



# Oinarrizko Programazioa

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



# Edukiak

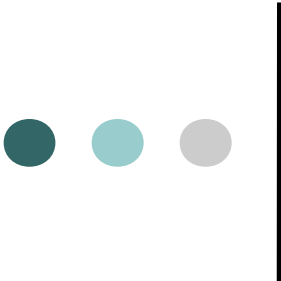
1. Sarrera
2. Programazioko oinarrizko kontzeptuak
3. **Programen beheranzko diseinua.**  
**Azpiprogramak: funtzioak eta prozedurak.**
4. Oinarrizko datu-egiturak
5. Programazio-lengoaiei erabilera
6. Aplikazio-adibideak



# 3. Programen beheranzko diseinua.

## Azpiprogramak: funtzioak eta prozedurak.

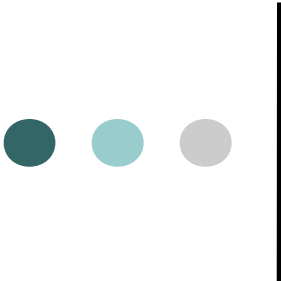
1. Sarrera.
2. Azpiprogramak: funtzioak eta prozedurak.
3. Azpiprogramen parametroak:
  - Sarrera-parametroak, irteera-parametroak eta sarrera-irteerakoak
  - Parametro formalak eta parametro errealak
4. Azpiprogramen zehaztapena:
  - Aurrebaldintza eta postbaldintza
5. Aldagaien eta parametroen esparrua eta ikusgarritasuna.
6. Azpiprogramak eta programazio-estiloa.



# 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 eraikitzeke 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 (II)

- 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 kontzeptu 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

Falta zitzaiguna



# Bi eratako azpiprogramak

## Funtzioak

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

$N := Pred(8) - 2$

## Prozedurak

- Zertarako
  - Eraitza bat baino gehiago kalkulatzeko
  - Aldagaien balioak aldatzeko, irakurtzeko, idazteko...
- **Agindu modura** erabiltzen da, algoritmo edo modulu batean
- Adibidez: *Idatzi\_Osoa* zenbaki oso bat idazteko prozedura

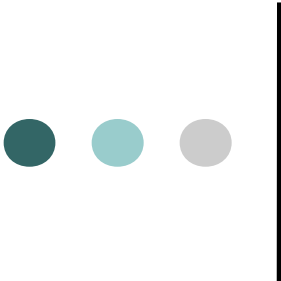
*Idatzi\_Osoa(N-4)*



# 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

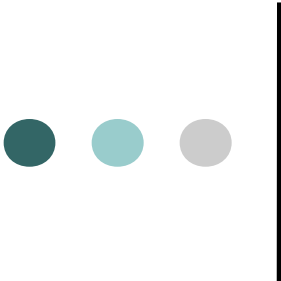




# Moduluak.

## 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 definitzen dira.
  - Horrela adierazten dugu objektu horiek ezin daitezkeela erabili modulu horretatik kanpo.



# Azpiprograma estandarrak. Programategiak

- Programazio-lengoaiek oso erabilgarriak diren azpiprograma edo modulu estandarrak eskaintzen dituzte.
  - Adibidez: funtzio matematikoak (sinua, kosinua, erro karratua, ...)
- Horrela,
  - programazio-lana errazten da,
  - programatzaile berriak ere adituek eginiko algoritmo sofistikatuak erabili ahal izango ditu.



# Ada programategiak

- [http://en.wikibooks.org/wiki/Ada\\_Programming/Libraries](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*

# Bi eratako azpiprogramak

## Funtzioak

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

$N := Pred(8) - 2$

## Prozedurak

- Zertarako
  - Eraitza bat baino gehiago kalkulatzeko
  - Aldagaien balioak aldatzeko, irakurtzeko, idazteko...
- **Agindu modura** erabiltzen da, algoritmo edo modulu batean
- Adibidez: *Idatzi\_Osoa* zenbaki oso bat idazteko prozedura

