



Oinarrizko Programazioa

4. Oinarrizko Datu-egiturak



Edukiak

1. Sarrera
2. Programazioko oinarrizko kontzeptuak
3. Programen beheranzko diseinua.
Azpiprogramak: funtzioak eta prozedurak
4. **Oinarrizko datu-egiturak**
5. Programazio-lengoaiei erabilera
6. Aplikazio-adibideak



Oinarrizko datu-egiturak

1. Sarrera
2. Bektoreak
3. Matrizeak
4. Erregistroak
5. Datu-egitura mistoak
6. Listak



Motibazioa

- Egin honako ariketa hau:

Puntuz bukatzen den karaktere-sekuentzia bat emanda, karaktere guztiak baina atzetik aurrera idatziko dituen algoritmoa espezifikatu eta egin.

Nola egin daiteke orain arte ikusitako baliabideekin?



Datu-mota egituratuak

- Orain arte definitutako datu-motak *sinpleak* izan dira: balio bakar bat daukate.
- Datu-mota ***egituratuak*** balio bat baino gehiago biltzen dute.
- Datu-mota egituratu arruntenak bi dira:
 - **Bektorea**: bere osagai guztiak mota berekoak dira
 - **Erregistroa**: osagai guztiak ez dira mota berekoak



Bektoreak

- Bektore motakoa den aldagai bateko **osagai guztiak mota berekoak** dira, eta indize baten arabera erreferentziatzen dira.
- Bektore bat definitzen denean zehazten da bere neurria (osagai kopurua), eta **neurri** hori **finkoa** da.
- Lau datu-mota berri erabili ahal izango dugu:
 - Oso_Bektore
 - Erreal_Bektore
 - Boolear_Bektore
 - Karaktere_Bektore



Bektorearen erazagupena

- Bektore motako aldagai (edo objektu) bat erazagutzeko :
 - Bektorearen izena
 - Osagai kopurua (indizeetarako tartea definituz)
 - Osagaien mota

Adibidez, 10 osagai duten bektoreak definituz:

```
N: constant integer:= 10;  
type Oso_Bektore is array (1..N) of Integer;  
type Karaktere_Bektore is array (1..N) of Character;
```

```
Kalifikazioak: Oso_Bektore;
```

Bektorearen erazagupena (II)

- Adibidez:

`Kalifikazioak: Oso_Bektore;`

Hamar aldagai berri, bakoitzean zenbaki oso bat edukitzeko:

`Kalifikazioak = (6, 5, 8, 4, 7, 6, 9, 7, 5, 8)`

Kalifikazioak(1)
Kalifikazioak(2)
Kalifikazioak(3)
Kalifikazioak(4)
Kalifikazioak(5)
Kalifikazioak(6)
Kalifikazioak(7)
Kalifikazioak(8)
Kalifikazioak(9)
Kalifikazioak(10)



Bektorearen osagaiak

- Bektoreko osagai bat adierazteko, bektorearen izena eta osagaiari dagokion indizea (parentesi artean) idatzi behar dira.
 - Kalifikazioak(6)
Kalifikazioak bektoreko seigarren osagaia da
 - Kalifikazioak(15)
?
 - Kalifikazioak(I)
?
 - Kalifikazioak (2*I-3)
?



Bektoreko osagaiak (II)

- Indizea adierazpen bat ere izan daiteke:
 - `Kalifikazioak(I)`
kalifikazioak bektoreko I-garren osagaia da.
I aldagaiak exekuzio-unean daukan balioa da indizea.
 - `Kalifikazioak(2*I-3)`
Une batean I-ren balioa 4 bada, 5.
- **KONTUZ!** `Kalifikazioak(2*I-3)`
Une batean I-ren balioa 8 bada, 13. osagaia aipatu nahi da.
Horrelakorik ez dagoenez, errore bat sortuko da:
(*Murriztapen-errorea /Constraint Error*)



Bektore datu-mota

- Balio posibleak
 - osagai bakoitzak bere motako balio bat dauka
 - bektore osoak, balio guztiek osaturiko egitura
- Eragiketak
 - osagai bakoitzak bere motako aldagai bakun bat bezala jokatzeko du
 - osagai-kopuru berdineko bektoreen artean
 - Bektore osoko asignazioa ($:=$)
 - Bektore osoko konparazioa ($=$, \neq)



Bektoreak Adan

Oso_Bektore **eta** *Karaktere_Bektore* datu-motak Adaz:

N: constant integer:= 10;

type *Oso_Bektore* is

array (1..N) of Integer;

type *Karaktere_Bektore* is

array (1..N) of Character;

Definitu ditugun *Oso_Bektore* **eta** *Karaktere_Bektore* mota horiek azpiprogrametako parametroetan ere erabil daitezke.



