

4.2 Gaia. Datu egituren diseinua erregistroekin

eman la zabal zazu



Universidad
del País Vasco

Euskal Herriko
Unibertsitatea

Aurkibidea

- **Gaiaren helburuak**
- Motibazioa
- Erregistroak
- Habiaratutako egiturak



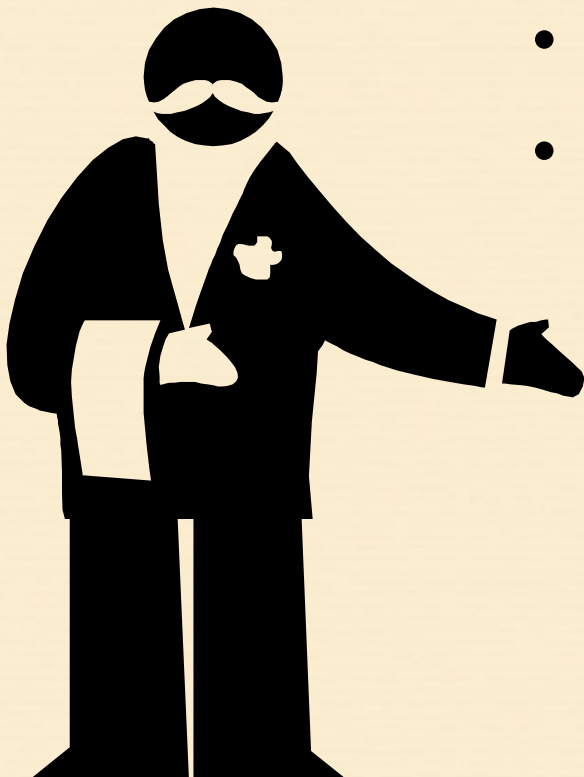
Gaiaren helburuak

- Erabili ditzakegun datu egiturak zabaltzea
 - Mota ezberdinetako elementuak gordetzeko erregistroak
 - Egitura habiaratuak
 - Erregistroak erregistroekin
 - Erregistroen arrayak
 - Erregistroak arrayekin



Aurkibidea

- Gaiaren helburuak
- **Motibazioa**
- Erregistroak
- Habiaratutako egiturak



Motibazioa

- Bizitza errealeko informatikaren atazek, datu egitura konplexuetan gordetako datuak erabiltzen dituzte
 - Oinarrizko Programazioa ikasgaiko ikasleen zerrenda
 - UPV/EHUko ikasle baten
 - ...
- Ezaugarri horietako datuak gordetzea ahalbidetuko diguten egiturak nola diseinatzaren diren ikusiko dugu

Motibazioa

- Gai honetan, ADA programazio lengoaia baino ez dugu ikusiko
 - Pythonek horrelako erregistroak simulatu ditzake, baina ohikoagoa da objektuak erabiltzea da. Objektuak bigarren lauhilkoan ikusiko dituzue (Javarekin)



Aurkibidea

- Gaiaren helburuak
- Motibazioa
- **Erregistroak**
- Habiaratutako egiturak



Erregistroa

- Datu mota egituratua da, zeinak mota berdineko zein ezberdineko datu multzo bat biltegitzen duen
 - Motaren definizioan, erregistroa osatzen duen eremu edo balio bakoitzari datu mota bat esleitzen zaio.

```
type Ikaslea is record  
    Esp zen: Integer;  
    Izena, Abizena: String(1..30);  
    Ikasturtea: Integer;  
    Taldea: Character;  
end record;
```



Erregistro motako aldagaiak

- Gainerako aldagaietan ikusitakoaren arabera erazagutzen dira

```
Ikas1, Ikas2: Ikaslea;
```

	Ikas1
Esp_Zen	??
Izena	??
Abizena	??
Kurtsoa	??
Taldea	??

	Ikas2
Esp_Zen	??
Izena	??
Abizena	??
Kurtsoa	??
Taldea	??

