Oinarrizko Programazioa

Klaseetako materiala

Programen beheranzko diseinua. Azpiprogramak: funtzioak eta prozedurak

(3. gaia)

Arantza Díaz de Ilarraza Sánchez
Kepa Sarasola Gabiola

Lengoaiak eta Sistema Informatikoak Saila
Euskal Herriko Unibertsitatea

Oinarrizko Programazioa

Gai zerrenda:

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

3. Programen beheranzko diseinua.
Azpiprogramak: funtzioak eta prozedurak.

Beheranzko diseinua. Sarrera

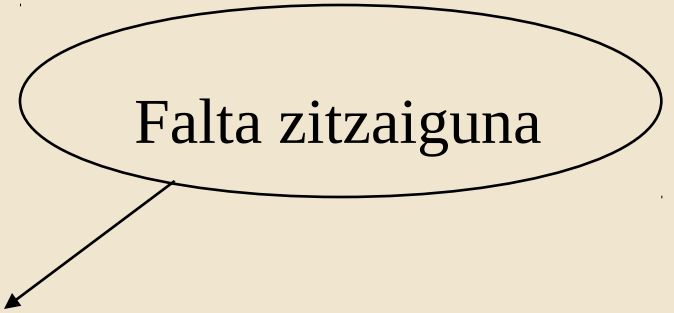
- Ez idatzi algoritmo luzeegiak. Saia zaitez **problema nagusia azpiproblema errazagotan banatzen** eta, geroago, hauek aparte definitzen
- Azpiprogramek zati askeez osatutako programa bat eraikitzeko balio dute, non azpiprograma bakoitzak xede bakun bat izango duen.
- Horrela programa luze baten eraikuntza asko errazten da azpiprograma bakoitza bereizita idatz eta azter baitaiteke.

Beheranzko diseinua. Sarrera

- Azpiprograma bat, *nola* funtzionatzen duen jakin gabe izan daiteke deitua
- Azpiprogramaren erabiltzaileak moduluak *zer* egiten duen bakarrik jakin behar du.
- Azpiprograma bat zertan erabiliko den jakin gabe irakur eta uler daiteke.
- Gai hauek bereizteari *abstrakzio* deritzo eta tresna intelektual ahaltsua da problema konplexuak ebazteko.

ALGORITMOEN ELEMENTUAK

- Objektuak
  - Informazioaren errepresentazioa.
  - Objektu sinpleak eta egituratuak
- Objektuak erabiltzeko oinarrizko aginduak
  - Datuak irakurri
  - Datuak idatzi
  - Asignazioa
- Adierazpenak
- Kontrol-egiturak
  - Baldintzazko egiturak
  - Iterazio-egiturak



- Moduluak
  - Azpiprogramak: Funtzioak eta prozedurak

Bi eratako azpiprogramak

Funtzioa

- Emaitza bat kalkulatzeko
- Balio modura erabiltzen da adierazpen batean
- Adibidez: *Pred* zenbaki oso baten aurrekoa itzultzeko

N: = *Pred(8) - 2*

Prozedura

- Emaitza bat baino gehiago kalkulatzekoa
  aldagaien balioak aldatzeko, irakurri eta idazteko
- Agindu modura erabiltzen da, algoritmo edo modulu batean
- Adibidez: *Idatzi_Osokoa* osoko bat idazteko prozedura

Idatzi_Osokoa(N–4)

Azpiprogramen parametroak

- Parametroen bitartez datuak pasatzen zaizkio azpiprogramari eta emaitzak jasotzen dira azpiprogramatik
- Honela azpiprogramaren definizioa orokorragoa da, eragin berdina lortu ahal izango da datu edo aldagai desberdinekin

Moduluak. Azpiprogramak

- Moduluak algoritmoak bezala definitzen dira
- Agindu-multzo batekin (gorputza)
- Moduluari deitzen zaionean gorputza egikaritzen da; ondoren algoritmoaren exekuzioak deiaren ondoko puntutik jarraituko du.
- Moduluko **barruko aldagaiak eta konstanteak** gorputza baino lehenago aipatzea hobe da.

