line(" TRUE  bueltatu beharko luke eta emaitza ");
   rdo:=bektore_desordenatuan_bai_ez(1,Bektore1);
    if rdo=true then
        Put("TRUE");
    else
        Put("FALSE");
    end if;
    New_Line(3);
    Put_Line("Return sakatu jarraitzeko");
    Skip_Line;
    New_Line(3);
    ----------------
   Bektore1 := (19, 3, 556, 72, 91, 11, 1, 15, 817, 199);
   Put_line("Proba3: bilatu nahi dugun balioa ez balego ...");
   Put_line(" bektore_ordenatuan_bai_ez(45, (19, 3, 556, 72, 91, 11, 1, 15, 817,
199))");
   Put_Line(" FALSE bueltatu beharko luke eta emaitza ");
   rdo:=bektore_desordenatuan_bai_ez(45, Bektore1);
   if rdo=true then
    Put("TRUE");
   else
    Put("FALSE");
   end if;
   New_Line(3);
   Put_Line("Return sakatu jarraitzeko");
   Skip_Line;
   New_Line(3);


end proba_bektore_desordenatuan_bai_ez;
```

2. Zenbakia bilatu osoko-bektore ordenatuan: N elementu dituen osoko-bektore ordenatu batean (handienetik txikira) zenbaki bat bilatzeko algoritmoa espezifikatu eta egin. Irteera, zenbakia bektorean dagoen ala ez izango da, TRUE baiezkoan, FALSE bestela. Horretarako erabili proba_bektore_ordenatuan_bai_ez.adb eta bektore_ordenatuan_bai_ez.adb.

bektore_ordenatuan_bai_ez.adb

```
with bektore_eta_matrizeak;
use bektore_eta_matrizeak;

function bektore_ordenatuan_bai_ez (elem: in integer; taula: in osokoenBektorea)
return boolean is
    rdo: boolean;
    i: integer;

begin

    rdo:= False;
    i:=1;

   loop exit when ( (rdo = True) or (taula(i) > elem) );
        if(taula(i) = elem) then
            rdo:=True;
        end if;
        i := i + 1;
   end loop;

   return (rdo);

end bektore_ordenatuan_bai_ez;
```

proba_bektore_ordenatuan_bai_ez.adb

```
with Ada.Text_Io;
use Ada.Text_Io;

with bektore_eta_matrizeak;
use bektore_eta_matrizeak;
with bektore_ordenatuan_bai_ez;

procedure proba_bektore_ordenatuan_bai_ez is
    Bektore1: osokoenBektorea(1..10);
    rdo: boolean;

begin

   Bektore1 := (30, 31, 255, 270, 290, 281, 630, 700, 900, 960);
   Put_line("Proba1: bilatu nahi dugun balioa erdian balego");
   Put_line(" bektore_ordenatuan_bai_ez(290, (30, 31, 255, 270, 290, 281, 630,
700, 900, 960))");
   Put_Line(" TRUE bueltatu beharko luke eta emaitza ");
   rdo:=bektore_ordenatuan_bai_ez(290, Bektore1);
    if rdo=true then
        Put("TRUE");
    else
        Put("FALSE");
    end if;
   New_Line(3);
   Put_Line("Return sakatu jarraitzeko");
```

```
   Skip_Line;
   New_Line(3);
    -------------------
   Bektore1 := (30, 31, 255, 270, 290, 281, 630, 700, 900, 960);
   Put_line("Proba2: bilatu nahi dugun balioa bukaeran balego");
   Put_line(" bektore_ordenatuan_bai_ez(960, (30, 31, 255, 270, 290, 281, 630,
700, 900, 960))");
   Put_Line(" TRUE  bueltatu beharko luke eta emaitza ");
    ---Deia falta da!!! bektore_ordenatuan_bai_ez(Parametroak falta dira);
    rdo:=bektore_ordenatuan_bai_ez(960, Bektore1);
    if rdo=true then
        Put("TRUE");
    else
        Put("FALSE");
    end if;

