5. gaia PROGRAMEN ERATORPEN FORMALA

5.1. Burstall-en metodoa: programa errekurtsiboak iteratibo bihurtzeko.

5.2. Metodoa eta adibideak.

5.1. Burstall-en metodoa: programa errekurtsiboak iteratibo bihurtzeko.

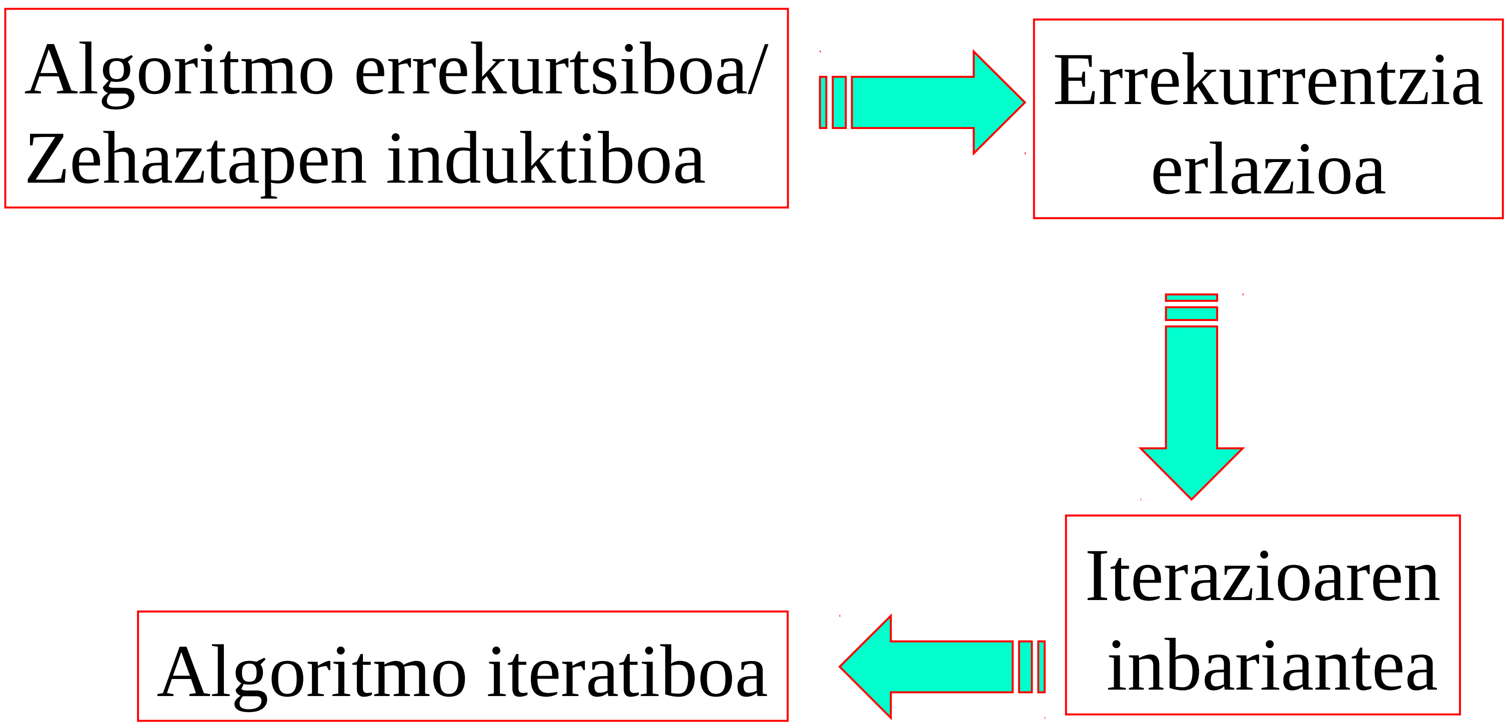
- *Programa errekurtsiboak soluziorik <u>zuzenena</u>, <u>sinpleena</u> eta <u>argiena</u> izan ohi dira zenbait problematarako; hala nola:*
  - *indukzio bidez definitutako funtzioak*
  - *datu-mota induktiboen tratamendua*
  - *izaera bereko azpiproblematan banatzen diren problemak*
- *Baina, askotan ez dira eraginkorrak:*
  - *kalkuluen errepikapena*
  - *parametroak gorde eta berreskuratzea*

Transformazioa: beste diseinu-metodo bat

- *Bestelako ebazpideak asmatzea zaila denean, ebazpide errekurtsiboa erabil daiteke abiapuntu gisa.*
  - *Horrela jokatuz gero, ebazpide eraginkorragoa lor daiteke, modu metodiko batez.*
  - *Azkeneko programaren zuzentasuna justifikatuta geratzen da.*

Gainera, transformazio-mota honen bidez <u>errekurtsibitatea eta honek iterazioarekin duen erlazioa</u> sakonki azter daiteke.