Adako array motaren atributuak

- **A'First:** A arrayaren indizearen barrutiko behe-mugaren balioa
- **A'Last:** A arrayaren indizearen barrutiko goi-mugaren balioa.
- **A'Length:** A arrayaren osagai-kopurua da.
- **A'Range:** A'First..A'Last barrutia adierazten du.

1. adibidea:

```
N: constant integer:= 10;  
type Oso_Bektore is array (1..N) of Integer;  
Kalifikazioak: Oso_Bektore;
```

```
Kalifikazioak'First  = 1  
Kalifikazioak'Last   = 10  
Kalifikazioak'Length = 10  
Kalifikazioak'Range  = 1..10
```



Adako array motaren atributuak (II)

2. adibidea:

```
type Urte_Bektore is array (1901..2050) of Integer;  
Hazkuntzak: Urte_Bektore;
```

```
Hazkuntzak'First   = 1901  
Hazkuntzak'Last    = 2050  
Hazkuntzak'Length  = 150  
Hazkuntzak'Range   = 1901..2050
```



Bektore datu-mota erabiltzen

- Badira bektoreen gainerako eragiketa batzuk, programaziolengoaiek normalean eskaintzen ez dituztenak, baina askotan egiten direnak:
 - Zerbait egin osagai bakoitzarekin
 - Idatzi osagaiak
 - Osagaiei balio bana asignatu
 - Bilatu balio bat
 - Bilatu propietate bat betetzen duen osagai bat
 - Ordenatu osagaiak
 - ...



Erabilera-adibidea

Oso-bektore bat emanda, bektoreko zenbakien batezbesteko aritmetikoa kalkulatzeko duen algoritmoa espezifikatu eta egin. Azpiprograma bezala implementatu.

```
funtzio Batezbestekoa (B: Oso_Bektore) itzuli Erreala
-- Aurrebaldintza:
-- Postbaldintza: Emaita = B bektoreko osagaien batezbesteko aritmetikoa
hasiera
    Batura := 0
    egin I guztietarako B'First tik B'Last raino
        Batura := Batura + B(I)
    amguztietarako
        Emaita := Erreal_Bihurtu (Batura)
                / Erreal_Bihurtu (B'Last-B'First+1)
    itzuli (Emaita)
amaia
```




Erabilera-adibidea (II)

***Oso-bektore bateko osagaiak idatziko dituen algoritmoa espezifikatu eta egin.
Azpiprograma bezala inplementatu.***

```
prozedura Idatzi_Bektorea (B: in Oso_Bektore) is  
-- Aurre:  $B = (z_1, \dots, z_n)$ , ( $z_1, \dots, z_n$ : osoak)  
-- Post:  $G = \langle z_1, \dots, z_n \rangle$ 
```

hasiera

```
    egin I guztietarako B'First tik B'Last raino  
        Idatzi_Osoa (B(I));  
    amguztietarako
```

amaia



Bestelakoak (ikus Watt-en liburua)

```
type Hilabeteak is (Urt,Ots,Mar,Api,Mai,Eka,Uzt,Abu,Ira,Urr,Aza,Abe);  
type Lerroak is array (1 .. 80) of Character;
```

```
Kar_Kop: array (Character) of Natural;  
Letra_Kop: array (Character range 'A' .. 'Z') of Natural;  
Hila_Egunak: array (Hilabeteak) of Integer range 28 .. 31;  
Hileko_Euria: array (Hilabeteak) of Float;  
Urteko_Euria: array (1900 .. 1999) of Hilabeteak;  
Lerroa: Lerroak;  
Pantaila: array (1 .. 24) of Lerroak;
```