   New_Line(3);
   Put_Line("Return sakatu jarraitzeko");
   Skip_Line;
   New_Line(3);
    -------------------
   Bektore1 := (30, 31, 255, 270, 290, 281, 630, 700, 900, 960);
   Put_line("Proba3: bilatu nahi dugun balioa ez balego ...");
   Put_line(" bektore_ordenatuan_bai_ez(45, (30, 31, 255, 270, 290, 281, 630,
700, 900, 960))");
   Put_Line(" FALSE bueltatu beharko luke eta emaitza ");
  ---Deia falta da!!! bektore_ordenatuan_bai_ez(Parametroak falta dira);
    rdo:=bektore_ordenatuan_bai_ez(45, Bektore1);
    if rdo=true then
        Put("TRUE");
    else
        Put("FALSE");
    end if;

   New_Line(3);
   Put_Line("Return sakatu jarraitzeko");
   Skip_Line;
   New_Line(3);


end proba_bektore_ordenatuan_bai_ez;
```

3.Zenbakia bilatu bektore batean eta bere posizioa itzuli: N elementu dituen osoko-bektore bat eta zenbaki bat emanda, zenbaki hori zerrendan balego, zein posiziotan dagoen itzuliko duen algoritmoa espezifikatu eta egin. Zenbakia hainbat aldiz agertzen bada, nahi duzuen posizioa itzuli baina ez balego, itzuli -1 posizioa. Horretarako erabili posizioan_dago.adb eta proba_posizioan_dago.adb

posizioan_dago.adb

```
with bektore_eta_matrizeak;
use bektore_eta_matrizeak;

function posizioan_dago (elem: integer; bektorea: osokoenBektorea) return
integer is

    rdo:boolean;
    i: integer;

begin

    i:=1;
    rdo:=False;

    loop exit when ((rdo = True) or (i = 10));
            i:=i+1;

            if(elem = bektorea(i)) then
                rdo:=True;
            end if;
    end loop;

    if(rdo = False) then
        i:=-1;
    end if;

    return(i);
end posizioan_dago;
```

proba_posizioan_dago.adb

```
with Ada.Text_Io;
use Ada.Text_Io;

with Ada.Integer_Text_IO;
use Ada.Integer_Text_IO;

with bektore_eta_matrizeak;
use bektore_eta_matrizeak;

with posizioan_dago;

procedure proba_posizioan_dago is
    Bektore1: osokoenBektorea(1..10);
    posizio: integer;

begin

   Bektore1 := (1, 3, 5, 17, 9, 11, 131, 15, 170, 19);
   Put_line("Proba1: bilatu nahi dugun balioa ez balego");
   Put_line(" posizioan_dago(13, (1, 3, 5, 17, 9, 11, 131, 15, 170, 19))");
```

```
    Put_Line(" -1 bueltatu beharko luke eta emaitza ");
    posizio:=posizioan_dago(43, Bektore1);
    Put(posizio);
    New_Line(3);
    Put_Line("Return sakatu jarraitzeko");
    Skip_Line;
    New_Line(3);

    Bektore1 := (1, 3, 5, 17, 9, 11, 131, 15, 170, 19);
    Put_line("Proba1: bilatu nahi dugun balioa ez balego");
    Put_line(" posizioan_dago(13, (1, 3, 5, 17, 9, 11, 131, 15, 170, 19))");
    Put_Line(" 5 bueltatu beharko luke eta emaitza ");
    posizio:=posizioan_dago(9, Bektore1);
    Put(posizio);
    New_Line(3);
    Put_Line("Sakatu return jarraitzeko");
    Skip_Line;
    New_Line(3);

    Bektore1 := (1, 3, 5, 17, 9, 11, 131, 15, 170, 19);
    Put_line("Proba1: bilatu nahi dugun balioa ez balego");
    Put_line(" posizioan_dago(13, (1, 3, 5, 17, 9, 11, 131, 15, 170, 19))");
    Put_Line(" 10 bueltatu beharko luke eta emaitza ");
    posizio:=posizioan_dago(19, Bektore1);
    Put(posizio);
    New_Line(3);
    Put_Line("Sakatu return jarraitzeko");
    Skip_Line;
    New_Line(3);

end proba_posizioan_dago;
```

4. Posizio bat eskuinera mugitu N elementuz osatutako osoko-bektore bat emanda, osagai guztiak posizio bat eskuinera mugitzen dituen algoritmoa espezifikatu eta egin. Azpiprograma modura inplementatu. Erabili eskuinera_mugitu.adb, eta proba_eskuinera_mugitu.adb

eskuinera_mugitu.adb