Burstall-en metodoa (***Burstal&Darlintong,1977***)

- *Funtzio errekurtsiboak iteratibo bihurtzeko balio du.*
- 

Burstall-en metodoa (***Burstal&Darlintong,1977***)

- *Definizio induktibotik eratortzen den <u>errekurrentzi erlaziotik</u> lortzen da <u>inbariantea</u>.*
- *Inbariante hori abiapuntu hartuta, <u>iterazioa formalki eratortzen da</u>: <u>hasieraketa</u>, <u>bukaerako tratamendua</u> eta <u>iterazioaren gorputza</u>.*
- *Definizio induktiboan eta inbariantean oinarrituta, <u>destolestaketa/tolestaketa</u> teknika erabiltzen da.*

Burstall-en metodoa. Adibidea: *Oinarri aldaketa*

$\text{oinald: Integer} \times \text{Integer} \rightarrow \text{Sekuentzia(Integer)}$

Aurre: $1<b<10 \wedge x\geq 0$

$$\text{oinald}(x,b) = \begin{cases} \langle x \rangle & \text{baldin } x<b \\ \text{oinald}(x/b,b) @ \langle x \bmod b \rangle & \text{bestela} \end{cases}$$

oinald(x,b) *x-ren b oinarriko errepresentazioa da.*

Honakoan definizio induktibotik abiatzen da. Berdin jokatzen da algoritmo errekurtsibotik abiatuta ere.

Adibidea: *Oinarri aldaketa (jarraipena)*

oinald(123,4)

destolestu/tolestu

= oinald(30,4)@<3>

destolestu

= (oinald(7,4)@<2>) @<3>

tolestu

= oinald(7,4)@<2,3>

destolestu

= (oinald(1,4)@<3>) @<2,3>

tolestu

= oinald(1,4)@<3,2,3>

= <1>@<3,2,3>

= <1,3,2,3>

$\boxed{\text{oinald(x,b)=oinald(y,b)@S}}$

Adibidea: *Oinarri aldaketa (jarraipena)*

- Inbariantea $\boxed{oinald(x,b)=oinald(y,b)@S}$

- Hasieraketa: `y:=x; S:=< >;`

  oinald(x,b) = oinald(y,b)@S

  = oinald(x,b)@< >

  = oinald(x,b)

- Bukaerakoa:

  y<b → oinald(x,b) = oinald(y,b)@S

  = <y>@S = y•S

  Hau da, iterazioaren bukaera baldintza y<b *da, eta itzuli beharreko emaitza:* y•S

Adibidea: *Oinarri aldaketa (jarraipena)*

- Iterazioaren gorputza:

oinald(x,b)

= oinald(y,b)@S

destolestu

tolestu

= (oinald(y/b,b)@<y mod b>)@S

= oinald(y/b,b)@(<y mod b>@S)

= oinald(y´, b)@ S´

Hau da, iterazioaren gorputza honako aldibereko asignazioa da:

(y,S):=(y/b,<y mod b>@S):

```
S:=<y mod b>@S;
y:=y/b;
```

edo

```
S:=(y mod b)•S;
y:=y/b;
```

Adibidea: *Oinarri aldaketa (jarraipena)*

```
function Oinald_it(x,b:Integer)
             return Integer is
 y:Integer; S:Sekuentzia(Integer);

begin {1<b<10 ∧ x≥0}
    y:=x; S:=< >;  {oinald(x,b)=oinald(y,b)@S}
 while not y<b loop
    S:=(y mod b)•S;
    y:=y/b;
 end loop;
 return y•S ;
end Oinald_it; {oinald_it(x,b)=oinald(x,b)}
```

Metodo orokorra

- *Funtzio baten definizio induktiboa daukagu:*

$$f:\ T_1 \times \ldots \times T_n \rightarrow T'$$

parametro formalak: $\bar{x}$

- *Definizio induktibo horretatik errekurrentzi erlazio bat ateratzen da, eta hau inbariante baten bidez adieraz daiteke:*

$$\mathrm{INB} \equiv \boxed{f(\bar{x}) = \mathrm{F}(f(\bar{y}), \bar{z})}$$

non, $\bar{y}, \bar{z}$ *aldagai berriak diren, iterazioan maneiatuko direnak.*

- *Funtzio iteratiboak honako forma hartzen du:*

Metodo orokorra

function f($\overline{x}$:$\overline{T}$) **return** T' **is**

$\overline{y}$:$\overline{T}$; $\overline{z}$: ?;