  Horrela, objektu horiek modulu horretatik kanpo ezin erabil daitezkeela adierazi nahi dugu.

Azpiprograma estandarrak Programategiak

- Programazio-lengoaiek oso erabilgarriak diren azpiprograma edo modulu estandarrak eskaintzen dituzte.
- Adibidez:
  funtzio matematikoak (sinua, kosinua, erro karratua, ...)
- Esfortzu-bikoizketa ekiditen da
- Programatzaile berriak ere adituek eginiko algoritmo sofistikatuak erabili ahal izango du.

Ada programategiak

- http://en.wikibooks.org/wiki/Ada_Programming/Libraries
  Ada Programming/Libraries (Ada lengoaiaren programategiak):
  - GNAT konpiladorearekin datozenak:
    - Standard, Ada, Interfaces, System, GNAT
  - Beste batzuk:
    - Multi Purpose, Container Libraries, GUI Libraries, Distributed Objects, Database, Web Programming, Input/Output

Funtzioak. Adibidea

Karaktere bat hartu eta letra bat den ala ez aztertzen duen *Alfabetikoa* izeneko funtzio bat.

Definizioa:

```
funtzio Alfabetikoa (Kar : Karaketere)
  itzuli Boolearra
hasiera
    baldin ((Kar >= 'A') eta (Kar <= 'Z'))
            edo  ((Kar >= 'a') eta (Kar <= 'z'))
    orduan itzuli egiazkoa
    bestela itzuli faltsua
amaia
```

Erabilera:

```
Irakurri_Karakterea (Iniziala)
baldin Alfabetikoa(Iniziala) orduan ...
```

Funtzioak. Adibidea

Bi osoko positiboen zatitzaile komunetatik handiena kalkulatzen duen ZKH izeneko funtzioa

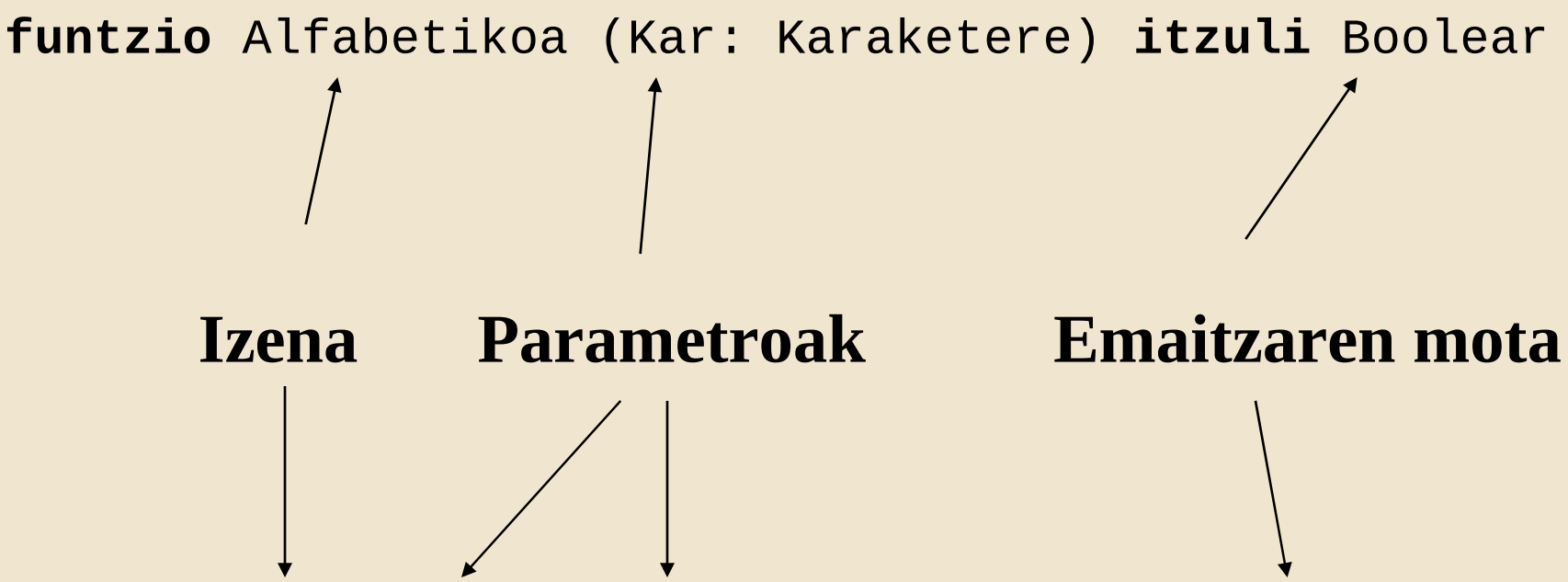
Definizioa:

```
funtzio ZKH (Zenb1, Zenb2 : Osoko) itzuli Osoko
       M, N, R        : Osoko;
   hasiera
       M := Zenb1
       N := Zenb2
       R := M mod N
       bitartean R/=0 egin
           M := N
           N := R
           R := M mod N
       ambitartean
       itzuli N
   amaia
```