```
with bektore_eta_matrizeak;
use bektore_eta_matrizeak;


function eskuinera_mugitu (taula: in osokoenBektorea) return osokoenBektorea is

    taula_mugituta: osokoenBektorea(1..10);
    i: integer;

begin

    i := 1;

    loop exit when (i > 10);

        if (i=10) then
            taula_mugituta(i-9) := taula(i);
        else
            taula_mugituta(i+1) := taula(i);
        end if;

        i := i + 1;

    end loop;

    return (taula_mugituta);

 end eskuinera_mugitu;
```

proba_eskuinera_mugitu.adb

```
with Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer_Text_Io;

with bektore_eta_matrizeak;
use bektore_eta_matrizeak;
with eskuinera_mugitu;

procedure proba_eskuinera_mugitu is

    t1: osokoenBektorea(1..10);
    t2: osokoenBektorea(1..10);

begin

    t1 := (1, 3, 5, 7, 9, 11, 13, 15, 17, 1);
    Put_line("Proba: Emaitza: (1, 1, 3, 5, 7, 9, 11, 13, 15, 17)");
    Put_line("izan beharko litzateke");
    put_line("eta emaitza  ");
    t2:=eskuinera_mugitu(t1);
    for i in 1..10 loop
        Put(t2(i));
        put(",");
    end loop;
```

```
        Skip_Line;
        New_Line(3);

        t1 := (1, 1, 1, 1, 1, 1, 3, 1, 1, 1);
        Put_line("Proba: Emaitza: (1, 1, 1, 1, 1, 1, 1, 3, 1, 1)");
        Put_line("izan beharko litzateke");
        put_line("eta emaitza  ");
        t2:=eskuinera_mugitu(t1);
        for i in 1..10 loop
            Put(t2(i));
            put(",");
        end loop;

    end proba_eskuinera_mugitu;
```

5. Eliminatu elementu bat zerrenda desordenatu batean. N elementuz osatutako osoko-bektore bat emanda, non behar bada bektorea ez den guztiz beteta egongo (beraz erregistro baten barruan egonda non erregistroak bektorez gain kopuru eremu bat ere izango duen). Eliminatu lehenengo elementua. Bektorearen elementuak ez daude ordenatuta. Horretarako, erabili eliminatu_lehenengo_elementua.adb, idatzi_lista.adb eta proba_eliminatu_lehenengo_elementua.adb

eliminatu_lehenengo_elementua.adb

```
with bektore_eta_matrizeak,Defs_lab;
use bektore_eta_matrizeak,Defs_lab;

function eliminatu_lehenengo_elementua (taula: in osokoen_zerrenda) return
osokoen_zerrenda is
   -- aurre: erregistro barruko bektoreak behintzat elementu bat izango du
   -- post: lehenengo elementua ezabatuko da

    taula1: osokoen_zerrenda;
begin

    taula1:=taula;

    taula1.zenbakiak(1) := taula1.zenbakiak(taula.kopurua);
    taula1.kopurua := (taula1.kopurua - 1);

    return(taula1);

end eliminatu_lehenengo_elementua;
```

proba_eliminatu_lehenengo_elementua.adb

```
with Ada.Text_Io;
use Ada.Text_Io;

with defs_lab; use defs_lab;
with idatzi_Lista, eliminatu_lehenengo_elementua;

procedure proba_eliminatu_lehenengo_elementua is

Lista1: osokoen_zerrenda;

begin
    Lista1.zenbakiak(1) := 1;
    Lista1.kopurua := 1;
    Put_line("1 kasua: elementu bakarreko zerrenda: (1)");
    Put_line(" Hasierako zerrendaren elementuak hauek dira: ");
    idatzi_Lista(Lista1);
    new_line;
    put_line(" Utzik gelditu beharko litzateke behin lehenengo elementua
eliminatzen dela eta gelditzen da: ");
    Lista1 := eliminatu_lehenengo_elementua(Lista1);
    idatzi_Lista(Lista1);
    New_Line(3);
    Put_Line("Return sakatu jarraitzeko");
    Skip_Line;
    New_Line(3);