Erregistroko eremuetara atzipena

```
get(Ikas1.Esp_Zen);  
get(Ikas1.Izena);  
Ikas1.Abizena := "Cousteau" -- 30 karaktere behar ditu!!!  
Ikas1.Talde := 'A';  
Ikas2.Kurtsoa := 1;  
Ikas1.Kurtsoa := Ikas2.Kurtsoa;
```

	Ikas1
Esp_Zen	1569
Izena	Iker
Abizena	Cousteau
Kurtsoa	1
Talde	'A'

	Ikas2
Esp_Zen	??
Izena	??
Abizena	??
Kurtsoa	1
Talde	??

Esleipena

```
Ikas2:= Ikas1;
```

	Ikas1
Esp_Zen	1569
Izena	Iker
Abizena	Cousteau
Kurtsoa	1
Talde	'A'

	Ikas2
Esp_Zen	1569
Izena	Iker
Abizena	Cousteau
Kurtsoa	1
Talde	'A'

Esleipena

```
Ikas2.Izena := "Jacques";  
Ikas2.Kurtsoa := 5;  
Ikas1 := Ikas2;
```

	Ikas1
Esp_Zen	1569
Izena	Jacques
Abizena	Cousteau
Kurtsoa	5
Talde	'A'

	Ikas2
Esp_Zen	1569
Izena	Jacques
Abizena	Cousteau
Kurtsoa	5
Talde	'A'

Erregistroekin eragiketak

- Erregistro motako aldagai baten eremu baten atzipenera eragiketa:
 - `Ikas1.Izena := Ikas2.Izena;`
 - `if (Ikas1.Kurtsoa = 2) then ...`
- Erregistro oso baten balioa esleitu mota berdineko aldagai bati:
 - `Ikas1 := Ikas2;`



Erregistroekin eragiketak

- Erregistro mota bereko balioen arteko konparaketa
 - $Ikas1 = Ikas2$
 - $Ikas1 \neq Ikas2$
- $<$, $>$, \leq eta \geq konparatzaileek ez dute zentzurik

Moten arteko komuztadura

```
type Ikaslea is record
  Esp_Zen: Integer;
  Izena, Abizena: String(1..30);
  Kurtsoa: Integer;
  Taldea: Character;
end record;
```

```
type Produktua is record
  Izena: Character;
  Prezioa: Integer;
  Izakinak: Integer;
end record;
```

```
-- Aldagaien erazagupena
Ikas1, Ikas2: Ikaslea;
ProdA, ProdB, ProdB: Produktua;

-- Esleipen posibleak
Ikas1 := Ikas2;
ProdA := ProdB;
Ikas1.Kurtsoa := ProdB.Izakinak;

-- Esleipen okerrak
Ikas1 := ProdB;
ProdA.Izakinak := Ikas1.Izena;
Ikas1.Izena := ProdB.Izena;
```



Erregistroen erabileraren abantailak

- Askoz aldagai eta parametro gutxiago definitzen dira
 - Ikasle1, Ikasle2, Ikasle3 vs
 - Zen_Esp1, Zen_Esp2, Zen_Esp3, Izena1, Izena2, Izena3, Abizena1, Abizena2, Abizena3, Kurtso1, Kurtso2, Kurtso3, Taldea1, Taldea2, Taldea3
- Gainera, diseinua eta inplementazioa errazten du
 - Irakurgarritasuna, eskalagarritasuna eta malgutasuna



Aurkibidea

- Gaiaren helburuak
- Motibazioa
- Erregistroak
- **Habiaratutako egiturak**



Erregistroak erregistroekin

- Erregistro baten eremuak erregistro motakoak izan daitezke

```
type T_pertsona is record  
  Identif: Integer;  
  Izena, Abizena: String(1..20);  
end record;
```

```
type T_bikotea is record  
  Pertsona1, Pertsona2: T_pertsona;  
  Helbidea: String(1..30);  
end record;
```