Bestelakoak (ikus Watt-en liburua)

```
Hila_Egunak := (31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31);
```

```
subtype Hautagaiak is Integer range 1 .. 4;
```

```
Botu_Kop : array (Hautagaiak) of Natural;
```

```
Botu_Kop := (0, 0, 0, 0);
```

edo

```
Botu_Kop := (1 .. 4 => 0);
```



Bestelakoak (ikus Watt-en liburua)

String datu-mota

Formalki honela definitzen da.

```
type String is array (Positive range <>) of Character;
```

Karaktere_Bektore motaren antzekoa da zenbait gehigarriekin:

- <, <=, >= eta > eragile erlazionalen bidez konpara daitezke.
- Array osoa idazkera laburtuan adieraz daiteke:

```
Egilea: array (1 .. 3) of String (1 .. 10);
```

```
Egilea(1) := "Atxaga"; -- 10 karaktere derrigorrez
```

- Konstanteetan tamaina ez da definitu behar:

```
Liburua : constant String := "Ada: Lengoaia eta Metodologia";
```



Matrizeak

- Bektore bat osatzen duten aldagai guztiak mota berekoak dira, eta indize **bakar baten** arabera erreferentziazten dira.
- Matrizeetan ere aldagai guztiak mota berekoak dira, baina **bi indize** erabiltzen dira osagaiak bereizteko
 - (Behar beste indize erabil daiteke, eta horrela dimentsio anitzeko egiturak sortu)
- Lau datu-mota berri erabili ahal izango dugu:
 - `Oso_Matrize`
 - `Erreal_Matrize`
 - `Boolear_Matrize`
 - `Karaktere_Matrize`



Matrizeen erazagupena

- Matrize motako aldagai (edo objektu) bat erazagutzeko :
 - Bektorearen izena
 - Osagai kopuruak (indize bietarako tarteak definituz)
 - Osagaien mota

Adibidez:

N: constant integer:= 3;

N_Lerro: constant Integer :=24;

N_Zutabe: constant Integer :=80;

type Oso_Matrize is array (1..N, 1..N) of Integer;

type Karaktere_Matrize is array (1..N_Lerro, 1..N_Zutabe) of Character;

M1, M2: Oso_Matrize;

Pantaila: Karaktere_Matrize;



Matrizearen osagaiak

- Matrizeko osagai bat adierazteko, matrizearen izena eta osagaiari dagozkion indizeak idatzi behar dira.
 - $\text{Pantaila}(6, 25)$
Pantaila matrizeko 6. errenkadako 25. osagaia da
 - $M1(I, J+1)$
 $M1$ matrizeko I . errenkadako $J+1$. osagaia da



Dimentsio anitzeko egiturak

Hiru edo indize gehiago dituzten array-ak ere erabil daitezke:

```
type N_Egitura is array (1..20, 1..20, 1..5) of Integer;
```




Matrizeen atributuak Adan

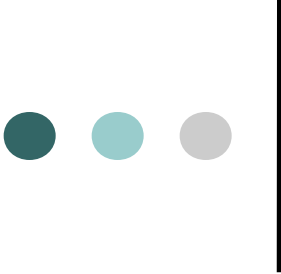
- **A'First(1)**: A matrizearen lehenengo indizearen barrutiko behe-mugaren balioa
- **A'First(2)**: bigarrenarena
- **A'Last(1)** , **A'Last(2)** : A matrizearen lehenengo eta bigarren indizearen barrutiko goi-mugaren balioak.
- Antzera: **A'Length(1)**, **A'Length(2)**, **A'Range(1)**, **A'Range(2)**

Adibidez:

```
M: array (2000..2005, 1..15) of Integer;
```

```
M'First(1) = 2000           M'Last(1) = 2005
```

```
M'First(2) = 1             M'Last(2) = 15
```



Matrize datu-mota erabiltzen

- Badira matrizeen gainerako eragiketa batzuk, programaziolengoaiek normalean eskaintzen ez dituztenak, baina askotan egiten direnak:
 - Zerbait egin osagai bakoitzarekin
 - Idatzi osagaiak
 - Osagaiei balio bana asignatu
 - Bilatu balio bat
 - Bilatu propietate bat betetzen duen osagai bat
 - Ordenatu osagaiak
 - Bi matrize batu, biderkatu...