    --- hainbat elementuko zerrenda
    Lista1.zenbakiak(1) := 3; Lista1.zenbakiak(2) := 1; Lista1.zenbakiak(3) :=
4;
    Lista1.zenbakiak(4) := 8; Lista1.zenbakiak(5) := 6;
```

```
    Lista1.kopurua := 5;
    Put_line("2. kasua: zerrenda ez ordenatua non elementuak ez dauden
ordenatuta: (3 1 4 8 6)");
    Put_line(" zerrenda originala hurrengoa da: ");
    idatzi_Lista(Lista1);
    put_line(" Eta emaitza lehenengo elementua kenduta izan beharko litzateke 1
4 8 6 eta :");
    Lista1 := eliminatu_lehenengo_elementua(Lista1);
    idatzi_Lista(Lista1);
    put(" da");
    New_Line(3);
    Put_Line("Return sakatu jarraitzeko");
    Skip_Line;
    New_Line(3);

    Lista1.zenbakiak(1) := 11; Lista1.zenbakiak(2) := 1; Lista1.zenbakiak(3) :=
5;
    Lista1.zenbakiak(4) := 5; Lista1.zenbakiak(5) := 4; Lista1.zenbakiak(6) :=
96;
    Lista1.zenbakiak(7) := 1; Lista1.zenbakiak(8) := 11; Lista1.zenbakiak(9) :=
12;
    Lista1.kopurua := 9;
    Put_line("2. kasua: zerrenda ez ordenatua non elementuak ez dauden
ordenatuta: (11 1 5 5 4 96 1 11 12)");
    Put_line(" zerrenda originala hurrengoa da: ");
    idatzi_Lista(Lista1);
    put_line(" Eta emaitza lehenengo elementua kenduta izan beharko litzateke 1
5 5 4 96 1 11 12 eta :");
    Lista1 := eliminatu_lehenengo_elementua(Lista1);
    idatzi_Lista(Lista1);
    put(" da");
    New_Line(3);
    Put_Line("Return sakatu jarraitzeko");
    Skip_Line;
    New_Line(3);

end proba_eliminatu_lehenengo_elementua;
```

6. Eliminatu elementu bat zerrenda ordenatu batean. N elementuz osatutako osoko-bektore bat emanda, non behar bada bektorea ez den guztiz beteta egongo (beraz erregistro baten barruan egonda non erregistroak bektorez gain kopuru eremu bat ere izango duen). Eliminatu lehenengo elementua. Bektorearen elementuak ordenatuta daude txikitik handira. Horretarako, erabili eliminatu_lehenengo_elementua_ordenatuta.adb, idatzi_lista.adb eta proba_eliminatu_lehenengo_elementua_ordenatuta.adb

eliminatu_lehenengo_elementua_ordenatuta.adb

```
with bektore_eta_matrizeak,Defs_lab;
use bektore_eta_matrizeak,Defs_lab;

function eliminatu_lehenengo_elementua_ordenatuta (zerrenda: in
Osokoen_zerrenda) return Osokoen_zerrenda is
    ezabatuta: Osokoen_zerrenda;
    kont: integer;
   -- aurre: erregistro barruko bektoreak behintzat elementu bat izango du,
   -- eta elementuak txikitik handira ordenatuta egongo dira
   -- post: lehenengo elementua ezabatuko da
begin
    kont := 0;
    ezabatuta.kopurua := (zerrenda.kopurua - 1);

    loop exit when (kont = (zerrenda.kopurua - 1)) or (zerrenda.kopurua = 1);
        kont := kont + 1;
        ezabatuta.zenbakiak(kont) := zerrenda.zenbakiak(kont + 1);
    end loop;
    return ezabatuta;
end eliminatu_lehenengo_elementua_ordenatuta;
```

proba_eliminatu_lehenengo_elementua_ordenatuta.adb