Erregistroak erregistroekin

```
Bikotea: T_bikotea;
```

Hedapena

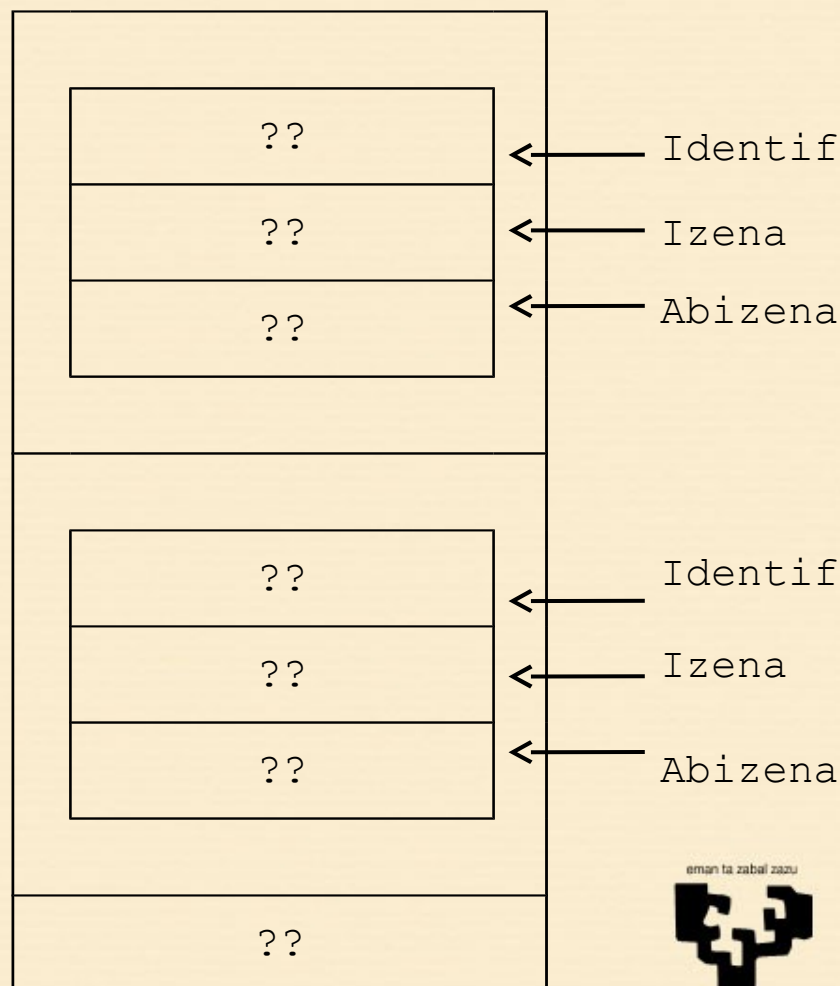
```
Bikotea.Pertsona1.Identif  
                .Izena  
                .Abizena  
    .Pertsona2.Identif  
                .Izena  
                .Abizena  
    .Helbidea
```