Erabilera:

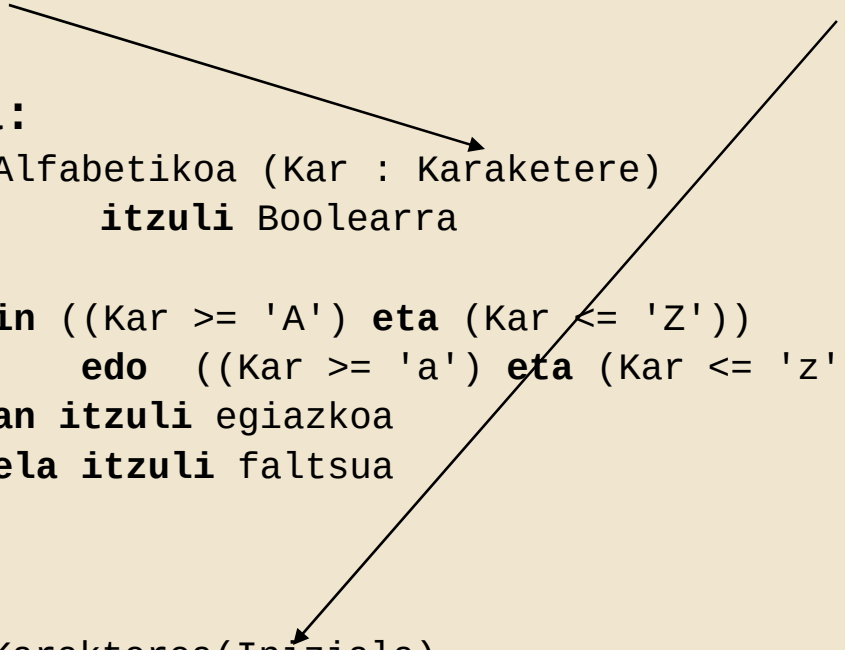
Y := 1+ ZKH(N+1, 72)

Funtzioak. Definizioa



Funtzioak. Definizioa

Parametro formala **Parametro erreala**



Definizioa:

```
funtzio Alfabetikoa (Kar : Karaketere)
             itzuli Boolearra
hasiera
    baldin ((Kar >= 'A') eta (Kar <= 'Z'))
            edo  ((Kar >= 'a') eta (Kar <= 'z'))
    orduan itzuli egiazkoa
    bestela itzuli faltsua
amaia
```

Erabilera:

```
Irakurri_Karakterea(Iniziala)
baldin Alfabetikoa(Iniziala) orduan ...
```