```
with Ada.Text_Io;
use Ada.Text_Io;

with defs_lab; use defs_lab;
with idatzi_Lista, eliminatu_lehenengo_elementua_ordenatuta;

procedure proba_eliminatu_lehenengo_elementua_ordenatuta is


Lista1: osokoen_zerrenda;

begin
   ---elementu bakarra
   Lista1.zenbakiak(1) := 1;
   Lista1.kopurua := 1;
   Put_line("1 kasua: elementu bakarreko zerrenda: (1)");
   Put_line(" Hasierako zerrendaren elementuak hauek dira: ");
   idatzi_Lista(Lista1);
   new_line;
   put_line(" Utzik gelditu beharko litzateke behin lehenengo elementua
eliminatzen dela eta gelditzen da: ");
   Lista1 := eliminatu_lehenengo_elementua_ordenatuta(Lista1);
   idatzi_Lista(Lista1);
   New_Line(3);
   Put_Line("Return sakatu jarraitzeko");
   Skip_Line;
   New_Line(3);
```

```
   --- hainbat elementuko zerrenda
   Lista1.zenbakiak(1) := 1; Lista1.zenbakiak(2) := 3; Lista1.zenbakiak(3) := 4;
   Lista1.zenbakiak(4) := 6; Lista1.zenbakiak(5) := 8;
   Lista1.kopurua := 5;
   Put_line("2. kasua: zerrenda ez ordenatua non elementuak ez dauden ordenatuta:
(1 3 4 6 8)");
   Put_line(" zerrenda originala hurrengoa da: ");
   idatzi_Lista(Lista1);
   put_line(" Eta emaitza lehenengo elementua kenduta izan beharko litzateke 3
4 6 8 eta :");
   Lista1 := eliminatu_lehenengo_elementua_ordenatuta(Lista1);
   idatzi_Lista(Lista1);
   put(" da");
   New_Line(3);
   Put_Line("Return sakatu jarraitzeko");
   Skip_Line;
   New_Line(3);

end proba_eliminatu_lehenengo_elementua_ordenatuta;
```

7. Txertatu zenbaki bat zerrendako Ngarren posizioan. N-1 elementu dituen zerrenda batean, non elementu bat txertatzeko tokia dagoen, txertatu ezazu zenbakia pos posizioan eta gainontzeko zenbakiak desplazatu itzazu eskuinera. Horretarako, erabili itzazu txertatu_zenbakia_pos.adb eta proba_txertatu_zenbakia_pos.adb

txertatu_zenbakia_pos.adb

```
with defs_lab;
use defs_lab;

procedure txertatu_zenbakia_pos (zenb: in integer; lista: in out
osokoen_zerrenda; pos: in integer) is
---sarrera: txeratzeko zenbaki bat, zenbakien zerrenda bat bertan txertatzeko eta
txertatzeko posizioa
--- aurre: 0< pos <= elementu kopurua +1,
--irteera: zerrendak elementu bat gehiago izango du, eta zenb pos posizioan
txeratuko da gainontzeko
    ---elementuak eskuinera desplazatuz

    kont, aux: integer;
begin

    lista.kopurua := lista.kopurua + 1;
    aux := 1;
    kont := pos;

    loop exit when ( (lista.kopurua = kont));
        lista.zenbakiak((lista.kopurua + 1 - aux)) :=
lista.zenbakiak(lista.kopurua - aux);
        kont := kont + 1;
        aux := aux + 1;
    end loop;

    lista.zenbakiak(pos) := zenb;

end txertatu_zenbakia_pos;
```

proba_txertatu_zenbakia_pos.adb

```
with Ada.Text_Io;
use Ada.Text_Io;

with defs_lab,txertatu_zenbakia_pos,idatzi_lista;
use defs_lab;
procedure proba_txertatu_zenbakia_pos is

