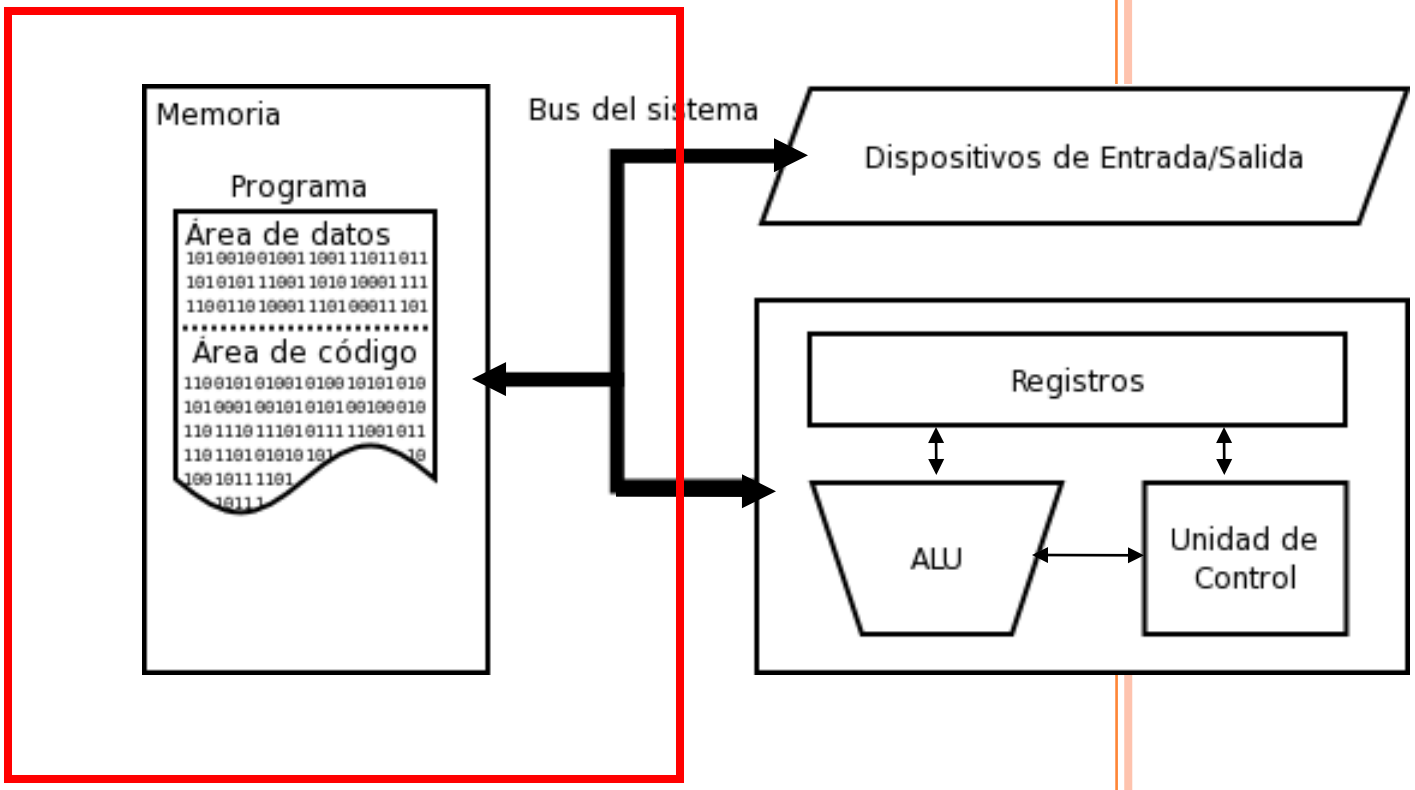
2. Gaia. Oinarrizko datuak eta eragiketak

Gogoratzen:



VON NEWMAN ARKITEKTURA

- Datuak
- Eragiketak

Datuak aldagaietan gordetzen dira

- Aldagaia: Memoria zati erreserbatu bat, helduko duguna izen (identifikadore) baten bitartez, eta datu bat gordeko duena, geroago datu horren balioa algoritmo edo programa baten bitartez aldatu ahal izateko.