Funtzioak. Definizioa

```
funtzio ZKH (Zenb1, Zenb2 : Osoko) itzuli Osoko
   M, N, R: Osoko;
hasiera
        M := Zenb1
        N := Zenb2
        R := M mod N
        bitartean R/=0 egin
            M := N
            N := R
            R := M mod N
        ambitartean
        itzuli N
amaia
```

Barruko aldagaiak

Gorputza.
Emaitza kalkulatzeko agindu-multzoa

itzuli agindua
- Emaitzaren balioa zehazteko (adierazpen bat)
- Exekuzioa bukatu

Funtzioak. Definizioa

- Funtzio-gorputza *M*, *N* eta *R* aldagaiak aurkezten dituen erazagupenarekin hasten da funtzio-gorputzaren barnean aldagai moduan erabiliko direnak dira horiek.
- Funtzioaren emaitza, algoritmo euklidearra inplementatzeko *M*, *N* eta *R* erabiltzen dituen begizta baten bidez kalkulatzen da.
- Halako batean, *itzuli* sententzia egikarituko da, *M*, *N*-ren multiplo zehatza denean, funtzioaren emaitza gisa *N*-ren azken balioa itzuliz

Funtzioak. Definizioa

- Funtzio-gorputz batean *itzuli* agindu bat edo gehiago egon daitezke.
- *itzuli* agindu bakoitzak funtzioaren emaitzaren motako adierazpen bat izan behar du.
- Beharrezkoa da *itzuli* agindu hauetako bat momenturen batean egikaritzea.
- *Itzuli*-ren ondoren dagoen adierazpena ebaluatzen da funtzioaren emaitza zehazteko, eta funtzio-gorputzaren egikaritzapena bukatu egiten da.
- Balioak emaitzaren motakoa behar du izan.

Funtzioak. Definizioa

- *Zenb1* eta *Zenb2* parametro formalak
  - *ez* dira aldagai modura erabili behar funtzio barruan
  - konstanteak baizik;
  - ezin dira eguneratu.
  - *ZKH* funtzio-gorputzak behar duenean egunera daitezkeen aldagai lokaletan kopiatzen ditu bi zenbakiak.
- *M*, *N* eta *R* barruko aldagaiak funtzio-gorputzaren barnean aldagai moduan erabiliko dira.
  - Funtzio horretatik kanpo ezin dira erabili

Bi eratako azpiprogramak

Funtzioa

- Emaitza bat kalkulatzeko
- Balio modura erabiltzen da adierazpen batean
- Adibidez: *Pred* zenbaki oso baten aurrekoa itzultzeko

N: = *Pred(8) - 2*

Prozedura

- Emaitza bat baino gehiago kalkulatzekoa
  aldagaien balioak aldatzeko, irakurri eta idazteko
- Agindu modura erabiltzen da, algoritmo edo modulu batean
- Adibidez: *Idatzi_Osokoa* osoko bat idazteko prozedura

Idatzi_Osokoa(N–4)

Prozedurak. Adibidea

Zuriune-kopuru jakin bat idatzi behar duen prozedura.

Definizioa:

```
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : datu Natural)
   hasiera
       egin Kont guztietarako 1 tik  Zuriune_Kop  raino
           Idatzi_Karakterea (' ')
       amguztietarako
   amaia
```

