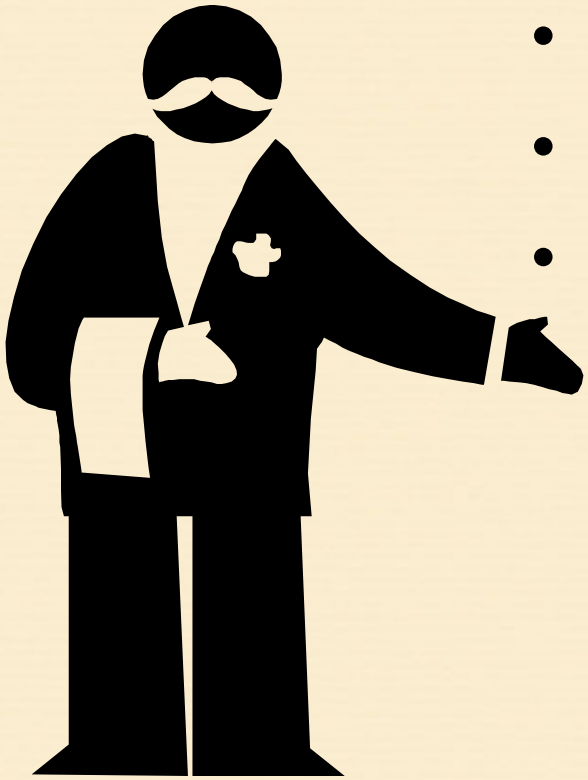
4.1 Gaia. Bektoreak (edo arrayak) eta matrizeak

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea

Aurkibidea

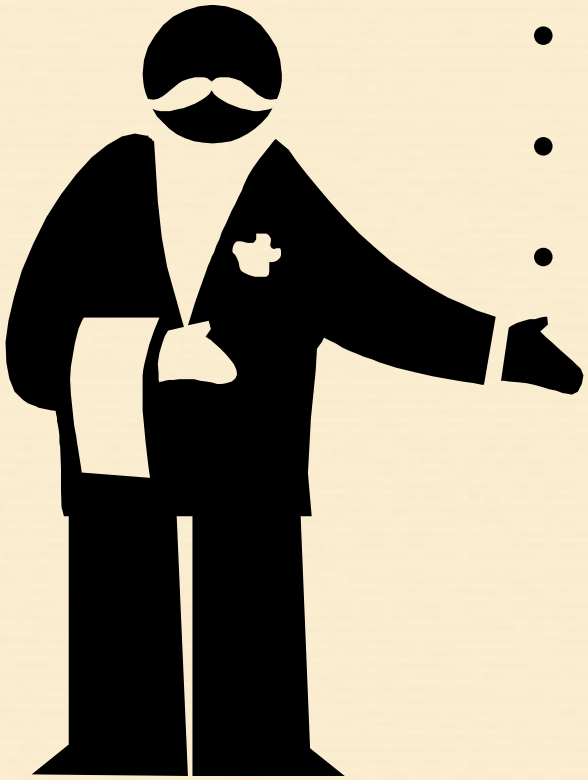
- **Gaiaren helburuak**
- Motibazioa
- Sekuentziak Adaz
- Sekuentziak Pythonez
- Matrizeak Adaz
- Matrizeak Pythonez



Gaiaren helburuak

- Elementuen sekuentzien erabilgarritasuna gogoraraztea
- Definitzea eta programetan erabiltzea
  - Dimentsio bakarreko bektoreak edo arrayak
  - Bi dimentsioetako bektoreak edo matrizeak

Aurkibidea



- Gaiaren helburuak
- **Motibazioa**
- Sekuentziak Adaz
- Sekuentziak Pythonez
- Matrizeak Adaz
- Matrizeak Pythonez