*Idatzi\_Osoa(N-4)*



# Funtzioak. Adibidea

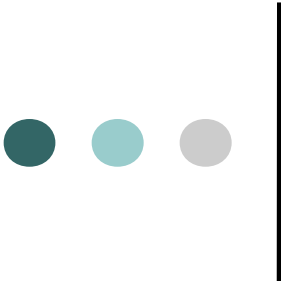
Karaktere bat hartu eta letra bat den ala ez aztertzen duen *Alfabetikoa* izeneko funtzioa:

## Definizioa:

```
funtzio Alfabetikoa (Kar : Karaktere) itzuli Boolear  
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 (II)

Bi zenbaki oso positiboen zatitzaile komunetatik handiena kalkulatzeko duen ZKH izeneko funtzioa:

**Definizioa:**

```
funtzio ZKH (Zenb1, Zenb2 : Oso) itzuli Oso
    M, N, R: Oso;
    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. Egitura

**funtzio** Alfabetikoa (Kar: Karaktere) **itzuli** Boolear

**Izena**

**Parametroak**

**Emaitzaren mota**

**funtzio** ZKH (Zenb1, Zenb2: Oso) **itzuli** Oso



# Funtzioak. Egitura (II)

Parametro formala

Parametro erreal

**Definizioa:**

```
funtzio Alfabetikoa (Kar : Karaktere)
  itzuli Boolear
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. Egitura (III)

## Gorputza:

Eraitza kalkulatzeko  
agindu-multzoa

```
funtzio ZKH (Zenb1, Zenb2: Oso) itzuli Oso  
    M, N, R: Oso;
```

← Barruko aldagaiak

**hasiera**

```
    M := Zenb1  
    N := Zenb2  
    R := M mod N  
    bitartean R/=0 egin
```

```
        M := N  
        N := R  
        R := M mod N
```

```
    amaitartean  
    itzuli N
```

**amaia**

← *itzuli* agindua

- Derrigorrez jarri beharrekoa
- Eraitza zein den zehazteko (adierazpen bat)
- Exekuzioa bukatu egiten da



# Funtzioak. Egitura (IV)

- Aurreko adibidearen harira:
  - Funtzio-gorputza  $M$ ,  $N$  eta  $R$  aldagaiak aurkezten dituen erazagupenarekin hasten da.
    - funtzio-gorputzaren barnean aldagai moduan erabiliko direnak dira horiek.
  - Funtzioak algoritmo euklidearra inplementatzen du.
    - $M$ ,  $N$  eta  $R$  erabiltzen dituen iterazio baten bidez kalkulatu da  $ZKH$ .
  - Halako batean, *itzuli* sententzia egikaritu da:
    - Horrenbestez,  $N$ ren azken balioa emaitza gisa itzuli, eta funtzioaren exekuzioa amaitu egiten da.



# Funtzioak. Egitura (V)

- Funtzio-gorputz batean *itzuli* agindu bat edo gehiago egon daitezke (bat gutxienez).
- *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.



# Funtzioak. Egitura (VI)

- *Zenb1* eta *Zenb2* parametro formalak
  - ez dira aldagai modura erabili behar funtzio barruan, konstante gisa baizik
    - Horrenbestez, 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.
  - Baina, funtzio horretatik kanpo ezin dira erabili

# Bi eratako azpiprogramak

## Funtzioak

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

$N := Pred(8) - 2$

## Prozedurak



- Zertarako
  - Eraitza bat baino gehiago kalkulatzeko
  - Aldagaien balioak aldatzeko, irakurtzeko, idazteko...
- **Agindu modura** erabiltzen da, algoritmo edo modulu batean
- Adibidez: *Idatzi\_Osoa* zenbaki oso bat idazteko prozedura

*Idatzi\_Osoa(N-4)*



# Prozedurak. Adibidea

Zuriune kopuru jakin bat idatzi behar duen prozedura:

## Definizioa:

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

## Erabilera:

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



# Prozedurak. Egitura

**algoritmo** Zuriuneak\_Idatzi (Zuriune\_Kop : **datu** Oso)

**Izena**

**Parametro-espezifikazioa**

**algoritmo** Zatiketa\_Moztua (M, N: **datu** Oso;  
Zatidura,  
Hondarra: **emaitza** Oso)



# Prozedurak. Egitura (II)

## Parametro formala

## Parametro errealak

### Definizioa:

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

### Erabilera:

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





# Prozedurak. Egitura (III)

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

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

**Gorputza:**

Emitza kalkulatzeko  
agindu-multzoa

*itzuli* agindurik ez dago



# Azpiprogramen parametroak

- Hiru parametro klase daude:
  - sarrerakoak
  - irteerakoak
  - sarrera-irteerakoak
- Datuak sartzeko ala emaitzak ateratzeko erabiliko diren, horretan datza diferentzia.



# Sarrera-parametroak

- Azpiprogramari datuak pasatzeko erabiltzen dira.
- Sarrera-parametro formal bakoitzak, dagokion parametro errealetik hartzen du bere balioa.
- *Konstante lokal* gisa jokatzen dute gorputza exekutatzen 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 : Oso)

**algoritmo** Zuriuneak\_Idatzi (Zuriune\_Kop : **datu** Oso)

- Parametro errealak adierazpen baten bidez erabiliko dira. Adibidez:
  - *Zuriuneak\_Idatzi (7)*
  - *Zuriuneak\_Idatzi (2\*N+4)*
  - *Zuriuneak\_Idatzi (Kop)*



# 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 jotzen du:
  - Prozedura-gorputzak balioa emango dio aldagai honi.
  - Azpiprogramaren exekuzioa bukatutakoan, parametro formalak duen balioa parametro errealari pasako zaio
- Irteera-parametro errealak bete behar dituen baldintzak:
  - Parametro formalaren mota berekoa izan behar du.
  - Derrigorrez aldagaia izan behar du.



# Irteera-parametroak (II)

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

