Oinarrizko Programazioa

Klaseetako materiala

Datu-mota egituratuak

(4. gaia)

Arantza Díaz de Ilarraza Sánchez

Kepa Sarasola Gabiola

Lengoaiak eta Sistema Informatikoak Saila

Euskal Herriko Unibertsitatea

Programazioa I

Gai zerrenda:

1. **Sarrera.**
2. **Programazioko oinarrizko kontzeptuak.**
3. ***Programen beheranzko diseinua.***
   Azpiprogramak: funtzioak eta prozedurak.
4. ***Datu-mota egituratuak.***
5. **Programazio-lengoaien erabilera.**
6. **Aplikazio-adibideak.**

4. Datu-Mota Egituratuak

4.1 Sarrera

4.2 Bektoreak

4.3 Matrizeak

4.4 Erregistroak

4.5 Datu-egitura mistoak

4.6 Listak

Motibazioa

Ariketa hau ezin da egin
orain arte aurkeztu ditugun tresnekin:

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

Datu-mota Egituratuak

- Orain arte definitutako datu-motak *sinpleak* izan dira: balio bakar bat daukate.

  Bakunak edo ***eskalarrak*** ere esaten zaie
- Datu-mota ***egituratuek*** 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 bat osatzen duten aldagai guztiak mota berekoak dira, eta indize baten arabera erreferentziatzen dira
- Lau datu-mota berri erabili ahal izango dugu:
  - Osoko_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:

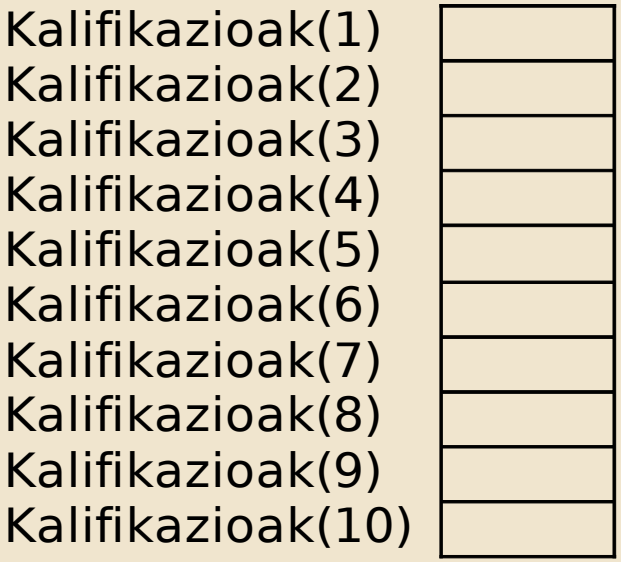
```
N: constant integer:= 10;
type Osoko_Bektore is array (1..N) of Integer;
type Karaktere_Bektore is array (1..N) of Character;

Kalifikazioak: Osoko_Bektore;
```

Bektorearen erazagupena

- Adibidez:

  Kalifikazioak: Osoko_Bektore;

  Hamar aldagai berri, bakoitzean zenbaki oso bat edukitzeko::



Bektoreko osagaiak

- Bektoreko osagai bat adierazteko, bektorearen izena eta osagaiari dagokion indizea idatzi behar dira.
  - Kalifikazioak(6)

    Kalifikazioak bektoreko seigarren osagaia da

  - Kalifikazioak(15)

    ?

  - Kalifikazioak(I)

    ?

  - Kalifikazioak (2*I-3)

    ?

Bektoreko osagaiak

- 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. Osagaia aipatzen da
- **KONTUZ!** Kalifikazioak(2*I -3)

  Une batean I-ren balioa 8 bada, 13. osagaia aipatu nahi da.

  Horrelakorik ez dagoenez, errore bat sortuko da.

  (Murrizpen-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 jokatzen du
  - osagai-kopuru berdineko bektoreen artean
    - Bektore osoko asignazioa
    - Bektore osoko konparazioa (=, /=)

Bektoreak Adan

`Osoko_Bektore` datu-mota Adaz:

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

Definitu ditugun `Osoko_Bektore` eta `Karaktere_Bektore` mota horiek azpiprogrametako parametroetan ere erabil daiteke.

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.

Adibidez:

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

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

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.

Adibidez:

```
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

Programazio-lengoaiak eskaintzen ez dituzten baina bektoreekin askotan egiten diren eragiketak:

- Zerbait egin osagai bakoitzarekin
- Idatzi osagaiak
- Irakurri balioak osagaiak definitzeko
- Bilatu balio bat
- Bilatu propietate bat betetzen duen osagai bat
- Ordenatu osagaiak
- ...

Erabilera-adibidea

Osoko-bektore bat emanda, bektoreko zenbakien batezbesteko aritmetikoa kalkulatzen duen algoritmoa espezifikatu eta egin. Azpiprograma bezala inplementatu.