Motibazioa

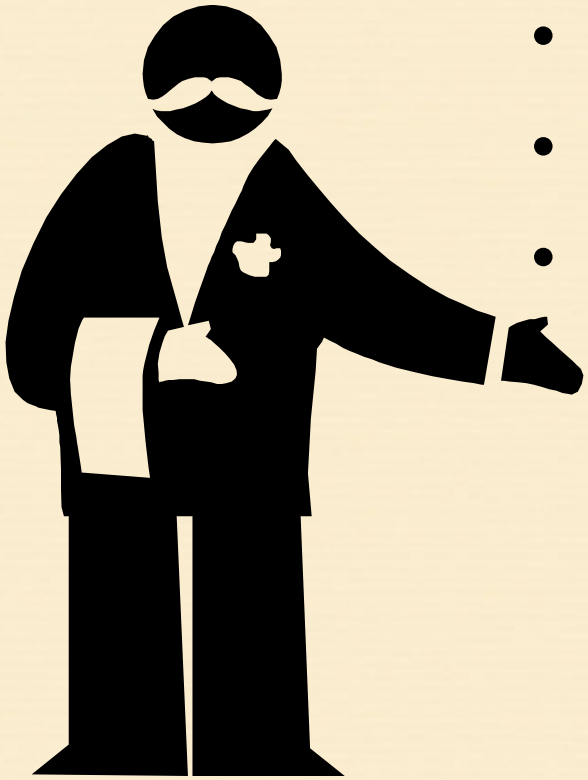
- Ataza “errazen” adibideak non oinarrizko datu motek ez duten balio
  - Teklatutik *n* zenbaki eskatu eta pantailatik inprimatu ordenatuta
  - Puntuz bukatzen den karaktere sekuentzia bat eskatuta, sekuentziako zenbakiak alderantzizko ordenean idatzi

Bektoreen erabileraren abantailak

- Mota berdineko elementuak multzokatzen dituzte, erabilera errazten
  - Zenbat aldagai behar dira OP ikasgaian matrikulatutako ikasle guztien adinak gordetzeko?
  - Zenbat kode-lerro behar dira adin guztiak 0 balioarekin hasieratzeko?
  - Zenbat parametro jasotzen ditu, matrikulatutako ikasle gazteenaren adina itzultzen duen azpiprogramak?

Aurkibidea

- Gaiaren helburuak
- Motibazioa
- **Sekuentziak Adaz**
- Sekuentziak Pythonez
- Matrizeak Adaz
- Matrizeak Pythonez



Sekuentziak Adaz

- *Array* datu motaren bidez inplementatzen dira
  - Memorian tokia erreserbatzen da, sekuentziako ondoz ondoko elementuentzat, eta elementu guztiak mota berdinekoak dira

```
NA: constant Integer := ...;

type T_adinak is array (1..NA) of Integer;
```

Sekuentziak Adaz

- Zenbat aldagai behar dira OP ikasgaian matrikulatutako ikasle guztien adinak gordetzeko?

```
Adinak: T_adinak;
```

Sekuentziak Adaz

- Zenbat kode-lerro behar dira adin guztiak 0 balioarekin hasieratzeko?
  - Soluzio posibleak:

```
for Ind in 1..NA loop
   Adinak(Ind):= 0;
end loop;
```

```
for Ind in Adinak'first..Adinak'last loop
   Adinak(Ind):= 0;
end loop;
```

```
Adinak := (others => 0);
```

Sekuentziak Adaz

- Zenbat parametro jasotzen ditu, matrikulatutako ikasle gazteenaren adina itzultzen duen azpiprogramak?
  - Soluzioa:

```
function gazteena(Adinen_Bektorea: in T_adinak) return Integer is

   ...
```

Beste adibide bat

```
package datuak is
   N: constant Integer := 12;
   type T_sekuentzia is array(1..N) of Integer
end datuak;
```

Algorítmoa

```
Elem: Integer;
Seku: 12 Integer;
hasieran_kokatu(Seku);
errepikatu kanpoan(Seku) bete arte;
   Elem <-- uneko_elementua(Seku);
   ...
  aurreratu(Seku);
amerrepikatu;
```