Erabilera-adibidea

Oso-matrize bat emanda, matrizeko zenbakien batezbesteko aritmetikoa kalkulatzeko duen algoritmoa espezifikatu eta egin. Azpiprograma bezala implementatu.

```
funtzio Batezbestekoa (B: Oso_Matrize) itzuli Erreala
-- Aurrebaldintza:
-- Postbaldintza: Eraitza = B matrizeko osagaien batezbesteko aritmetikoa
hasiera
    Batura := 0
    egin I guztietarako B'First(1) tik B'Last(1) raino
        egin J guztietarako B'First(2) tik B'Last(2) raino
            Batura := Batura + B(I, J)
        amguztietarako
    amguztietarako
    Eraitza := Erreal_Bihurtu (Batura)
                / Erreal_Bihurtu ((B'Last(1)-B'First(1)+1) * (B'Last(2)-B'First(2)+1))
    itzuli (Eraitza)
amaia
```



Erabilera-adibidea (II)

***Oso-matrize bateko osagaiak idatziko dituen algoritmoa espezifikatu eta egin.
Azpiprograma bezala inplementatu.***

```
prozedura Idatzi_Matrizea (B: in Oso_Bektore) is  
  -- Aurre:  
  -- Post: G (oso-sekuentzia), B matrizeko osagaiak idatzi dira  
hasiera  
    egin I guztietarako M'First(1) tik M'Last(1) raino  
      egin J guztietarako M'First(2) tik M'Last(2) raino  
        Idatzi_Osoa (M(I, J));  
      amguztietarako  
        Lerro_Berrira_Pasa;  
    amguztietarako  
amaia
```



Erregistroak

- Datu-mota *egituratuek* balio bat baino gehiago biltzen dute.
- Datu-mota egituratu arruntenak bi dira:
 - Bektorea:
 - osagai guztiak mota berekoak dira
 - osagaiak indexazio bidez erreferentziatuko dira
 - **Erregistroa:**
 - osagai guztiak ez dira mota berekoak
 - osagaiak identifikadore bidez erreferentziatuko dira



Erregistroen abantaila

- Objektu-bilduma bat erregistro batean biltzearen abantailak:
 - objektuek batak bestearekin duten erlazioa esplizituki adierazten da.
 - erregistroa objektu bakun bat bezala maneiatu edo bere osagaiak indibidualki maneiatzearen artean aukera dezakegu, uneko beharren arabera.



Erregistroen erazagupena

- Erregistro motako objektu bat erazagutzeko:
 - Objektuaren izena
 - Erregistro-motaren definizioa
 - erregistro-osagai bakoitzaren identifikadorea eta bere mota zehazten ditu
- Adibidez:

Hitza: **record**

Karaktereak: `String(1..20);`

Luzera: `Integer;`

end record;



Erregistroen erazagupena (II)

```
Hitza: record  
    Karaktereak: String(1..20) ;  
    Luzera: Integer ;  
end record;
```

```
type Hitz is record  
    Karaktereak: String(1..20) ;  
    Luzera: Integer ;  
end record;  
  
Hitza: Hitz ;
```

- Bi erazagupenak baliokideak dira
- Komeni da *Hitz* mota definitzea, azpiprogrametako parametroekin erabili ahal izateko



Erregistroen osagaien erreferentzia

- Erregistroaren izena, puntu bat eta osagaiari dagokion identifikadorea idatzi behar dira.
- Adibidez:
 - *Hitza.Luzera*
 - Hitza aldagaiaren luzera.
 - *Hitza.Karaktereak*
 - Hitza aldagaiaren *Karaktereak* taula
 - *Hitza.Karaktereak(2)*
 - Hitza aldagaiaren *Karaktereak* taulako 2. osagaia.



Erregistro datu-mota

- Balio posibleak
 - Osagai bakoitzak bere motako balio bat dauka
 - Erregistro osoak balio guztiek osaturiko egitura
- Eragiketak
 - Osagai bakoitzak bere motako aldagai bakun bat bezala jokatzeko du
 - Mota bereko erregistroen artean
 - Erregistro osoko asignazioa ($:=$)
 - Erregistro osoko konparazioa ($=$, \neq)



Datu-egitura mistoak

- Datu-egitura:
 - Datu-elementuen bilduma, osagaien arteko erlazioa erakusteko moduan antolatuta.
- Datu-egituren oinarritzko eraikuntza-blokeak:
 - Bektoreak eta erregistroak.
- Bektoreen eta erregistroen osagaiak eurak ere bektoreak edo erregistroak izan daitezke.
- Horrek nahi adinako konplexutasuneko datu-egiturak definitzeko aukera ematen digu.