    Lista1: osokoen_zerrenda;

begin
    Lista1.zenbakiak(1) := 10;
    Lista1.kopurua := 1;
    Put_line("1 kasua: elementu bakarreko zerrenda: (1)");
    Put_line(" Hasierako posizioan 5 bat txertatuko da: ");

    idatzi_Lista(Lista1);
    new_line;
    put_line(" zerrenda 5, 10 zenbakiz osatuta agertuko da: ");
    txertatu_zenbakia_pos(5, Lista1, 1);
    idatzi_Lista(Lista1);
    New_Line(3);
```

```
      Put_Line("Return sakatu jarraitzeko");
      Skip_Line;
      ----------------------------

      Lista1.zenbakiak(1) := 1; Lista1.zenbakiak(2) := 6;
      Lista1.zenbakiak(3) := 15; Lista1.zenbakiak(4) := 7;
      Lista1.kopurua := 4;
      Put_line("2 kasua: lau elementuko zerrenda (1, 6, 15, 7)");
      Put_line(" Hirugarren posizioan 2 bat txertatuko da: ");

      idatzi_Lista(Lista1);
      new_line;
      put_line(" zerrenda 1, 6, 2, 15, 7 zenbakiz osatuta agertuko da: ");
      txertatu_zenbakia_pos(2, Lista1, 3);
      idatzi_Lista(Lista1);
      New_Line(3);
      Put_Line("Return sakatu jarraitzeko");
      Skip_Line;
      ---------------------------------
      Lista1.zenbakiak(1) := 1; Lista1.zenbakiak(2) := 2;
      Lista1.zenbakiak(3) := 3; Lista1.zenbakiak(4) := 4;
      Lista1.zenbakiak(5) := 5; Lista1.zenbakiak(6) := 6;
      Lista1.zenbakiak(7) := 7;
      Lista1.kopurua := 7;
      Put_line("3 kasua: zazpi elementuko zerrenda (1, 2, 3, 4, 5, 6, 7)");
      Put_line(" Laugarren posizioan 4 bat txertatuko da: ");

      idatzi_Lista(Lista1);
      new_line;
      put_line(" zerrenda 1, 2, 3, 4, 4, 5, 6, 7 zenbakiz osatuta agertuko da: ");
      txertatu_zenbakia_pos(4, Lista1, 4);
      idatzi_Lista(Lista1);
      New_Line(3);
      Put_Line("Return sakatu jarraitzeko");
      Skip_Line;


end proba_txertatu_zenbakia_pos;
```

Defs_lab.adb

```
with bektore_eta_matrizeak;
use bektore_eta_matrizeak;

package defs_Lab is

Max_Elem: constant Integer := 10;

type Osokoen_zerrenda is record
     kopurua:integer;
     zenbakiak:osokoenBektorea(1..10);
end record;

end defs_Lab;
```

bektore_eta_matrizeak.adb

```
package bektore_eta_matrizeak is

   type osokoenBektorea is array (Integer range <>) of Integer;


   type errealenBektorea is array (Integer range <>) of Float;


   type boolearrenBektorea is array (Integer range <>) of Boolean;


   type karakterrenBektorea is array (Integer range <>) of Character;



   type osokoenMatrize is array (Integer range <>, Integer range <>) of Integer;


   type errealenMatrize is array (Integer range <>, Integer range <>) of Float;


   type boolearrenMatrize is array (Integer range <>, Integer range <>) of
Boolean;


   type karaktereenMatrize is array (Integer range <>, Integer range <>) of
Character;


end bektore_eta_matrizeak;
```

Idatzi_lista.adb

```
with Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer_Text_Io;

with Defs_Lab;
use Defs_Lab;


procedure idatzi_lista ( L : in  osokoen_zerrenda ) is
   --Pre:
   --Post:  se han escrito en la secuencia de salida los valores de L
   --          desde 1 hasta L.cont

begin
   for I in 1 .. L.kopurua loop
      Put(L.zenbakiak(I), width => 3);
   end loop;
   New_Line;

end idatzi_lista;
```