Pertsona1

Pertsona2

Helbidea

Bikotea



Eremuen esleipena

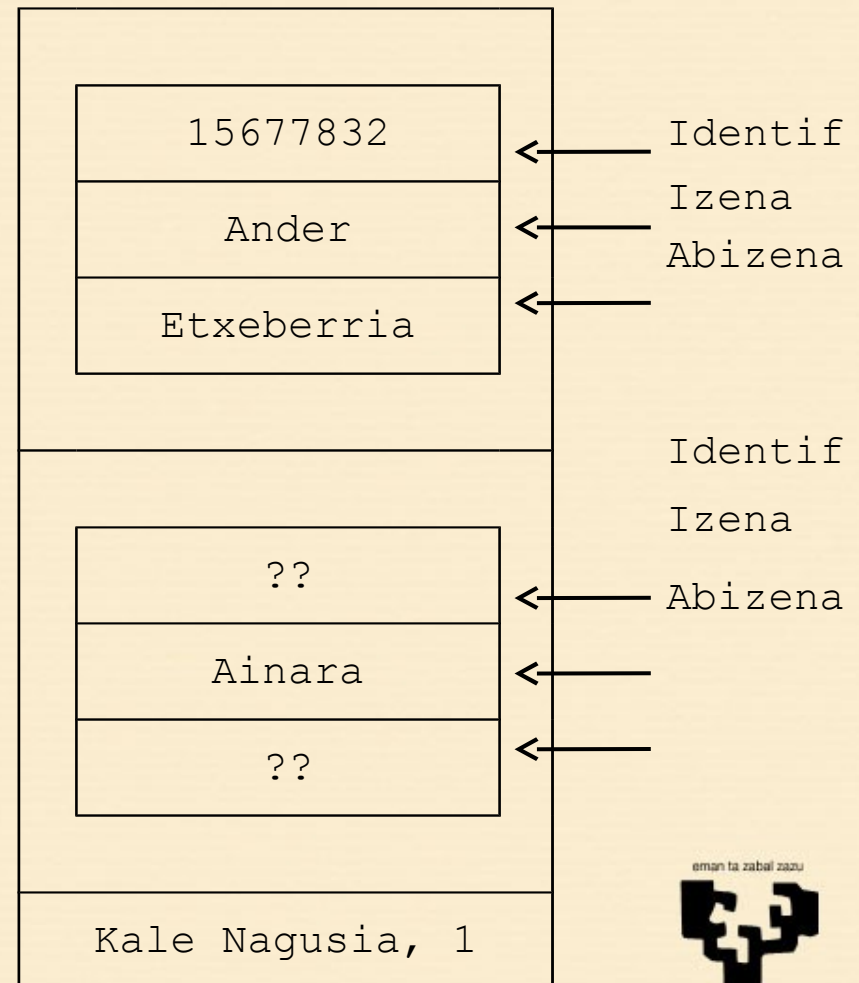
```
Bikotea.Domicilio := "Kale Nagusia, 1";  
Bikotea.Pertsonal1.Identif := 15677832;  
Bikotea.Pertsonal1.Izena := "Ander";  
Bikotea.Pertsonal1.Abizena := "Etxeberria";  
Bikotea.Pertsona2.Izena := "Ainhua";
```

Pertsonal1

Pertsona2

Helbidea

Bikotea



Erregistro motako parametroak

```
procedure irakurri (P: out T_pertsona) is
-- Aurre: Sarrerako sekuentzian (teklatutik) zenbaki bat eta
bi karaktere kate (20 karakteretakoak) pertsona
identifikatzen dutenak
-- Post: datuak P-n gorde dira
begin
    get(P.Identif);
    get(P.Izena);
    get(P.Abizena);
end irakurri;
```

```
procedure idatzi (P: in T_pertsona) is
-- Post: P-ren datuak irteera estandarrean idatzi dira
begin
    put(P.Identif);
    put(P.Izena);
    put(P.Abizena);
end idatzi;
```



Erregistro motako parametroak

```
procedure bikotea_esleitu (Bikotea: in out T_bikotea; Pos: in Integer;  
                          Pertsona: in T_pertsona) is  
-- Aurre: Pos 1 edo 2 da  
-- Post: Pertsona1 eremuak Pertsona-ren balioa dauka Pos-en balioa 1  
bada, edo Pertsona 2 eramuak dauka Po-en balioa 2 bada.  
  
begin  
  
    if (Pos=1) then  
        Bikotea.Pertsona1 := Pertsona;  
    else  
        Bikotea.Pertsona2 := Pertsona;  
    end if;  
  
end bikotea_esleitu;
```



Erregistroak eta bektoreak konbinatuz

- Erregistroak eta bektoreak konbinatu daitezke hamaika datu ezberdinetako egiturak ahalbidetuz
 - Geroz eta egitura konplexuak, orduan eta errazagoa izango da hedapenarekin lan egitea