Erabilera:

```
Zuriuneak_Idatzi (10)
Zuriuneak_Idatzi (N-1)
```

Prozedurak. Definizioa

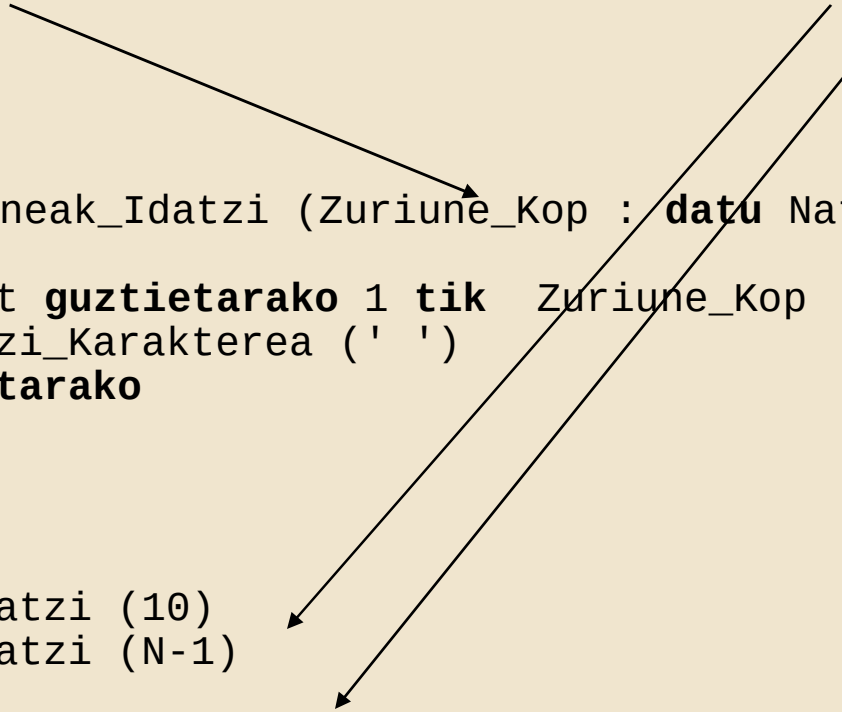
```
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : datu Natural)
```

Izena **Parametro-espezifikazioa**

```
algoritmo Zatiketa_Moztua (M, N: datu Osoko;
                           Zatidura,
                           Hondarra: emaitza Osoko)
```

Prozedurak. Adibidea

Parametro formala **Parametro erreala.**



Definizioa:

```
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : datu Natural)
   hasiera
       egin Kont guztietarako 1 tik  Zuriune_Kop  raino
           Idatzi_Karakterea (' ')
       amguztietarako
   amaia
```

Erabilera:

```
   Zuriuneak_Idatzi (10)
   Zuriuneak_Idatzi (N-1)
```

Prozedurak. Definizioa

```
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : datu Natural)

hasiera
   egin Kont guztietarako 1 tik  Zuriune_Kop  raino
      Idatzi_Karakterea (' ')
   amguztietarako
amaia
```

Gorputza

Emaitza kalkulatzeko agindu-multzoa

itzuli agindurik ez dago

Barruko objektuak desagertzen dira

- Azpiprograma bat exekutatu ondoren, azpiprograma barruan erazagututako aldagai eta objektu formal guztiak desagertzen dira.

- Horrela, hauek betetzen zuten memoria libre uzten da beste xede batzuetarako

Azpiprogramen parametroak

- Hiru parametro klase daude:
  - sarrerakoak
  - irteerakoak
  - sarrera-irteerakoak
- Diferentzia, datuak atera ala sartzeko erabiliko diren zehaztean datza

Sarrera-parametroak

- Azpiprogramari balioak emateko erabiltzen dira
- Sarrera-parametro formal bakoitzak, dagokion parametro errealetik hartzen du bere balioa
- *Konstante lokal* gisa jokatzen dute gorputza egikaritzen denean
- Parametro errealak parametro formalaren mota berekoa behar du izan
- Parametro-espezifikazioan bi puntuen ondoren *datu* hitza idazten da. (hitz berezi hori aukerakoa da) Hauek baliokideak dira:

```
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : Osoko)
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : datu Osoko)
```

- Parametro errealak adierazpen baten bidez erabiliko dira. Adibidez:
  - `Zuriuneak_Idatzi (7)`
  - `Zuriuneak_Idatzi (2*N+4)`

Irteera-parametroak

- Prozeduretan kalkulatzen diren balioak itzultzeko erabiltzen dira
- Parametro-espezifikazioan, bi puntuen ondoren *emaitza* hitza idatziz zehazten dira irteera-parametro gisa.
- Irteera-parametro formal bakoitzak barruko *aldagai* baten gisa jokatzen du. Prozedura-gorputzak balioa emango dio aldagai honi. Azpiprogramaren exekuzioa bukatutakoan, parametro formalak duen balioa parametro errealari pasako zaio
- Sarrera-irteera-parametroekin bezala, parametro errealak parametro formalaren mota bereko aldagaia behar du izan.