ADA

```
Elem, Ind: Integer;
Seku: T_sekuentzia;
Ind := 1;
loop exit when Ind > Seku'last;
   Elem := Seku(Ind);
   ...
   Ind := Ind+1;
end loop;
```

Array bat definitzeko bi modu

- Tamaina definiziotik mugatzen
  - type T_adinak is array(1..100) of Integer;
    - Adinak: T_adinak;
- Mugatu gabe (erazagupenean mugatzen da)
  - type T_adinak is array(Integer range <>) of Integer;
    - Adinak1: T_adinak(1..100);
    - Adinak2: T_adinak(1..50);

Hemendik aurrera

- Pakete batean lau datu mota definitu ditzakegu eta horrela gure programetatik erabili
  - with bektoreak; use bektoreak;

```
package bektoreak is

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

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

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

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

end bektoreak;
```

Ariketak

- Datu mota bat emanda
  - type T_ikasleen_NANak is array (1..100) of Integer;
- Azpiprogramak idatzi
  - function/procedure?? irakurri_sekuentzia (.....) ..... is
  - function/procedure?? sekuentziako_handiena (.....) ..... is
  - function/procedure?? sekuentzia_inprimatu (.....) ..... is

Ariketak

- Datu motaren definizioa emanda
  - type T_hilabeteak is array (1..12) of string(1..10);
  - edo bere baliokidea:
    - subtype T_hilabete_izena is string(1..10);
    - type T_hilabeteak is array(1..12) of T_hilabete_izena;
- Idatzi hilabete_izena_lortu azpiprograma

```
sarrera: osoko bat
Aurre: 1 <= hilabetea:balioa1 <=12
Irteera: izen bat
Post: hilabeteari dagokion izena, 1: Urtarrila, 2: Otsaila, etab.
```

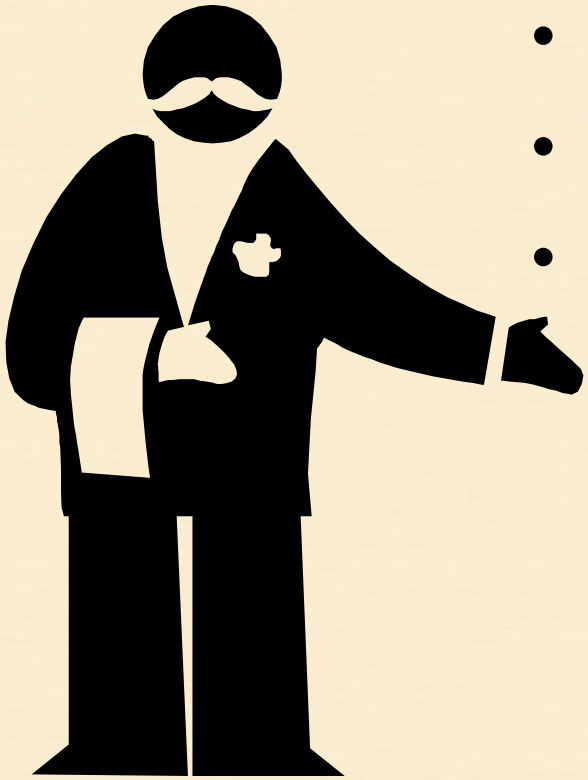
Ariketak

- Datu mota baten definizioa emanda
  - type Osokoen_Bektorea is array (1..100) of Integer;
- ezkerrera_mugitu azpiprograma idatzi, zeinak Osokoen_Bektorea jasotzen duen, eta bere elementuak posizio bat ezkerrera mugitzen dituen