Erregistroen arraya

```
type T_pertsona is record
  Identif: Integer;
  Izena, Abizena: String(1..20);
end record;
type T_Taula_pertsonak is array (1..5) of T_pertsona;
```

Hedapena

Pertsonak(1..5).Identif
 .Izena
 .Abizena

```
Pertsonak: T_Taula_pertsonak;  
Pertsonak(3).Izena := "Jon";
```

1	2	3	4	5
??	??	??	??	??
??	??	Jon	??	??
??	??	??	??	??

eman ta zabal zazu



Erregistroa array motako eremu batekin

```
type T_Taula_km is array (1..12) of Integer;  
type T_Korrikalari record  
  Identif: Integer;  
  Izena, Abizena: String(1..20);  
  Egindako_km: T_Taula_km;  
end record;
```

```
Korrikalari: T_Korrikalari;  
Korrikalari.Izena := "Ane";  
Korrikalari.Egindako_km(5) := 580;
```

Hedapena

Korrikalari.Identif
 .Izena
 .Abizena
 .Egindako_km(1...12)

??											
Ane											
??											
??	??	??	??	580	??	??	??	??	??	??	??



Ariketa ebatzia

- Datu egitura bat definituko da klaseko afarira joango direnen izen eta abizenak gordetzeko
- Zerrendan hutsetik abiatu beharrean, zerrenda betearekin hasiko gara, klasekide guztiak dituelarik, eta afarira etorriko ez direnak zerrendatik kenduko ditugu



Datu egitura

120 matrikulatu daude OP
ikasgaian, 01 eta 31 taldeak
kontuan hartuta

```
type Ikasle is record
  Izena: String(1..30);
  Abizena: String(1..30);
end record;
type Ikasle_Bektorea is array (1 .. 120) of Ikasle;
```

Hedapena

```
Klaseko_ikasleak(1...120).Izena
                           .Abizena
```



Soluzioaren inplementazioa

```
procedure ikasleak_kudeatu is
  Klaseko_ikasleak: Ikasle_Bektorea;
  Izena: String(1..30);
begin
  egitura_bete(Klaseko_ikasleak);
  get(Izena);
  loop exit when Izena = " ";
    ikaslea_ezabatu(Klaseko_ikasleak, Izena);
    get(Izena);
  end loop;
end ikasleak_kudeatu;
```

```
procedure egitura_bete
  (Ikasleak: out Ikasle_Bektorea) is
  Ind: Integer;
begin
    for Ind in 1..120 loop
      get(Ikasleak(Ind).Izena);
      get(Ikasleak(Ind).Abizena);
    end loop;
end egitura_bete;
```

```
procedure ikaslea_ezbatu(Eguneratzeiko_ikasleak: in out Ikasle_Bektorea;
                        Ezabatzeko_izena: in String(1..30)) is
  Ind: Integer;
begin
    Ind := posizioa(Eguneratzeiko_ikasleak, Ezabatzeko_izena);
    ezabatu(Eguneratzeiko_ikasleak, Ind);
end ikaslea_ezbatu;
```



posizioa

```
function posizioa (Ikasleak: in Ikasle_Bektorea;  
                  Ezabatzeko_izena: in String(1..30)) return Integer is  
  Indizea: Integer;  
  Aurkitua: Boolean := False;  
begin  
    Indizea:=1;  
    loop exit when Indizea > Ikasleak'last or Aurkitua;  
    if Ikasleak(Indizea).Izena = Ezabatzeko_izena then  
      Aurkitua:= True;  
    else  
      Indizea:= Indizea+1;  
    end if;  
  end loop;  
  return Indizea;  
end posizioa;
```



ezabatu

```
procedure ezabatu(Ikasleak: in out Ikasle_Bektorea;Pos: in Integer) is  
  begin
```

```
    ???????
```

```
end ezabatu;
```