- Gogoratu, aldagai baten balioa aldakorra (hortik bere izena) izaten da, algoritmoaren exekuzioan normalean balioa aldatzen dute.
- Aldagai bakoitzak memoria zati bat du esleitua.

- Aldagai baten identifikadorea, izena, aldagai horrek errepresentatzen duena deskribatu beharko luke :

  maximo, *minimo*, *gehiketa, kontadorea*

  eta ez i, j, a, *gauza, x, y, z*

Irakurri (a);
Irakurri (b);
Irakurri (c);
a < 24 **eta** b < 60 **eta** c < 60 **balira orduan** egokia
bestela okerra

Izenen hautapenarenondorioz zaila egiten zaigu algoritmo honek duen helburua jakitea
Konparatu orain hurrengo bertsioarekin:

Irakurri(orduak);
Irakurri(minutuak);
Irakurri(segunduak);
orduak < 24 **eta** minutuak < 60 **eta** segunduak < 60 **balira orduan** egokia
bestela okerra

Oinarrizko Datu Motak

Osoa (integer)	45000 -7 …	16 bit edo 32 bit
Erreala (float)	3.1416 9.2E-6 …	32 bit edo 64 bit
Boolearra (boolean)	true False	1 bit
Karakterea (character)	‘a’ ‘c’ ‘d’ ‘w’ ‘x’….	8 bit
Karaktere katea (string)	“hola” “Ander”	8*posizio kopurua

Aldagaien izenak eratzeko arau orokorrak

- Izenak eratzeko, letrak ('A', ..., 'Z', 'a', ..., 'z'), digituak ('0', ..., '9') eta '_' erabil daitezke bakarrik.
- Izenaren lehenengo karakterea ezin da digitua izan.
- Izen batek gehienez 32 karaktere izan ditzake.

Zenbaki osoak matematikan

- Infinitu zenbaki oso daude matematikan:

$$\mathbb{Z} = \{ \ldots, -4, -3, -2, -1, 0, 1, 2, 3, 4, \ldots \}$$

$$(-\infty, \infty)$$

Zenbaki osoak ordenagailu batentzat: integer

- integer (16 bit erabiliz): [-32768, 32767]

  Adibideak: 18654, -5003, 88, 107, -6

- integer (32 bit erabiliz):

[-2.147.483.648, 2.147.483.647]

Adibideak: 18654, -6, 88, 200050

Zenbaki osoak ordenagailu batentzat: integer

- Oharra: KONTUZ puntuekin!!!.

18.765 ← EZ da integer bat

18765 ← BADA integer bat

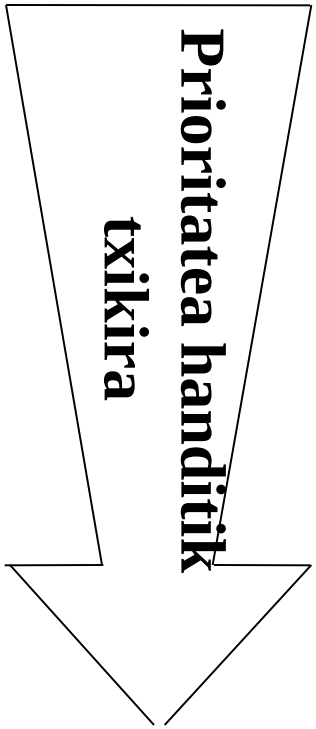
Zenbaki osoentzako eragileak

Operadore diadikoak	Operadore monadikoak
• Gehiketa: +	balio absolutoa: abs()
• Kenketa: –	zeinu aldaketa: -
• Biderketa: *	castinga: float()
• Zatiketa osoa: /	
• Zatiketaren hondarra: rem	
• Potentzia **	

(12 / 5 → 2)

KONTUZ!! Bi zenbaki osoen arteko zatiketak zenbaki oso bat bueltatuko dugu

Zenbaki osoentzako eragileak



- (zeinua)

*, /, %

+, – (gehiketa eta kenketa)

Expresioak

Expresio baten definizioa: balio bakar bat bueltatzen duen hainbat eragiketen konbinazioa.