1	2	3	4	5	6	7	8	9	10
13	7	1	3	9	12	23	5	8	3

1	2	3	4	5	6	7	8	9	10
7	1	3	9	12	23	5	8	3	13

Aurkibidea



- Gaiaren helburuak
- Motibazioa
- Sekuentziak Adaz
- **Sekuentziak Pythonez**
- Matrizeak Adaz
- Matrizeak Pythonez

Sekuentziak Pythonez

- Hainbat modu daude inplementatzeko. Adibidez, *list* datu mota erabiliz
  - Ez dira erazagutzen, zuzenean balioen sekuentzia bat esleitzen zaie

```
bektore1 = [1, 7, 11, 23, 4]

bektore2 = ['a', 'e', 'i', 'o', 'u']
```

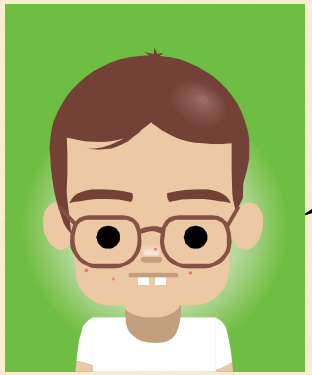
eman ta zabal zazu

Sekuentziak Pythonez

- Posizioak beti 0tik *len()*-1-era doaz
  - Aurredefinitutako *len()* azpiprogramak zerrenda baten elementu kopurua itzultzen du

```
for i in range(0,len(bektore1)):
    print (bektore1[i])
```

0	1	2	3	4
1	7	11	23	4



Gogoratu, range funtzioak hasierako elementua jasotzen duela, baina ez azkenekoa

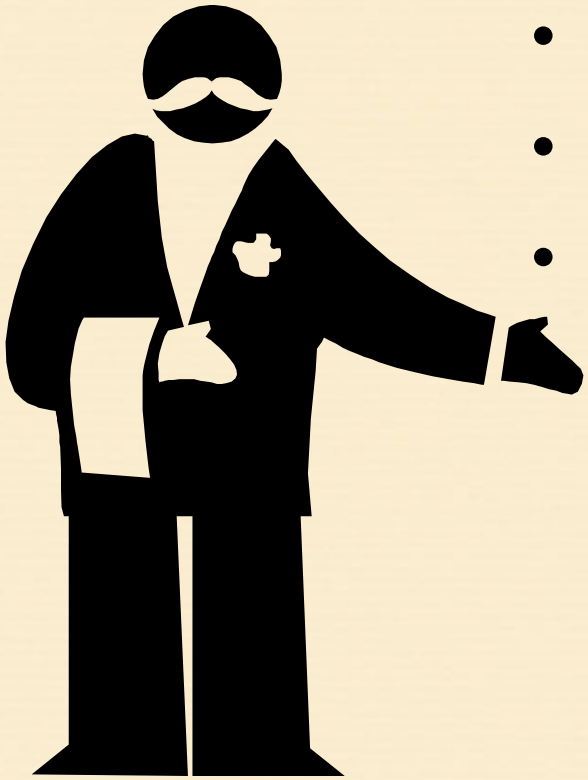
Adibidea

- Sekuentzia baten balio maximoa

```
def maximoa(bek):
    emaitza = bek[0]
    for i in range(1, len(bek)):
        if emaitza < bek[i]:
            emaitza = bek[i]
    return emaitza
```

```
def nagusia():
    zerrenda = [1, 7, 11, 23, 4]
    print(maximoa(zerrenda))
```

Aurkibidea



- Gaiaren helburuak
- Motibazioa
- Sekuentziak Adaz
- Sekuentziak Pythonez
- **Matrizeak Adaz**
- Matrizeak Pythonez

Matrizeak Adaz

- Bi dimentsiotako array moduan defini daitezke
  - Dimentsio gehiagotara orokortu daiteke
- Array-en array moduan ere defini daitezke

Bi dimentsiotako arraya

- Definiziotik bere tamaina zehazten
  - type M_adinak is array(1..4, 1..7) of Integer;
    - Adinak: M_adinak;