**Nola ezabatuko dugu
elementua bektoretik?**

Diseinuko arazoa

- 120 ikasleren informazioa bektore batean gordetzen ari gara, eta honen tamaina beti 120koa izango da
 - Elementu bat ezabatuz gero, jada ez zaigu 120 ikasleren informazioa gordetzea interesatzen, 119rena baizik
 - Programa exekutatuz doan bitartean, geroz eta ikasle gutxiagoren informazioa gorde nahiko dugu

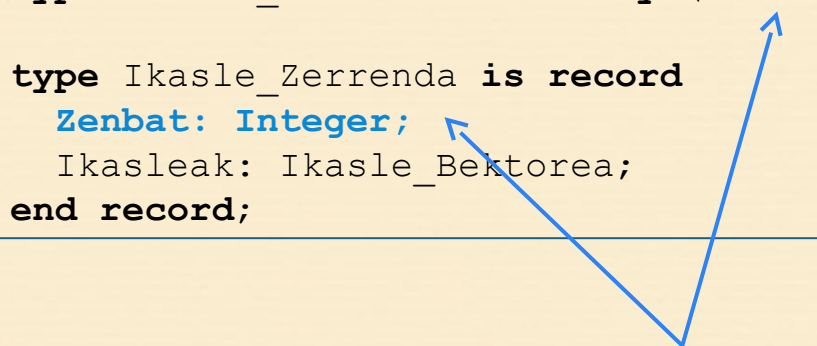
Nola konpon dezakegu
hori?

Datu egitura egokia

- Arraya erregistro baten barruan kapsulatu beharko dugu

```
type Ikasle is record
  Izena: String(1..30);
  Abizena: String(1..30);
end record;
type Ikasle_Bektorea is array (1..120) of Ikasle;

type Ikasle_Zerrenda is record
  Zenbat: Integer;
  Ikasleak: Ikasle_Bektorea;
end record;
```



Bektorean beti egongo dira 120 "ikasle".
Baina lehenengo *Zenbat* ikasleen
informazioa bakarrik interesatzen zaigu
(gainerakoak zaborra izango dira)

Hedapena

```
Klasea.Zenbat
      .Ikasleak(1...120).Izena
                        .Abizena
```

Inplementazioa

```
procedure ikasleak_kudeatu is  
    Klaseko_ikasleak: Ikasle_zerrenda;  
    Izena: String(1..30);  
begin  
    Klaseko_ikasleak.Zenbat:= 0;  
    egitura_bete(Klaseko_ikasleak);  
    get(Izena);  
    loop exit when Nombre = "          ";  
        ikaslea_ezabatu(Klaseko_ikasleak, Izena);  
        get(Izena);  
    end loop;  
end ikasleak_kudeatu;
```



Inplementazioa

```
procedure egitura_bete(IZ: out Ikasle_Zerrenda) is
  Ize, Abi: String(1..30);
begin
  get(Ize);
  get(Abi);
  loop exit when Ize = "                ";
    IZ.Zenbat := IZ.Zenbat + 1;
    IZ.Ikasleak(IZ.Zenbat).Izena:= Ize;
    IZ.Ikasleak(IZ.Zenbat).Abizena:= Abi;
  end loop;
end egitura_bete;
```

```
procedure ikaslea_ezabatu(Eguneratzeko_ikasleak: in out Ikasle_Zerrenda;
                          Ezabatzeko_Izena: in String(1..30)) is
  Ind: Integer;
begin
  Ind := posizioa(Eguneratzeko_ikasleak, Ezabatzeko_izena);
  ezabatu(Eguneratzeko_ikasleak, Ind);
end ikaslea_ezabatu;
```



Inplementazioa

```
function posicion (IZ: in Ikasle_Zerrenda;  
                  Izena: in String(1..30)) return Integer is  
    Indizea: Integer;  
    Aurkitua: Boolean := False;  
begin  
    Indizea:=1;  
    loop exit when Indizea > IZ.Zenbat or Aurkitua;  
        if IZ.Ikasleak(Indizea).Izena = Izena then  
            Aurkitua:= True;  
        else  
            Indizea:= Indizea+1;  
        end if;  
    end loop;  
    return Indizea;  
end posizioa;
```