*, / eta rem + eta - gaineko prioritatea erakusten dute.

Parentesiak erabil daitezke azpiexpresioak multzokatzeko eta prioritateak berdefinitzeko.

```
1
2+2
4 + 6/2
(4+6) / 2
( 3*5 + 12 ) rem 7
```

Zenbaki errealak ordenadorearentzat: float

- float (32 bit)

  domeinua: [(+/-)1038 y (+/-)10-38]

  Adibideak: 3.1416, 0.0027, 255.78

Zenbaki errealak ordenadorearentzat: float

- Zenbaki errealak notazio exponentzialean ere agertu daitezke:

42.054e-12 → 42.054×10^{-12}

648e7 → 648×10^{7}

Zenbaki errealak ordenadorearentzat :float

- KONTUZ BERRIRO!! Puntua erabiltzen da dezimalak markatzeko:

0,0865 ← EZ

0.0865 ← BAI

80,37 ← EZ

80.37 ← BAI

Zenbaki errealentzako eragileak **:float**

Operadore diadikoak

- Gehiketa: +
- Kenketa: –
- Biderketa: *
- Zatiketa: /
- Potentzia **

Operadore monadikoak

balio absolutoa: abs()

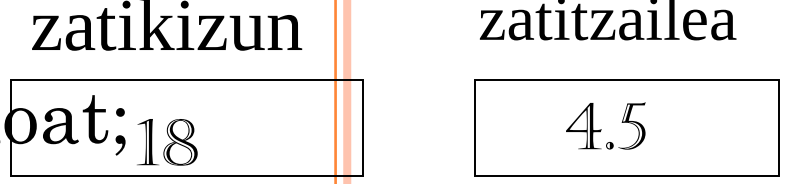
zeinu aldaketa: -

12.5 / 5.0 → 2.5

Zenbaki errealak:**float**

- Bi zenbaki osoren arteko zatiketa erreala burutzeko biak errealak izan behar dira eta emaitza jasoko duen aldagaia ere bai: :

```
zatikizun: integer;
emaitza,zatitzailea: float;
```



emaitza ← zatikizun / zatitzailea

Boolearrak (Boolean)

- boolean (1 bit)
- domeinua: {true, false} (egia y gezurra)

Eragiketak:

1) Eragiketa logikoak.

Eragigaiak eta emaitza boolearrak dira

and (eta) , or (edo), not (ez)

- Egiazko taulak

A	**B**	**A y B** **and**	**A o B** **or**	**no A** **not**
True	True			
True	False			
False	True			
False	False			

2) Eragile erlazionalak.

Mota berdineko eragileak eta emaitza boolearra. = berdin?, <= txikiago edo berdin?