Erabilera-adibidea

Puntuz bukatzen den karaktere-sekuentzia bat emanda, karaktere guztiak baina atzetik aurrera idatziko dituen algoritmoa espezifikatu eta egin.

- Demagun gehien jota 100 karaktere izango direla.
- Horrelako datu-egitura bat erabiliko dugu:

```
N: constant Integer := 100;  
type Karaktere_Bektore is array (1..N) of Character;  
type Karaktere_Lista is record  
    Karaktereak: Karaktere_Bektore;  
    Luzera: Integer;  
end record;  
  
S: Karaktere_Lista;
```



Diseinua

algoritmo Alderantziz

-- Aurre: $F = \langle k_1, \dots, k_n, '.' \rangle$, (k_1, \dots, k_n : karaktereak), non:

-- $k_1, \dots, k_n \neq '.'$ eta $n < 100$

-- Post: $G = \langle k_n, k_{n-1}, \dots, k_1 \rangle$

S : Karaktere_Lista

hasiera

Testua_Gorde(S)

Idatzi_Atzekoz_Aurrera (S)

amaia



Diseinua (II)

```
Algoritmo Testua_Gorde (S: emaitza Karaktere_Lista)
-- Aurre: F=<k1,...,kn, '.'>, (k1,...,kn: karaktere), non:
--      k1,...,kn ≠ '.' eta 0 ≤ n < 100
-- Post: S.Karakterek=(k1,k2,...,kn,...) eta S.Luzera = n
```

hasiera

```
    Irakurri_Karakterea (Kar)
    I := 0
    bitartean Kar /= '.' egin
        I := I + 1
        S.Karakterek(I) := Kar
        Irakurri_Karakterea (Kar)
    ambitartean
    S.Luzera := I
```

amaia



Diseinua (III)

```
Algoritmo Idatzi_Atzekoz_Aurrera (S: datu Karaktere_Lista)
-- Aurre: S.Karaktereak=( $k_1, k_2, \dots, k_n, \dots$ ) eta S.Luzera = n ( $0 \leq n < 100$ )
-- Post: G=< $k_n, k_{n-1}, \dots, k_1$ >
hasiera
    egin I guztietarako alderantziz 1 etik S.Luzera raino
        Idatzi_Karakterea (S.Karaktereak(I))
    ambitartean
amaia
```

Oharra, Adaz honela izan zitekeen:

```
for I in reverse 1..S.Luzera loop
    Idatzi_Karakterea (S.Karaktereak(I));
end loop;
```

Adibideak

```
type Hitz is record
```

```
    Karaktereak: String(1..20);  
    Luzera: Integer;
```

```
end record;
```

```
type Hiztegi is array(1..1000) of Hitz;
```

```
H: Hiztegi;
```

H (1)

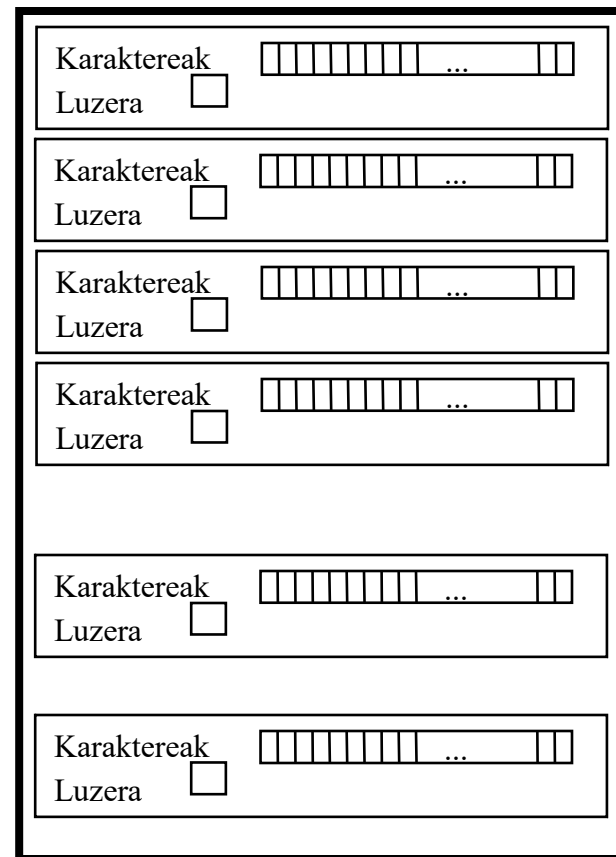
H (2)