```
funtzio Batezbestekoa (B: Osoko_Bektore) itzuli  Erreala
```

Aurrebaldintza:

Postbaldintza: Emaitza = B bektoreko osagaien batezbesteko aritmetikoa

```
 hasiera
    Batura := 0
    egin I guztietarako B'First tik B'Last raino
    Batura := Batura + B(I)
    amguztietarako
    Emaitza := Erreal_Bihurtu (Batura)
               / Erreal_Bihurtu (B'Last-B'First+1)
    itzuli (Emaitza)
amaia
```

Erabilera-adibidea

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

```
prozedura Idatzi_Bektore (B: in Osoko_Bektore) is
-- Aurre:
-- Post: S (osoko-sekuentzia) B bektoreko osagaiak idatzi dira
hasi
   egin I guztietarako B'First tik B'Last raino
      Idatzi_Osokoa (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);
 edo
Hila_Egunak := (ots => 28, Api | Eka | Ira | Aza => 30,
          Urt | Mar | Mai | Uzt | Abu | Urr | Abe => 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)*

```
Botu_Kop      : array (Hautagaiak) of Natural
                          := (Hautagaiak => 0);
Hila_Egunak : constant array (Hilabeteak) of Integer
              := (Ots => 28,
                  Mar | Eka | Ira | Aza => 30,
                  Urt | Mar | Mai | Uzt | Abu | Urr | Abe => 31);
```

Bestelakoak *(ikus Watt-en liburua)*

String ***datu mota***

Formalki honela definitzen da.

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

Gure *Karaktereen_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 bat**en arabera erreferentziatzen 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:
  - Osoko_Matrize
  - Erreal_Matrize
  - Boolear_Matrize
  - Karaktere_Matrize

Matrizeen erazagupena

- Matrize motako aldagai bat erazagutzeko :
  - Matrizearen izena
  - Osagai kopuruak (indize bietarako tartea definituz)
  - Osagaien mota

Adibidez:

```
N: constant integer:= 3;
N_lerro: constant Integer :=24;
N_Zutabe: constant Integer :=80;
type Osoko_Matrize is array (1..N, 1..N) of Integer;
type Karaktere_Matrize is array (1..N_lerro, 1..N_Zutabe) of Character;

M1, M2: Osoko_Matrize;
Pantaila: Karaktere_Matrize;
```

Matrizeen osagaiak

- Matrizeko osagai bat adierazteko, bektorearen izena eta osagaiari dagozkion indizeak idatzi behar dira.
  - Pantaila(6, 25)

    Pantaila matrizeko 6. errenkadako 25. osagaia da
  - M1(I, J+1)

    M1 matrizeko I. errenkadako J+1. osagaia da

Matrizeak

Hiru edo indize gehiago dituen array bat ere erabil daiteke:

```
type N_Bektore 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 balioa.
- Antzera: A'Length(1), A'Length(2), A'Range(1), A'Range(2)

Adibidez:

M1 : 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

Programazio-lengoaiak eskaintzen ez dituzten baina bektoreekin askotan egiten diren eragiketak:

- Zerbait egin osagai bakoitzarekin
- Idatzi osagaiak
- Irakurri balioak osagaiak definitzeko
- Bilatu balio bat
- Bilatu propietate bat betetzen duen osagai bat
- Ordenatu osagaiak
- Bi matrize batu, biderkatu, ...

Erabilera-adibidea

Osoko_Matrize bat emanda, matrizeko zenbakien batezbesteko aritmetikoa kalkulatzen duen algoritmoa espezifikatu eta egin. Azpiprograma bezala inplementatu.

```
funtzio Batezbestekoa (B: Osoko_Matrize) itzuli  Erreala
```

Aurrebaldintza:

Postbaldintza: Emaitza = B bektoreko 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
    Emaitza := Erreal_Bihurtu (Batura)
               / Erreal_Bihurtu ((B'Last(1)-B'First(1)+1) * (B'Last(2)-B'First(2)+1))
    itzuli (Emaitza)
amaia
```

Erabilera-adibidea

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

```
prozedura Idatzi_Matrize (M: in Osoko_Matrize) is
-- Aurre:
-- Post: S (osoko-sekuentzia) M matrizeko osagaiak idatzi dira
hasi
   egin I guztietarako M'First(1) tik M'Last(1) raino
      egin J guztietarako M'First(2) tik M'Last(2) raino
         Idatzi_Osokoa (M(I, J));
      amguztietarako
      Lerro_Berrira_Pasa;
   amguztietarako
amaia;
```