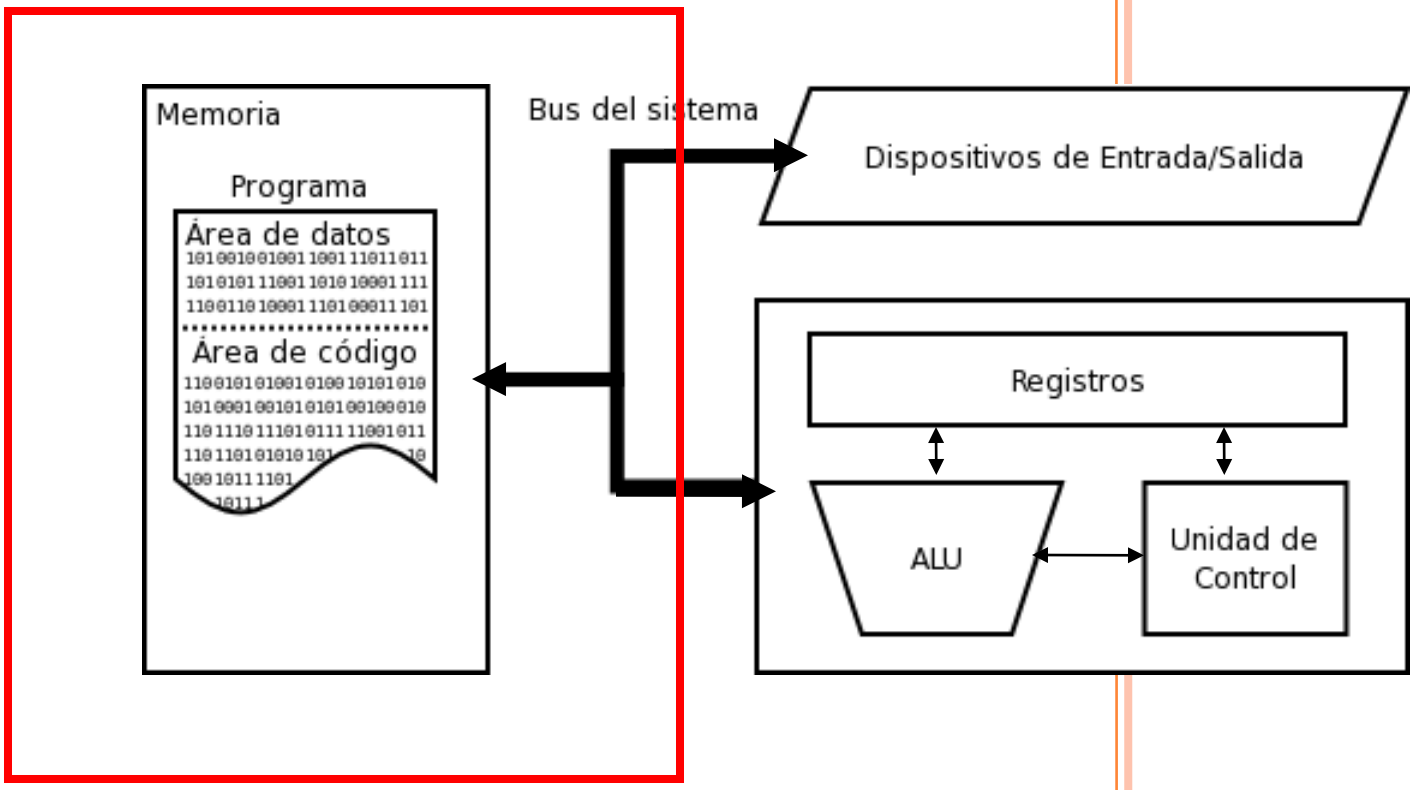
/= ezberdin?, >= handiago edo berdin?

< txikiago?, > handiago?

Karaktereaak: character

- Letra maiuskulak: '**A**', '**B**', ..., '**Z**'
- Letra minuskulak: '**a**', '**b**', ..., '**z**'
- Digituak: '**0**', '**1**', '**2**', ..., '**8**', '**9**'
- Puntuazio-zeinuak: '**.**', '**,**', '**;**', '**:**', '**?**', '**¿**', '**¡**', '**!**', '**(**', '**)**', '**{**', ...
- Karaktere bereziak: '**@**', '**#**', '**$**', '**&**', '*****', '**|**', '**%**', '**€**', ...

Gogoratu:



VON NEWMAN ARKITEKTURA

Oinarrizko eragiketak

- Oinarrizko eragiketak
- Kontrol eragiketak

Oinarrizko eragiketak pseudokodean

- Oinarrizko eragiketak:
  - esleipena : ⟵
  - idatzi( ) : Irteera estandarrean.
  - irakurri( ) : Sarrera estandarretik.