H (3)

H (4)

H (999)

H(1000)





Adibideak (II)

```
type Talde is array(1..100) of Ikasle;  
type Ikasle is record  
    Kodea: Integer;  
    Izena, Deitura1, Deitura2: String (1..20);  
    Kalifikazioak: Oso_Bektore (1..10);  
end record;
```



Adibideak (III)

```
type Talde is record  
    Ikasleak: array(1..100) of Ikasle;  
    Batezbestekoak: Erreal_Bektore(1..10);  
end record ;
```

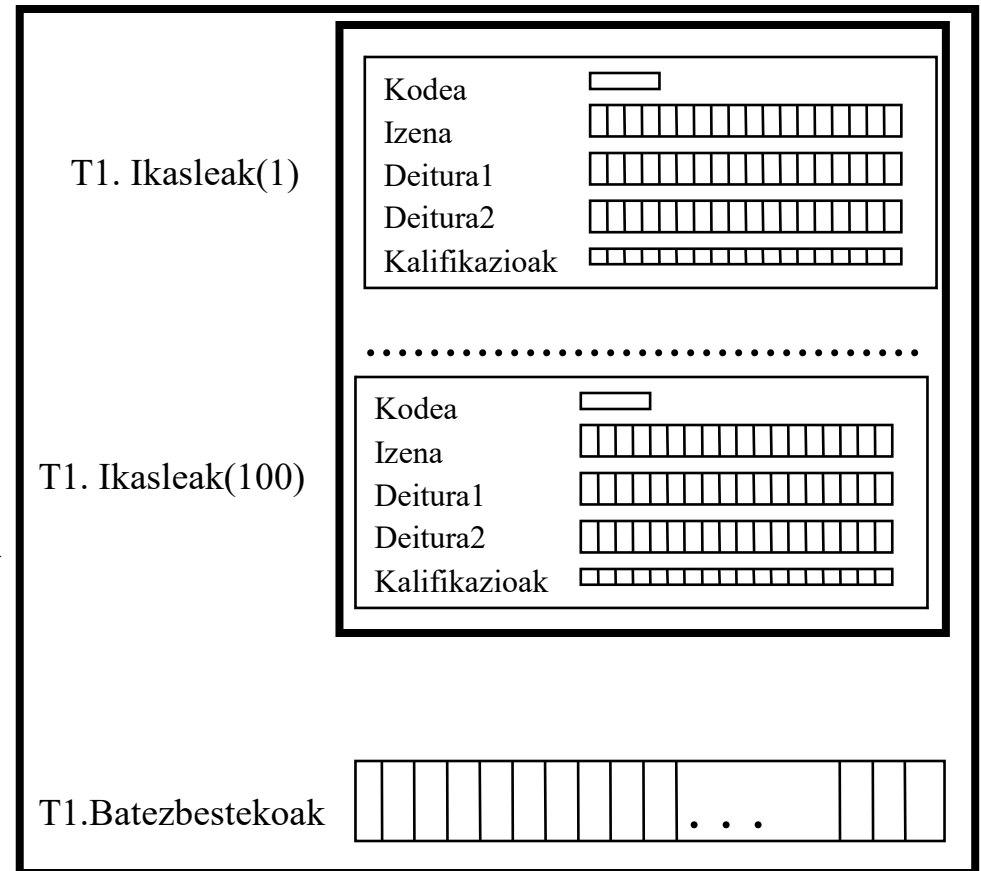
```
type Ikasle is record  
    Kodea: Integer;  
    Izena, Deitural, Deitura2: string(1..20);  
    Kalifikazioak: Oso_Bektore(1..10);  
end record;
```