Erregistroak (tuplak)

- 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 bateko osagai moduan 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

```
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.Ad.:
  - *Hitza.Luzera*

Hitza aldagaiaren luzera.

  - *Hitza.Karaktereak*
    - Hitza aldagaiaren *Karaktereak* taula
  - *Hitza.Karaktereak(2)*
    - Hitza aldagaiaren *Karaktereak* taulako 2. osagaia.

Erregistro datu-motak

- Balio posibleak
  - osagai bakoitzak bere motako balio bat dauka
  - erregistro osoak balio guztiek osaturiko egitura
- Eragiketak
  - osagai bakoitzak bere motako aldagai bakun bat bezala jokatzen du
  - mota bereko erregistroen artean
    - erregistro osoko asignazioa
    - erregistro osoko konparazioa (=, /=)

Datu-egitura mistoak

- Datu-egitura:
  - Osagaien arteko erlazioa erakusteko moduan antolatuta dagoen datu-elementuen bilduma.
- Datu-egituren oinarrizko eraikuntza-blokeak:
  - Bektoreak eta erregistroak.
- Bektoreen eta erregistroen osagaiak eurak ere bektoreak edo erregistroak izan daitezke.
- Honek nahi adinako konplexutasuneko datu-egiturak definitzeko ahalbidetzen gaitu.

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_Sekuentzia is record
   Karaktereak: Karaktere_Bektore
   Luzera: Integer ;
end record;
S : Karaktere_Sekuentzia
```

Diseinua

algoritmo Alderantziz

Aurrebaldintza: S1 (karaktere-sekuentzia). Puntuz bukatzen da eta ez dago beste punturik. Gehienez 100 karaktere dauka.

Postbaldintza: S2 (karaktere-sekuentzia). S1 sekuentziako karaktereak dauzka baina alderantzizko ordenan

```
Bektore : Karaktere_Sekuentzia
hasiera
   Testua_Gorde(S)
   Idatzi_Atzekoz_Aurrera (S)
amaia
```

Diseinua

Algoritmo Testua_Gorde (S1: **emaitza** Karaktere_Sekuentzia)

Aurrebaldintza: S (karaktere-sekuentzia). Puntuz bukatzen da eta ez dago beste punturik.

Postbaldintza: S1.Luzera-k adierazten du S sekuentziaren karaktere-kopurua puntua kontatu gabe. S sekuentziako karaktereak S1 sekuentzian gorde dira lehenengo posiziotik S1.luzera-garren posiziorainо

```
hasiera
   Irakurri_Karakterea (Kar)
   I:= 0
   bitartean  Kar /= '.'  egin
    I := I + 1
    S1.Karaktereak(I) := Kar
    Irakurri_Karakterea (Kar)
   ambitartean
   S1.Luzera := I
amaia
```

Diseinua

Algoritmo Idatzi_Atzekoz_Aurrera (S: **datu** Karaktere_Sekuentzia)

Aurrebaldintza: S.Luzera>=0

Postbaldintza: SF (karaktere-sekuentzia) S sekuentziako karaktereak pantailan idatzi dira lehenengo posiziotik S.Luzera-garren posizioraino

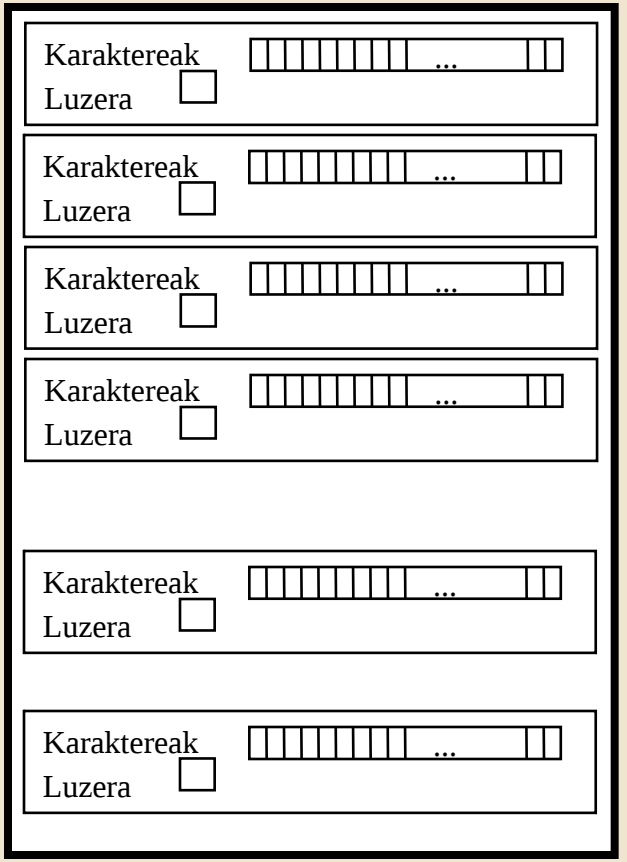
```
hasiera
   egin I guztietarako S'First tik S'First+S.Luzera-1 raino
        Idatzi_Karakterea (S.Karaktereak(I))
   ambitartean
amaia
```