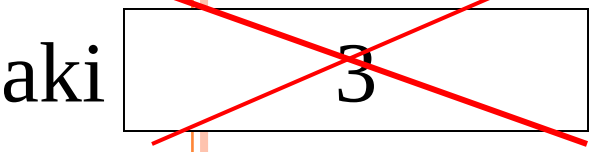
Esleipena

- Esleipenaren formalizazioa pseudokodean:

  aldagaia ⟵ expresioa;

Deklarazioa eta esleipena

zenbaki: integer;



Lerro honetan memorian 16 bit erreserbatu ditugu zenbakia deitzen den aldagai batentzat.

Aldagai honen balioa ezezaguna da momentu honetan, ez bait dakigu memorian zer zegoen momentuan.

zenbaki ⟵ 3;

esleipenaren bitartez orain balio jakin bat jarri dugu zenbaki deitzen den 16 biteko memoria zati horretan

Deklarazioa eta esleipena

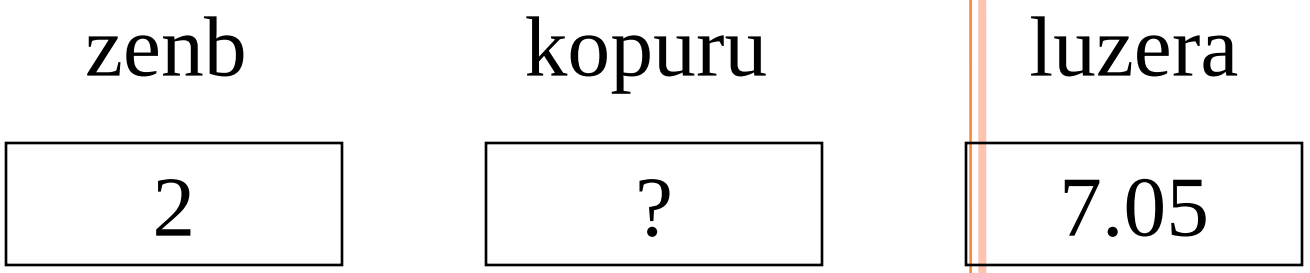
- Adibideak:

  zenb, kopuru, : integer;

  luzera: float;

  zenb ⟵ 2;

  luzera ⟵ 7.05;



Deklarazioa esleipena eta euren arteko komunztadura

- Aldagai guztiak mota batekin deklaratzen dira (integer, float, character, boolean, string ...)
- literal bakoitzak inplizituki mota bat du

Suposatu dezagun aldagai batek (zenbaki demagun) integer bezala deklaratu dela:

zenbaki: integer

Moten komunztadura erroreak:

```
zenbaki   ←——   'a‘ ERRORE!!
zenbaki   ←——   Egia ERRORE!!
zenbaki   ←——   "Comisión apertura” ERRORE!!
```

Deklarazioa esleipena eta euren arteko komunztadura

Izena string motatako aldagai bat balitz

eta **aurkituta** boolear motatakoa, hurrengo esleipenak ere okerrak lirateke:

Zenbaki ⟵ Izena

Izena ⟵ Zenbaki

Zenbaki ⟵ Zenbaki + Izena

Izena ⟵ aurkituta

Sarrera/Irteera estandarra

- Programatik “Kanpo” balioak atzitzeko erabiltzen dira (edo programatik kanpo balioak bueltatzeko)
- Adibidez teklatu bidez balioak atzitzeko edo pantailan balioak edo mezuak erakusteko.
- pseudokodean

  irakurri(zenbaki)

  idatzi(zenbaki)

```
Irakurri zenb1 eta zenb2
idatzi zenb1 “ eta “ zenb2 “-ren gehiketa
  “ zenb1 + zenb2 “ da”
idatzi “eta kenketa " zenb1 - zenb2
```

Kontrol egiturak

- Eragiketen exekuzio fluxua kontrolatzeko erabiltzen dira.
- Kontrol egiturarik ez balego eragiketak **sekuentzian** egikarituko lirateke (alegia, idatzita dauden orden berdinean):

```
  irakurri zenb1 eta zenb2
  idatzi zenb1 " eta " zenb2 "-ren gehiketa  "
zenb1 + zenb2 "    da"
  idatzi "eta kenketa " zenb1 - zenb2
```