Adibideak (IV)

```
type Talde is record  
  Ikasleak: array(1..100) of Ikasle;  
  Batezbestekoak: Erreal_Bektore(1..10);  
end record ;
```

```
type Ikasle is record  
  Kodea: Integer;  
  Izena, Deitura1, Deitura2: String(1..20);  
  Kalifikazioak: Oso_Bektore(1..10);  
end record;
```

```
T1: Talde;
```





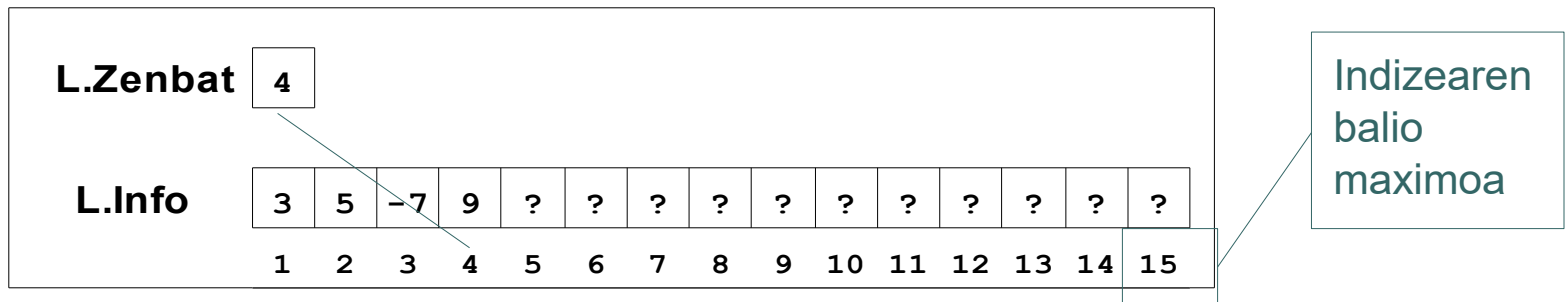
Listak

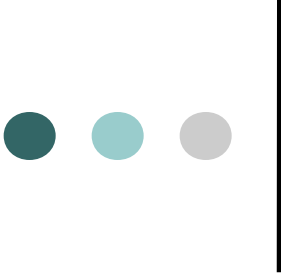
- Lista motako aldagaiek bi ezaugarri nagusi dituzte:
 - Luzera aldakorra
 - Elementuak txertatu eta ezabatu egin daitezke
 - Osagai guztiak mota berekoak dira
- Oinarrizko lista motak:
 - Oso_Lista
 - Erreal_Lista
 - Boolear_Lista
 - Karaktere_Lista
- Oinarrizko listez gain, osagai egituratuak dituztenak ere defini daitezke

Lista estatikoak

- Egitura mistoak dira, bi eremu dituztenak:
 - Osagaien kopurua
 - Osagaien bektorea
- Lista estatikoak definitzean, osagaien kopuru maximoa ezarri behar da.
- Adibidez:

$L = \langle 3, 5, -7, 9 \rangle$





Lista estatikoak

Erazagupena

- Honela errepresentatzen dira Adan (zenbaki osoen) lista estatikoak:

```
Max : constant Natural := 15;  
subtype Indize1_Max is Natural range 1 .. Max;  
subtype Indize0_Max is Natural range 0 .. Max;  
type Oso_Bektore is array (Indize1_Max) of Integer;  
type Oso_Lista is record  
    Zenbat : Indize0_Max;    -- listako elementuen kopurua  
    Info : Oso_Bektore;      -- osagaien bektorea  
end record;
```