```
algoritmo Zatiketa_Moztua
    (Zatikizun, Zatitzaile: datu Oso
     Zatidura, Hondarra: emaitza Oso)
```

**hasiera**

```
Zatidura := Zatikizun / Zatitzaile
Hondarra := Zatikizun mod Zatitzaile
```

**amaia**

*Zatiketa\_Moztua* prozedurak bi datu jaso, eta bi emaitza itzultzen ditu.

- Parametro errealak aldagaiak izango dira. Adibidez:

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

- Parametro errealek (aldagaiek) egokitzen zaizkien parametro formalen mota bera izan behar dute.



# Sarrera-Irteera parametroak

- Prozeduretan erabiltzen dira, eta parametro errealak (deian erabilitako aldagaiak) *eguneratzeko zeregina dute*.
- Honela jotzen da:
  - Deia egitean parametro errealaren balioa prozedurari pasatzen zaio (dagokion parametro formalari).
  - Prozeduraren gorputzean parametroaren balioa eguneratu egiten da.
  - Prozeduraren exekuzioa amaitutakoan, parametro errealak balio eguneratua hartzen du.
- Parametro-espezifikazioan, bi puntuen ondoren *datu emaitza* idatziz adierazten dira sarrera-irteerako parametroak.
  - Adibidez:

```
algoritmo Gehitu (Kont : datu emaitza Oso)
hasiera
    Kont := Kont + 1;
amaia Gehitu;
```
  - Horrela dei diezaiokegu prozedura honi:  
Gehitu (N)



# Sarrera-Irteera parametroak (II)

- Sarrera-irteera erako parametro formal bakoitzak prozedurako *barruko aldagai* baten moduan jotzen du.
- Prozedura-gorputzean sartzean, parametro formalaren 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

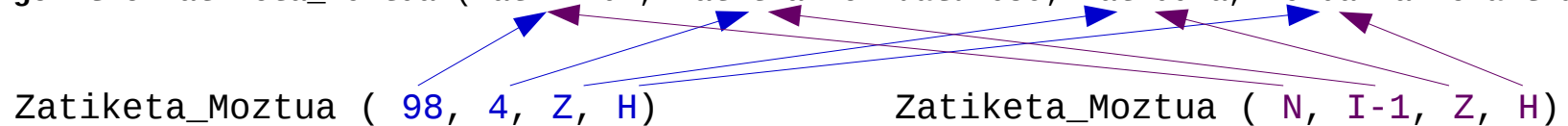
- Funtzioek sarrera-parametroak bakarrik eduki ditzakete.
- Murritzapen hori ondo justifikatuta dago:
  - *irteera*- edo *sarrera-irteera* erako parametroak onartuko balira, funtzioaren barrukoak ez diren aldagaien balioak ere aldatuko lirateke.
  - Horrelako fenomenoei *albo-ondorio* deritze.
- Funtzioetan, albo-ondorioak ez dira gomendagarriak
- Funtzioak balio bat itzuli behar du, ... eta kito!
  - Balio hori kalkulatzean besterik aldatzen badu, algoritmoen ulerkuntza zaildu egiten da.



# Parametro formalak eta errealak

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

**algoritmo** Zatiketa\_Moztua (Zatikizun, Zatitzaile: **datu** Oso; Zatikidura, Hondarra: **emaitza** Oso)



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

`Zatiketa_Moztua (Zatikizun => 98, Zatitzaile=> 4,  
Hondarra => H, Zatikidura=> Z)`

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



# Azpiprogramei eginiko deiak. Bete beharreko arauak

- **Zenbat parametro formal** definitu diren azpiprogramaren burukoan, **hainbeste parametro erreal** erabiliko dira deian.
- Parametro **formalen motek eta** dagozkien **errealen motek bat etorri behar dute**.
- Parametro formalen eta errealen parekatzea posizio bidez egiten da, besterik ezean.
  - Parekatze esplizitua ere egin daiteke.



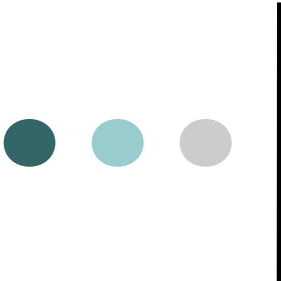
# Azpiprogramei eginiko deiak. Bete beharreko arauak (II)

- Funtzio-deiak:
  - **Funtzio-deiak balioen parekoak dira.**
  - **Funtzio-deiaren motak bat etorri behar du funtzioak itzuliko duen **emaitzaren motarekin**.**
  - **Parametro guztiek datu erakoak izan behar dute.**



# Azpiprogramei eginiko deiak. Bete beharreko arauak (III)

- Prozedura-deiak:
  - **Prozedura-deiak aginduen parekoak** dira.
  - Irteera- eta sarrera-irteera parametro errealek aldagaiak izan behar dute.



# Azpiprogramen zehaztapena.

## Aurrebaldintza eta postbaldintza

- Zehaztapena oso inportantea da azpiprogramen definizioetan.
- Azpiprogramaren erabiltzaileak moduluak **zer** egiten duen jakin behar du, ez besterik.
  - Azpiprograma erabiltzeko, nahikoa izango zaio azpiprogramaren burukoa eta zehaztapena ulertzearekin.
- Zehaztapenean ez dira berriro errepikatzen zeintzuk diren parametroak eta beren motak:
  - Bakarrik zehazten da zeintzuk diren betetzen dituzten propietateak.

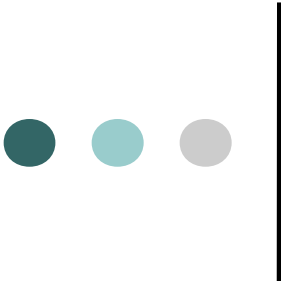


# Azpiprogramen zehaztapena.

## Adibideak

```
funtzio Alfabetikoa (Kar : Karaktere) itzuli Boolear
  --Aurrebaldintza:
  --Postbaldintza: Eraitza = egiazkoa baldin Kar letra bada,
  --                  Eraitza = faltsua bestela
```

```
algoritmo Zuriuneak_Idatzi (Zuriune_Kop : Oso)
  --Aurrebaldintza: Zuriune_Kop >= 0
  --Postbaldintza: Irt_Estandarra = <' ', ' ', ..., ' '>, non
  --                  zuriuneen kopurua Zuriune_Kop baita.
```



# Azpiprogramen zehaztapena.

## Sarrera-Irteera parametroak

```
algoritmo Gehitu (Kont : datu emaitza Oso)
```

```
-- Aurrebaldintza:
```

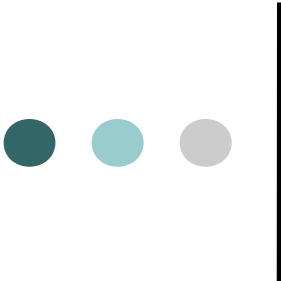
```
-- Postbaldintza: Kont = Kont + 1
```

?