Inplementazioa

```
procedure ezabatu(IZ: in out Ikasle_zerrenda; Pos: in Integer) is  
  begin  
    ezkerrea_mugitu(IZ, Pos);  
    IZ.Zenbat := IZ.Zenbat - 1;  
end eliminar;
```

```
procedure ezkerrera_mugitu(IZ: in out Ikasle_Zerrenda; Ind: in Integer) is  
  Indizea: Integer := Ind;  
begin  
    loop exit when Indizea = IZ.Zenbat;  
      IZ.Ikasleak(Indizea) := IZ.Ikasleak(Indizea+1);  
      Indizea := Indizea + 1;  
    end loop;  
end ezkerrera_mugitu;
```



Proposatutako ariketa 1

- Sarrera estandarrean 10 saltzaileren datuak daude. Saltzaile bakoitzeko bere identifikatzailea, izena, abizeta eta 5 zenbaki, azken 5 hilabetetan ibilitako kilometroak adierazten dituztenak

```
123 Jorge Pastor 0 48 100 500 230
600 Iñigo Balda 800 1000 0 900
2500
...
```

Proposatutako ariketa 1

- Eskatzen dena:
 - Datu horiek gordetzeko datu egiturak erazagutu (motak eta aldagaiak)
 - Funtzio bat idatzi, zeinak sarrera parametro gisa saltzaileen zerrenda jasotzen duen eta ibilitako kilometroetan altuena eta saltzailearen izena itzuliko duen
 - Zein motatakoa izango da funtzioak itzuliko duen balioa?



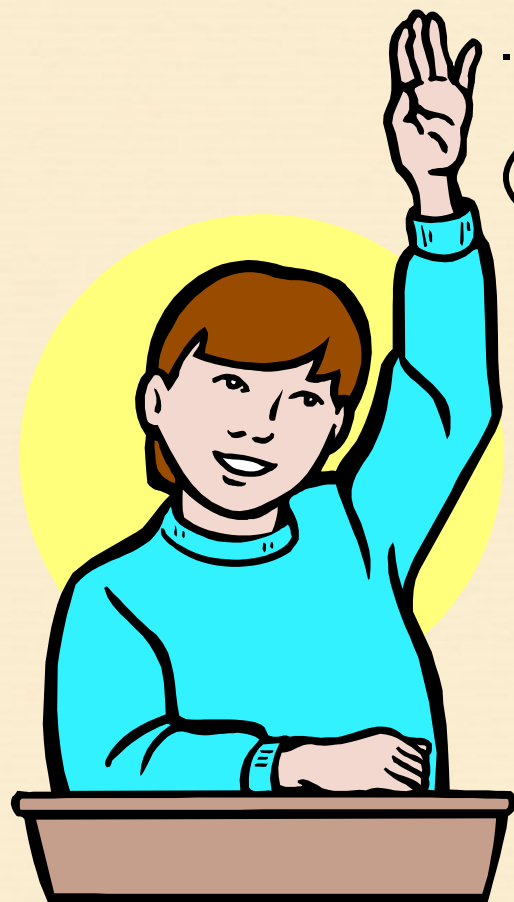
Proposatutako ariketa 2

- Ikasleen informazioa (izena eta notak) gordeko duen datu egitura definitu
 - Gehienez 100 ikasle egongo dira, eta ikasleko 15 nota, baina baliteke horrenbeste ikasleren informazioa ez izatea, edota horrenbeste nota ikasle bakoitzarentzat
 - Hau da, zabor elementuak egon daitezke

Proposatutako ariketa 2

- Eskatzen dena:
 - Azpiprograma bat idatzi zeinak egituraren amaieran ikasle bat txertatuko duen
 - Azpiprograma bat idatzi zeinak sarrera-parametro gisa egitura hori jasoko duen eta batazbesteko nota orokorra itzuliko duen





Galderarik?