	1	2	3	4	5	6	7
1	17	19	18	18	19	17	19
2	21	18	17	19	18	18	17
3	18	20	18	21	18	22	18
4	20	18	18	19	19	20	21

Bi dimentsiotako arraya

- Tamaina mugatu gabe (erazagupenean mugatzen da)
  - type M_adinak is array(Integer range <>, Integer range <>) of Integer;
    - Adinak1: M_adinak(1..100, 1..30);
    - Adinak2: M_adinak(1..4, 1..7);

Arrayen arraya

- type T_adinen_errenkada is array (1..7) of Integer;
- type M_adinak is array(1..4) of T_adinen_errenkada;
    - Adinak: M_adinak;

	1	2	3	4	5	6	7
1	17	19	18	18	19	17	19
2	21	18	17	19	18	18	17
3	18	20	18	21	18	22	18
4	20	18	18	19	19	20	21

Lehen bezala

- Pakete batean lau datu mota berri defini ditzakegu, gure programetan erabili ahal izateko

```
package matrizeak is

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

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

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

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

end matrizeak;
```

Elementuen atzipena

- Bi indize erabiliko dira, bata errenkadari erreferentzia egiteko eta bestea zutabeari

```
Pantaila: Karaktereen_Matrizea(1..24, 1..80);
...
Pantaila(6,25) := 'F';
 -- 6. errenkadako eta 25 zutabeko karakterea F izango da
```

```
type T_errenkada_boolak is array (1..10) of Boolean;
type M_boolearrak is array (1..10) of T_errenkada_boolak;
...
Itsasontziak: M_boolearrak;
...
Itsasontziak(I)(J+1) := True;
-- i errenkada eta j+1 zutabeko elementua True izango da
```

Tarteak: bi dimentsiotako arraya

- Esplizituki adierazi behar da zein dimentsiori egiten dioten erreferentzia 'first eta 'last adierazpenek

```
package datuak is
  type Osokoen_Matrizea is array(Integer range <>, Integer range <>) of Integer;
end datuak;
```

```
procedure pantailan_idatzi (M: in Osokoen_Matrizea) is
begin
  for errenkada in M'first(1)..M'last(1) loop
    for zutabea in M'first(2)..M'last(2) loop
       put(M(errenkada, zutabea));
    end loop
  end loop;
end pantailan_idatzi;
```