Oharra, Adaz honela izan zitekeen:

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

Adibideak (1)

```
type Hitz is record
        Karaktereak: string(1..20) ;
        Luzera:    integer ;
    end record;

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

H: Hiztegi;
```



Adibideak (2)

- **type** Talde **is array**(1..100) **of** Ikasle;

  type Ikasle **is record**

  Kodea: integer;

  Izena, Deitura1, Deitura2: string (1..20) ;

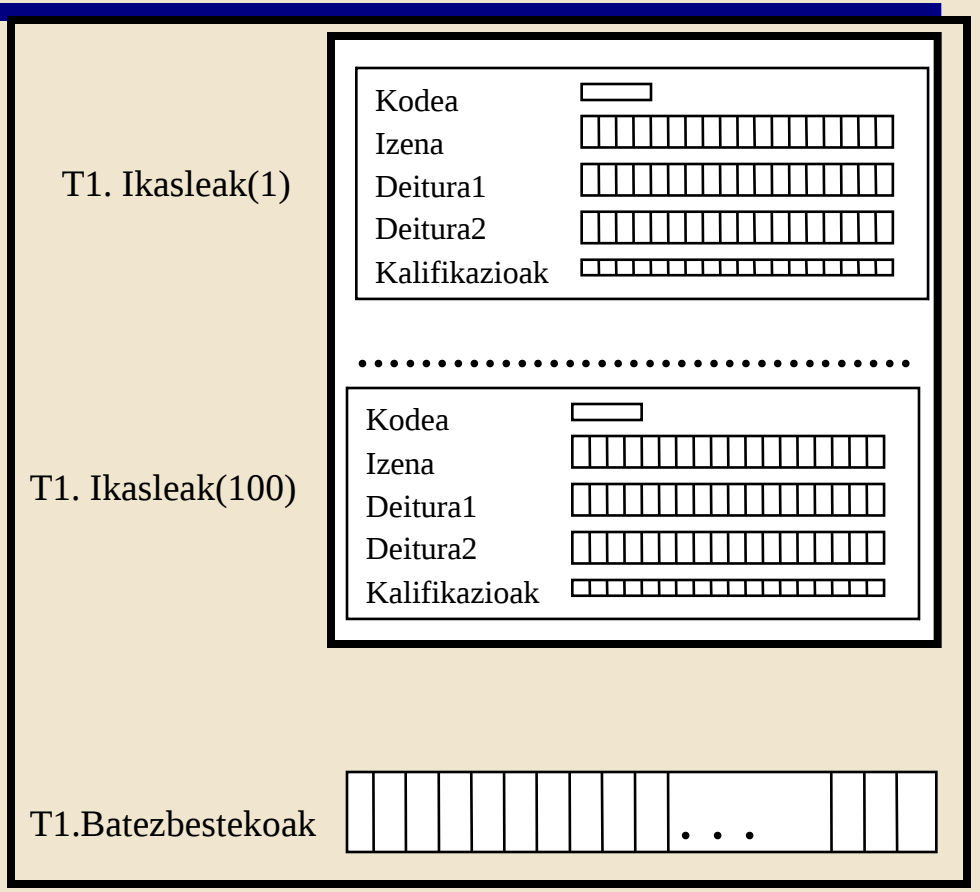
  Kalifikazioak: Osokoen_Bektore (1..10) ;

  end record;

Adibideak (3)

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


type Ikasle is record
    Kodea: integer;
    Izena,Deitura1,Deitura2:String(1..20) ;

    Kalifikazioak: Osokoen_Bektore(1..10);
 end record;
```

T1: Talde;



Adibideak (4)

```
type Talde is record
    Ikasleak: array(1..100) of Ikasle;
    Batezbestekoak: Errealen_Bektore(1..10)
end record ;
type Ikasle is record
    Kodea: integer;
    Izena, Deitura1, Deitura2: string(1..20) ;
    Kalifikazioak: Osokoen_Bektore(1..10) ;
 end record;
```