Lista estatikoak Erabilera

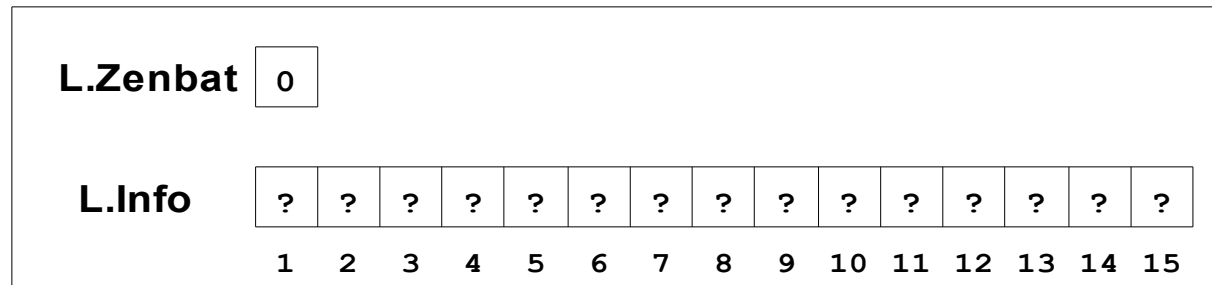
- Hasieratzea
 - Lista hutsa sortzea
 - Irakurriz kargatzea
- Lista bateko elementu **guztiei** tratamendu bat egitea (eskema iteratibo totala)
- Lista bateko elementu **batzuei** tratamendu bat egitea (eskema iteratibo partziala)
- Elementu bat txertatzea
 - Desplazamenduaren beharra
- Elementu bat ezabatzea
 - Desplazamenduaren beharra
- Ordenatzea

Lista estatikoak

Erabilera (II)

- Lista hutsa sortzea

```
L: Oso_Lista;  
-- L hutsa:  
L= < >
```



```
L.Zenbat := 0;
```


Lista estatikoak Erabilera (III)

- Hasieratzea

```
L: Oso_Lista;
```

```
-- demagun horrela hasieratu nahi dugula L lista:
```

```
  L= < 3, 5, -7, 9>
```

L.Zenbat

4

L.Info

3	5	-7	9	?	?	?	?	?	?	?	?	?	?	?
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

```
L.Zenbat:=4;
```

```
L.Info(1..L.Zenbat) := (3,5,-7,9);
```

```
-- L.Info(1) = 3
```

```
-- ..
```

```
-- L.Info(L.Zenbat) = 9
```



Lista estatikoak Erabilera (IV)

- Oso-lista bat hasieratzea 0 zenbakiarekin bukatzen den zenbaki sekuentzia bat sarrera estandarretik irakurritz.

```
Algoritmo Irakurri_Oso_Lista (L: emaitza Oso_Lista)
-- Aurre: F=<z1,...,zn,0>, (z1,...,zn: oso), non:
--           z1,...,zn ≠ 0 eta 0 ≤ n < 15
-- Post: L.Info=(z1,z2,...,zn,...) eta L.Zenbat = n
```

hasiera

```
Irakurri_Osoa (Z)
I:= 0
bitartean  Z /= 0  egin
    I := I + 1
    L.Info(I) := Z
    Irakurri_Osoa (Z)
amaitartean
L.Zenbat := I
```

amaia

Lista estatikoak Erabilera (V)

- Elementu bat txertatzea.
 - Adibidez, txertatu 6 osoa 3. posizioan:

Hasierako egoera:

$L = \langle 3, 5, -7, 9 \rangle$

L.Zenbat	4																														
L.Info	<table><tr><td>3</td><td>5</td><td>-7</td><td>9</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr></table>	3	5	-7	9	?	?	?	?	?	?	?	?	?	?	?	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
3	5	-7	9	?	?	?	?	?	?	?	?	?	?	?																	
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15																	

Bukaerako egoera:

$L = \langle 3, 5, 6, -7, 9 \rangle$

L.Zenbat

5

L.Info

3	5	6	-7	9	?	?	?	?	?	?	?	?	?	?
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Lista estatikoak Erabilera (VI)

- Elementu bat ezabatzea.
 - Adibidez, ezabatu 2. posizioko elementua:

Hasierako egoera:

$L = \langle 3, 5, -7, 9 \rangle$

L.Zenbat	4																														
L.Info	<table><tr><td>3</td><td>5</td><td>-7</td><td>9</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr></table>	3	5	-7	9	?	?	?	?	?	?	?	?	?	?	?	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
3	5	-7	9	?	?	?	?	?	?	?	?	?	?	?																	
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15																	

Bukaerako egoera:

$L = \langle 3, -7, 9 \rangle$

L.Zenbat

3

L.Info

3

-7

9

?

?

?

?

?

?

?

?

?

?

?

?

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15