$\overline{y}$,$\overline{z}$ aldagaiak hasieratu INB hasieran bete dadin

begin

Has; INB $\equiv$ { f($\overline{x}$)= F(f($\overline{y}$),$\overline{z}$) }

while not bukbal($\overline{y}$) **loop**

($\overline{y}$,$\overline{z}$):=($\overline{y}$',$\overline{z}$');

destolestu/tolestu

end loop;

Emaitza;

end f;

f($\overline{y}$) ezin da gehiago destolestu kasu sinplerik gertatu gabe. f($\overline{y}$)-k une honetan duen balioa INB formulan jarriz lortzen da emaitza

Metodo orokorra: Laburpena

- *Errekurrentzi erlazioa aztertu eta INB lortu (beharrezkoak diren aldagai berriak erabili)*
- *Aldagai berri horiek* hasieratu *inbariantea bete dadin*
- *Inbariantea gehiago tolestu ezin denean* bukatzen da *prozesu iteratiboa*. Emaitza *lortzeko une horretako balioa INB inbariantean ordeztu*
- *Inbariantea* destolestu eta tolestu, *inbariantea kontserbatzen dela*, iterazio-pauso bakoitzean *zer egin erabakitzeko*

Metodo orokorra

- *Eskema orokortzen:*

$$f(\bar{x}) = \underline{\text{if}}\ \text{a}(\bar{x})\ \underline{\text{then}}\ \text{g}(\bar{x})$$
$$\underline{\text{else}}\ \text{f}(t(\bar{x})) \otimes h(\bar{x})$$

- Kasu sinple bat baino gehiago:

  Bukaerako baldintza: disjuntzioa

  Kasu bakoitzeko emaitza desberdina: baldintzazko emaitza.

- Kasu induktibo bat baino gehiago:

  Inbarianteak kasu guztiak bildu behar ditu, destolestu/tolestu prozesuan nabarmentzen denez.

Metodo orokorra

- Eragiketa $\otimes$ ez-elkarkorra

  Eragiketa orokorragoa duen inbariantea, hau da:

  $f(\bar{x}) = f(\bar{y}) \otimes z$ formularen ordez:

- Erekurtsibitate anitza: $f(\bar{x}) = F(f(\bar{y}), \bar{z})$

  Errekurrentzia destolestean dei bat baino gehiago eduki behar dira kontuan.

  Aztertu behar da zein dei-konbinazio destolestu behar den, gero tolestu ahal izateko.

Adibidea: *sekuentzien nahasketa*

$$\text{nahastu}:\text{sekuentzia}(T) \times \text{sekuentzia}(T) \rightarrow \text{sekuentzia}(T)$$

(1) nahastu(< >,r) = r

(2) nahastu(s,< >) = s

$$\text{(3) nahastu}(x\bullet s, y\bullet r) = \begin{cases} x\bullet\text{nahastu}(s, y\bullet r) & \text{baldin } x \leq y \\ y\bullet\text{nahastu}(x\bullet s, r) & \text{bestela} \end{cases}$$

- nahastu(<1,4,6>,<2,3>) = 1•nahastu(<4,6>,<2,3>)

  = 1•(2•nahastu(<4,6>,<3>))

  $= _{\bullet\ \text{ez-elkarkorra}}$ <1,2>@nahastu(<4,6>,<3>)

  = <1,2,3>@nahastu(<4,6>,<>)

  = <1,2,3,4,6>

- <u>Inbariantea</u>: $\boxed{\text{nahastu}(s,r) = u@\text{nahastu}(v,w)}$

Adibidea: *sekuentzien nahasketa (jarraipena)*

- <u>Has</u> ≡ `u:=< >; v:=s; w:=r;`

  nahastu(s,r) = u@nahastu(v,w)

  = < >@nahastu(s,r) = nahastu(s,r)

- sinple(v,w) ≡ hutsa_da(v) ∨ hutsa_da(w)

nahastu(s,r)

$$= u@nahastu(v,w) = \begin{cases} u@w \text{ baldin } hutsa\_da(v) \\ u@v \text{ baldin } hutsa\_da(w) \end{cases}$$

```
while not(hutsa_da(v) or hutsa_da(w)) loop
 …
end loop;
if hutsa_da(v) then return u@w;
else return u@v; end if;
```