Irteera-parametroak

- Horrelako parametroekin, emaitza bat baino gehiago duten azpiprogramak idatz ditzakegu (funtzioek balio bakarra itzultzen dute)

```
algoritmo Zatiketa_Moztua
             (Zatikizun, Zatitzaile: datu Osoko
              Zatidura, Hondarra: emaitza Osoko)
hasiera
   Zatidura := Zatikizun / Zatitzaile
   Hondarra := Zatikizun mod Zatitzaile
amaia
```

Zatiketa_Moztua prozedurak bi datu jasotzen ditu eta bi emaitza itzuliko ditu.

- Parametro erreala aldagai bat izan beharko da. Adibidez:

```
Zatiketa_Moztua (107, 7, N1, N2)
Zatiketa_Moztua (107*33, 7, Z, H)
```

- Parametroak aldagaiaren mota bera izan behar du

Sarrera-irteera parametroak

- Prozedura batek parametro erreal moduan pasatutako aldagai bat *eguneratzeko erabiltzen dira*
- Parametro errealaren balioa prozedurari pasatu, eguneratu eta, hau egin ondoren, balioa prozeduratik kanpo bueltatzea nahiko dugu.
- Parametro-espezifikazioan, bi puntuen ondoren *datu-emaitza* hitza idatziz zehazten da hau.

```
algoritmo Gehitu (Kont : datu emaitza Osokoa)
hasiera
        Kont := Kont + 1;
amaia Gehitu;
```

- Horrela dei diezaiokegu prozedura honi:

```
Gehitu (N)
```

Sarrera-irteera parametroak

- Sarrera-irteera erako parametro formal bakoitzak prozedurako *barruko aldagai* baten moduan jokatzen du.
- Prozedura-gorputzean sartzean, aldagai lokalaren balioa berari dagokion parametro errealaren balioa da. Prozedura-gorputza egikaritzen ari den bitartean honi esleitzen zaion edozein balio, dagokion parametro errealari ere pasatuko zaio.
- Irteera-parametroekin bezala, parametro errealak parametro formalaren mota bereko aldagaia behar du izan.

Funtzioen parametroak

- Funtzioek sarrera-parametroak bakarrik eduki ditzakete
- Honek arrazoi on bat du:
  - irteera- edo sarrera-irteera erako parametroa duen funtzio batek, deitzen zaionean, bere barrukoa ez den aldagai baten balioa alda dezake.
- Horrelako fenomenoari *albo-ondorio* deritzo.
- Funtzioetan, albo-ondorio hauek ez dira gomendagarriak
- Funtzioak balio bat itzuli behar du, ... eta kito!
  hori kalkulatzean besterik aldatzen badu algoritmoen irakurgarritasuna zailtzen dute-eta .

Parametro formalak eta errealak

- Azpiprograma-dei batean, parametro formal bakoitzeko parametro erreal bat egon behar da.
- Parametro errealak posizioaren arabera parekatuko dira parametro formalekin.

```
Zatiketa_Moztua ( 98, 4, Z, H)
Zatiketa_Moztua ( N, I-1, Z, H)
```

- Edo ordena diferente izan daiteke baina orduan parametro formal bakoitzari zein parametro erreal dagokion zehaztu behar da

```
Zatiketa_Moztua (Zatikizun => 98, Zatitzaile=> 4,
                 Hondarra => H, Zatidura=> Z)
Zatiketa_Moztua (Hondarra => H, Zatidura=> Z,
                 Zatikizun => N, Zatitzaile=> I-1)
    )
```

Azpiprogramen zehaztapena: aurrebaldintza eta postbaldintza

- Zehaztapena oso inportantea da azpiprogramen definizioetan.
- Azpiprogramaren erabiltzaileak moduluak *zer* egiten duen bakarrik jakin behar du.