```
hasiera
```

```
    Kont := Kont + 1;
```

```
amaia Gehitu;
```

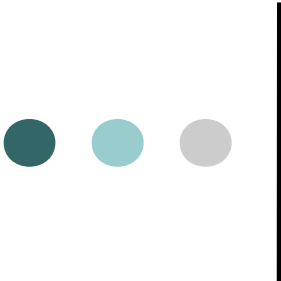


# Azpiprogramen zehaztapena.

## Sarrera-Irteera parametroak (II)

- Arazoa: parametro formalak balio desberdina du sarreran eta irteeran.
  - Zehaztapenean parametro formala erabiltzen dugunean, zein baliori buruz ari gara, sarrerakoaz ala irteerakoaz?
  - Nola adierazi parametro horri buruzko propietateak?
- Jarraibideak:
  - Aurrebaldintzan adierazten diren propietateak sarrerako balioei buruzkoak dira. Ez dago zalantzarik.
  - Postbaldintzan garbi adierazi behar da zeini buruz ari garen, sarrerako balioaz ala irteerako emaitzaz.





# Azpiprogramen zehaztapena. Sarrera-Irteera parametroak (III)

*Gehitu* azpiprograma bi modutan espezifikatuta:

Modu formalagoan:

```
algoritmo Gehitu (Kont : datu emaitza Oso)

-- Aurrebaldintza: Kont = a
-- (demagun Kont aldagaiak a balioa duela)
-- Postbaldintza: Kont = a + 1

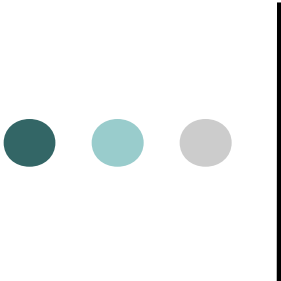
hasiera
    Kont := Kont + 1;
amaia Gehitu;
```

Modu informalagoan:

```
algoritmo Gehitu (Kont : datu emaitza Oso)

-- Aurrebaldintza:
-- Postbaldintza: Kont = Kont-en hasierako
--                               balioa + 1