<u>Emaitza</u>

Adibidea: *sekuentzien nahasketa (jarraipena)*

lehena(v)≤lehena(w)→ nahastu(s,r)=u@nahastu(v,w)

= u@(lehena(v)•nahastu(hondarra(v),w))

= (u@<lehena(v)>)@nahastu(hondarra(v),w)
= u´ @nahastu(v´, w´)

lehena(v)>lehena(w)→ nahastu(s,r) =u@nahastu(v,w)

= u@(lehena(w)•nahastu(v,hondarra(w))

= (u@<lehena(w)>)@nahastu(v,hondarra(w))
= u´ @nahastu(v´, w´)

```
if lehena(v)≤lehena(w)
   then u:=u@lehena(v);v:=hondarra(v);
   else u:=u@lehena(w);w:=hondarra(w);
end if;
```

Adibidea: *sekuentzien nahasketa (jarraipena)*

```
function nahastu_it(s,r:sekuentzia(T))
      return sekuentzia(T) is
  u,v,w:sekuentzia(T);
begin   {true}            E ≡ luz(v)+luz(w)
u:=< >; v:=s; w:=r; {nahastu(s,r)=u@nahastu(v,w)}
while not(hutsa_da(v) or hutsa_da(w))loop
   if lehena(v)≤lehena(w)
   then u:=u@lehena(v);v:=hondarra(v);
   else u:=u@lehena(w);w:=hondarra(w);
   end if;
end loop;
if hutsa_da(v) then return u@w;
else return u@v; end if;
end nahastu_it; {nahastu_it(s,r)=nahastu(s,r)}
```

Errekurtsibitate anitzeko adibidea

```
function fib (n:integer) return integer is
begin  {n≥0}
      if n≤1 then return n;
      else return fib(n-1)+fib(n-2); end if;
end fib;  {fib(n)= s_n}
```

$s_0=0,\ s_1=1,\ s_n= s_{n-1}+ s_{n-2}\ (n\geq 2)$

Errekurrentzi erlazioa:*(argumentu handiena duen deia destolestu)*

$fib(5) = \underline{fib(4)}+fib(3) = \underline{(fib(3)+fib(2))}+fib(3)$

$= 2.\underline{fib(3)}+fib(2) = 2.(\underline{fib(2)+fib(1)})+fib(2)$

$= 3.\underline{fib(2)}+2.fib(1) = 3.(\underline{fib(1)+fib(0)})+2.fib(1)$

$= 5.fib(1)+3.fib(0) = 5.1+3.0 = 5$

Errekurtsibitate anitzeko adibidea (jarraipena)

- Inbarianteak:
  - ❶ fib(n) = x.fib(u) + z.fib(v)
  - ❷ fib(n) = x.fib(u) + z.fib(u-1)
  - ❸ fib(n) = x.fib(v+1) + z.fib(v)
- Has ≡ `x:=0; z:=1; v:=n;`

  fib(n) = x.fib(v+1) + z.fib(v)

  = 0.fib(n+1) + 1.fib(n) = fib(n)
- sinple(v+1,v) ≡

  `v=0` → fib(n) = x.fib(v+1) + z.fib(v)

  = x.fib(1) + z.fib(0)

  = x → Emaitza ≡ `return x;`

Errekurtsibitatate anitzeko adibidea (jarraipena)

- <u>Iterazioaren gorputza</u>:

  $fib(n) = x.fib(v+1) + z.fib(v)$

  $= x.(fib(v)+fib(v-1)) + z.fib(v)$

  $= (\underline{x+z}).fib(\underline{v}) + \underline{x}.fib(\underline{v-1})$

  $= \underline{x'}. fib(\underline{v'+1}) + \underline{z'}.fib(\underline{v'})$

  Aldi bereko asignazioa da:

  $(x,z,v):=(x+z,x,v-1)$

  Inplementazioa:

```
lag:=x;
x:=x+z;
z:=lag;
v:=v-1;
```

Errekurtsibitate anitzeko adibidea (jarraipena)

```
function fib_it (n:integer) return integer is
  x,z,v,lag:integer;
begin {n≥0}                    E ≡ v
x:=0; z:=1; v:=n; {fib(n) = x.fib(v+1) + z.fib(v)}
while not v=0 loop
   lag:=x;
   x:=x+z;
   z:=lag;
   v:=v-1;
end loop;
return x;
end fib_it; {fib_it(n) = fib(n)}
```