  Nahikoa izango zaio azpiprogramaren burukoa eta zehaztapena
- Zehaztapenean ez ditugu berriro errepikatzen zeintzuk diren parametroak eta beren motak.

  Bakarrik zehazten da zeintzuk diren betetzen dituzten propietateak

Adibidea

```
funtzio Alfabetikoa (Kar : Karaketere) itzuli Boolearra
    --Aurrebaldintza:
    --Postbaldintza: itzultzen du egiazkoa Kar letra bada
                     bestela faltsua
```

```
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : Osoko)
    --Aurrebaldintza: Zuriune_Kop >= 0
    --Postbaldintza: idatzi dira Zuriune_Kop zuriune
```

Aldagaien esparrua eta ikusgarritasuna

- Aldagai baten **esparrua**: non erabil daitekeen aldagai hori algoritmo(edo programa) multzo batean.
- Aldagai bat bere esparruko puntuetan **ikusgarri** dela esaten dugu
- Hasteko, gure algoritmoetan **aldagaiaren esparrua azpiprogramarena da**, alegia, aldagaia bere barruan erazagututa duen azpiprogramarena
- Geroago esparruaren definizio osatuago bat ikusiko dugu

Programazio-estiloa

- Azpiprogramek izenak ematen dizkiete programen zatiei. Azpiprogramentzako (eta objektu eta datu-motak moduko hainbat entitateentzako) **izen zentzudunak** aukeratuz gero, programa "testu" gisa irakur daiteke.
- Azpiprogramarentzat izen egokia aukeratzea errazagoa da, moduluak **xede bakun eta sinpleren bat** baldin badu.
- Azpiprograma bati **toki desberdin askotatik dei dakioke**, nahi izanez gero parametro erreal desberdinekin. Honek programa-testuaren bikoizte aspergarria (eta errore-sortzailea) ekiditen du.

Programazio-estiloa

- Erraza da azpiprograma-gorputzaren bertsio bat beste batekin ordezkatzea. Azpiprogramaren *erazagupena* aldatzen ez den bitartean, programaren gainerako zatietan egiten diren deiak ez dira aldatu beharko.
- Iruzkinak lasai erabil itzazu programaren funtzioa adierazi eta zati bakoitzaren funtzionamendua esplikatzeko.
  - Iruzkinak adan: "--" ondoren lerro bukaeraraino
- Identifikadore ahalik eta deskriptiboenak aukera itzazu.
  - Erabil ezazu arau bat maiuskulak edo minuskulak erabiltzeko:
    - Lehen letra larriz eta besteak xehez (`Adibide_Bat`)
    - Denak larriz (`ADIBIDE_BAT`)
    - Denak xehez (`adibide_bat`)

  ...baina beti era berean!

Objektuei orientatutako programazioa

- Azpiprogramak antolatzeko teknika bat

  "Ezaugarri eta izaera propioak dituzten eta beren artean komunika daitezkeen hainbat azpiprogramaz (objektuz) osatutako programak egiteko teknika."(http://zthiztegia.elhuyar.org/)

- Azpiprogramak eurek erabiltzen dituzten objektuen arabera multzokatzen dira.
  - Ada: paketetan (*package*)
  - Java, Python, C++: klasetan (*class*)

Objektuei orientatutako programazioa Adan

- Pakete berri bat definitu behar da **datu-mota berri bat** sortzeko.
  - *Datu-motaren **balioak** (osagaiak eta beren arteko antolaketa)*
  - *Datu-motaren **eragiketak***
- Bi fitxategi definitu behar dira:
  - **Espezifikazioa (.*ads*)**: eragiketen zehaztapenekin (datu-motaren erabiltzaileak hori bakarrik ikusiko du)
  - **Inplementazioa (.*adb*)**: eragiketa guzti horien inplementazioa (bakoitza azpiprograma bat)