hasiera
    Kont := Kont + 1;
amaia Gehitu;
```



# Aldagaien eta parametroen esparrua eta ikusgarritasuna

- Aldagai baten **esparrua**: non erabil daitekeen aldagai hori algoritmo (edo programa) multzo batean.
- Aldagai bat bere esparruan **ikusgarri** dela esaten dugu.
- Hasteko, gure algoritmoetan **aldagaiaren esparrua azpiprogramarena da**, alegia, aldagaia erazagututa dagoen azpiprogramarena.
- **Parametro formalen esparrua** ere definituta dauden **azpiprogramarena da**.
- Aldagai lokalak eta parametroak desagertu egiten dira azpiprogramaren exekuzioa amaitutakoan.
  - Horrela, horiek betetzen zuten memoria libre uzten da beste zeregin batzuetarako.



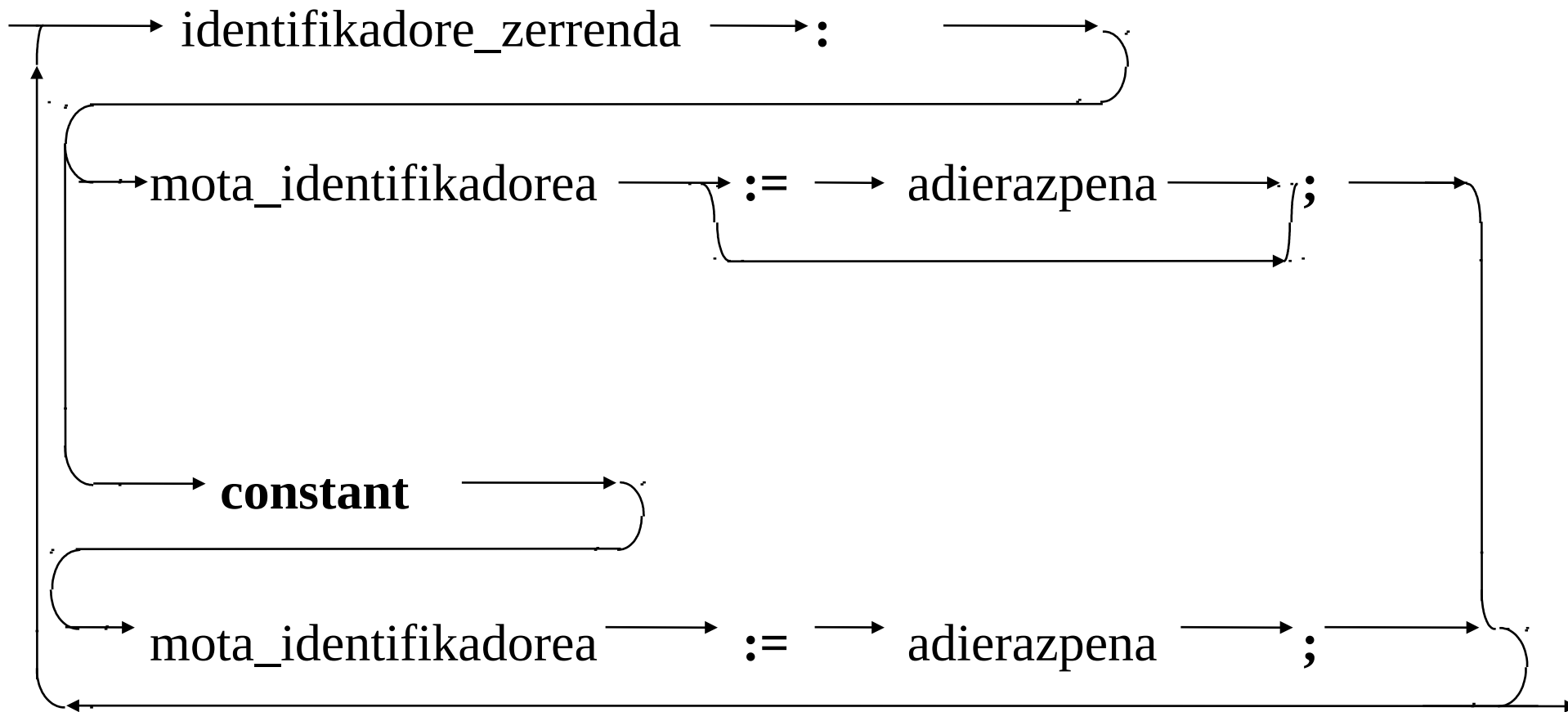
# Azpiprogramen idazkera ADAn (sinplifikatua)

```
procedure ident_prozedura (atal_formala) is  
    erazagupen_atala  
begin  
    sententzia_sekuentzia  
end ident_prozedura;
```

- Atal formalean parametroak eta beren motak definitzen dira.
  - Sarrera-parametroak: *in*
  - Irteera-parametroak: *out*
  - Sarrera-irteera parametroak: *in out*

# Azpiprogramak ADAn

## Erazagupen\_atala (sinplifikatua)





# Prozedura baten adibidea ADAn (I)

Ordua\_Erakutsi programak, gauerdiaz gero pasatu diren segundoak irakurtzen ditu eta 24 orduko adierazpidea erabiliz idazten du O:M:S formatuan.

Adibidez: 54450 sarrera-datuarekin programaren irteera hau da:

**< Ordua:   15:       7:       30 >**

```
with Irakurri_Osokoa, Idatzi_Osokoa, Idatzi_Katea ;
```

```
procedure Ordua_Erakutsi is
  Seg_Minutuko: constant Integer := 60;
  Min_Ordubeteko: constant Integer := 60;
  Seg_Ordubeteko: constant Integer := Seg_Minutuko * Min_Ordubeteko;
  Ordua      : Integer;
  O, M, S : Integer;
```



# Prozedura baten adibidea ADAn (II)

**begin**