Kontrol egiturak

Nola topatu zein den zenb1 eta zenb2 arteko handiena?

```
irakurri zenb1 eta zenb2
idatzi zenb1 " eta " zenb2 "-ren gehiketa  "
                zenb1 + zenb2 " da"
idatzi "eta kenketa " zenb1 - zenb2
```

Kontrol egiturak

Kontrol egiturak

- Operadore erlazional (>, <, =, /=, >=, <=)
- eta/edo logikoen (and, or, xor, not) bitartez sortutako baldintzetan oinarritzen dira.

Horrela, baldintza egia edo gezurra bezala ebaluatu ostean eragiketa batzuk edo besteak egikarituko dira. Hau da exekuzio fluxua bide bat edo bestea jarraituko du (gurutze bide bat bezala)

Kontrol egiturak

Adibidez: zenb1 < zenb2

Egia ala gezurra bueltatzen duen expresio boolearra da.

Orain hau jakinda planteatu dezakegu nola topatu zenb1 eta zenb2 arteko handiena:

```
zenb1, zenb2: integer;
irakurri zenb1 eta zenb2
zenb1 >= zenb2 egia_balitz
orduan
  idatzi “handiena " zenb1 “da”
bestela
  idatzi “handiena " zenb2 “da”
amaiera
```

Kontrol egiturak

Ideia hurrengoa da:

baldintza **egia_balitz**
orduan egikaritu
eragiketa multzo_1
bestela egikaritu
eragiketa multzo_2
amaiera

Baldintza betetzen denean

bakarrik kasu horretan eragiketa multzo_1 egikarituko da

Baldintza ez bada betetzen

bakarrik kasu horretan eragiketa multzo_1 egikarituko da

Kontrol egiturak

Operadore berdinak erabili daitezke ere karaktere eta stringekin, sortzen den konparaketa alfabetikoa izanik:

```
Nombre = "Aitor“ egia_balitz
'a' <= 'f‘ egia_balitz
"elefante" < "mosquito“ egia_balitz
```

Beti bezala, kasu honetan ere moten komunztadura ere aplikatzen da. Beraz ezingo genuke konparatu:

```
4 < "mosquito"
```

baina bai: `"4" < "mosquito"`

```
'4' >= 'a'
```

```
zenb1,zenb2,max: integer;
irakurri zenb1 eta zenb2
zenb1 > zenb2
egia_balitz
orduan max ← zenb1
bestela
        max ← zenb2
amaiera
idatzi "handiena" zenb1 "da"
```

```
zenb1,zenb2,max: integer;
irakurri zenb1 eta zenb2
zenb1 >zenb2 egia_balitz
orduan
        max ← zenb1
         idatzi "handiena" max "da"
 bestela
     max ← zenb2
      idatzi "handiena" max "da"
  amaiera
```

```
zenb1,zenb2,max: integer;
irakurri zenb1 eta zenb2
max ← zenb2
zenb1 >= zenb2
egia_balitz
orduan
max ← zenb1
amaiera
idatzi "handiena" max "da"
```

```
zenb1,zenb2,max: integer;
irakurri zenb1 eta zenb2
 zenb1 > zenb2
egia_balitz
orduan max ← zenb1
bestela
        max ← zenb2
amaiera
idatzi "handiena" max "da"
```

```
zenb1,zenb2,max: integer;
irakurri zenb1 eta zenb2

(zenb1 < zenb2)
egia_balitz
orduan max ← zenb2
      idatzi "handiena" max "da"
bestela
       max ← zenb2
       idatzi "handiena" max "da"
 amaiera
```

```
zenb1,zenb2,max: integer;
irakurri zenb1 eta zenb2

(zenb1 < zenb2)
egia_balitz
orduan max ← zenb2
 amaiera
idatzi "handiena" max "da"
```