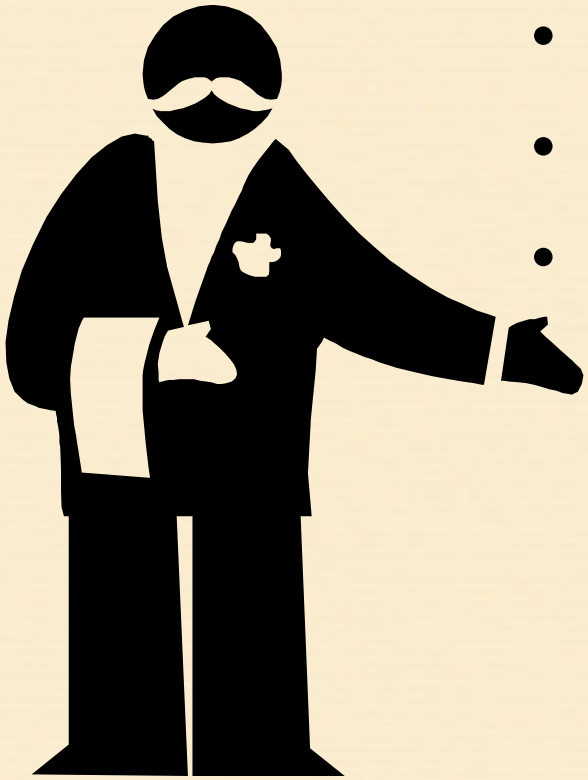
Tarteak: arrayen arraya

- Kasu honetan, 'first eta 'last dimentsio bakarreko arrayei aplikatzen zaie, beraz ez da bereizketarik egin behar

```
package datuak is
 type T_osokoen_errenkada is array (1..10) of Integer;
 type M_osokoak is array (1..10) of T_osokoen_errenkada;
end datuak;
```

```
procedure pantailan_idatzi (M: in M_osokoak) is
begin
  for errenkada in M'first..M'last loop
    for zutabe_elem in M(errenkada)'first..M(errenkada)'last loop
       put(M(errenkada)(zutabe_elem));
    end loop
  end loop;
end pantailan_idatzi;
```

Aurkibidea



- Gaiaren helburuak
- Motibazioa
- Sekuentziak Adaz
- Sekuentziak Pythonez
- Matrizeak Adaz
- **Matrizeak Pythonez**

Matrizeak Pythonez

- Zerrenden zerrenda moduan definitzen ditugu

```
zutabeak = [None]*7
M_adinak = [zutabeak]*4
...
M_adinak[0]=[17,19,18,18,19,17,19]
M_adinak[1]=[21,18,17,19,18,18,17]
M_adinak[2]=[18,20,18,21,18,22,18]
M_adinak[3]=[20,18,18,19,19,20,21]
```

	0	1	2	3	4	5	6
0	17	19	18	18	19	17	19
1	21	18	17	19	18	18	17
2	18	20	18	21	18	22	18
3	20	18	18	19	19	20	21

Adibidea

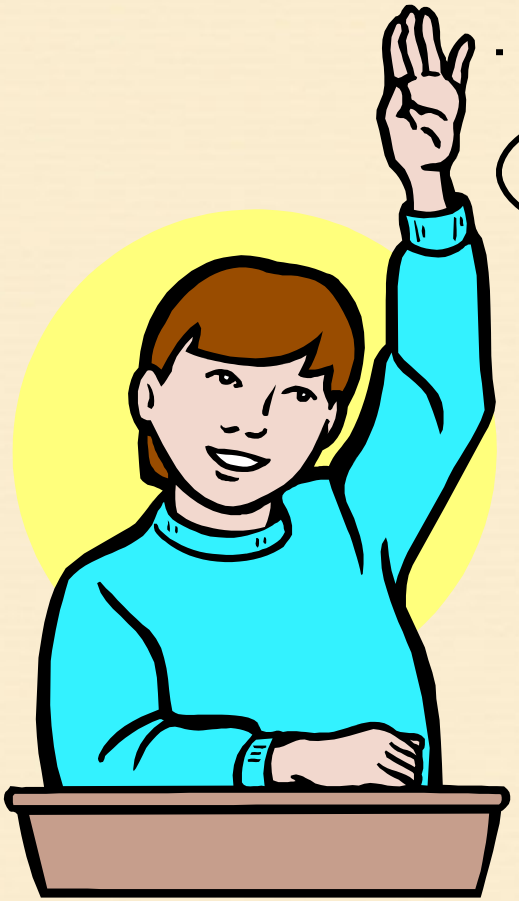
- Matrize baten elementuak inprimatu

```
def inprimatu(Matrizea):

    errenkadaKop = len(Matrizea)
    zutabeKop = len(Matrizea[0])

    for posE in range(0,errenkadaKop):
        for posK in range(0,zutabeKop):
            print(Matrizea[posE][posK], end='')
        print() #lerro saltoa
```

```
def principal():
    zutabeak=[None]*9
    M=[zutabeak]*5
    M[0]=[1,2,3,4,5,6,7,8,9]
    M[1]=[11,12,13,14,15,16,17,18,19]
    M[2]=[21,22,23,24,25,26,27,28,29]
    M[3]=[31,32,33,34,35,36,37,38,39]
    M[4]=[41,42,43,44,45,46,47,48,49]
    inprimatu(M)
```



Galderarik?