```
    Irakurri_Osoa (Ordua);                -- gauerdiaz geroko segundoak
    H := Ordua / Seg_Ordubeteko;           -- gauerdiaz geroko orduak
    Ordua := Ordua rem Seg_Ordubeteko;     -- azken orduaz geroko segundoak
    M := Ordua / Seg_Minutuko;             -- azken orduaz geroko minutuak
    S := Ordua rem Seg_Minutuko;           -- azken minutuaz geroko segundoak
    Idatzi_Katea ("Ordua:  ");
    Idatzi_Osoa (0); Idatzi_Katea (":");
    Idatzi_Osoa (M); Idatzi_Katea (":");
    Idatzi_Osoa (S);
end Ordua_Erakutsi;
```



# Funtzio baten adibidea ADAn

```
function Faktoriala (N : in Integer) return Integer is  
  -- Aurrebaldintza: N > 0  
  -- Postbaldintza: Emaizta = N zenbakiaren faktoriala  
  Fakt : Integer := 1;  
  N, I : Integer;  
begin  
  I := 1;  
  while I <= N loop  
    Fakt := Fakt * I ;  
    I := I + 1 ;  
  end loop ;  
  return Fakt ;  
end Faktoriala ;
```

# Azpiprograma bat erabili beste azpiprograma batetik ADAn (I)

- Aukera1:
  - Azpiprograma lagungarria (prozedura edo funtzioa) fitxategi batean definitzen da
    - Fitxategiaren izena = azpiprogramaren izena
  - Azpiprogramak beste edozein kanpoko programatan erabil daitezke
    - Azpiprograma lagungarria erabili nahi duen azpiprograma nagusiaren fitxategian, definizioa baino lehen, *with* erako sententzia bat erabiltzen da.

***b.adb*** fitxategia:

```
procedure B ... is  
  
begin  
  ...  
end B;
```

***a.adb*** fitxategia:

```
with B;  
procedure A ... is  
begin  
  ...  
  B (...)  
  ...  
end A;
```



# Azpiprograma bat erabili beste azpiprograma batetik ADAn (II)

- Aukera2:
  - Azpiprograma lagungarria azpiprograma nagusiaren **barruan** definitzen da.
    - Kasu honetan azpiprograma lagungarri hori ezin da erabili azpiprograma nagusi horretatik kanpo, azpiprograma **lokala** izango baita.

***a.adb*** fitxategia:

```
procedure A ... is
    ...
    procedure B ... is
        begin
            ...
        end B;
    begin
        ...
        B(...)
        ...
    end A;
```



# Programazio-estiloari buruzko oharrak

- Azpiprogramek izenak ematen dizkiete programen zatiei. Azpiprogramei (eta objektu edo datu-motak bezalako entitateei) **izen zentzudunak** emanaz gero, programa "testu" gisa irakur daiteke.
- Azpiprogramarentzat izen egokia aukeratzea errazagoa da, baldin eta 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-estiloari buruzko oharrak (II)

- Azpiprogramaren *zehaztapena* aldatzen ez den bitartean, azpiprogramari egiten zaizkion deiak ez dira aldatu beharko.
  - Azpiprogramaren gorputza aldatzen bada ere, zehaztapenari eutsiz gero, programa deitzaileek ez dute zertan aldatu.
- Komeni da iruzkinak erabiltzea azpiprogramaren eginkizuna eta funtzionamendua ondo azaltzeko. Halaber, azpiprograman erabiltzen diren aldagaien eta parametroen betekizuna iruzkinen bidez zehaztea mesedegarria da.
  - Iruzkinak ADAn: "--" ondoren lerro bukaeraraino
- Identifikadore ahalik eta deskriptiboenak aukeratu behar dira.
  - Identifikadoreak ADAn: hitzak letra maiuskulez hasi eta azpimarrez bereizi.
    - Adibidez: Zuriuneak\_Idatzi, Irakurri\_Karakterea, Seg\_Minutuko



# Programazio-estiloari buruzko oharrak (III)

- Ada programazio-estiloa:
  - [http://en.wikibooks.org/wiki/Ada\\_Style\\_Guide](http://en.wikibooks.org/wiki/Ada_Style_Guide)