2.4 Egitura errepikakorrak

For, while, do …loop

Errepikapena

- Orain arte ikusi diren programetan agindu bakoitza 1 edo 0 aldiz exekutatu zitekeen

- Kasu askotan aginduak errepikatzea interesatzen zaigu
  - Sarrerako datuak asko dira eta berdin tratatu nahi dira
  - Ekintza edo kalkulu bat askotan errepikatu nahi da

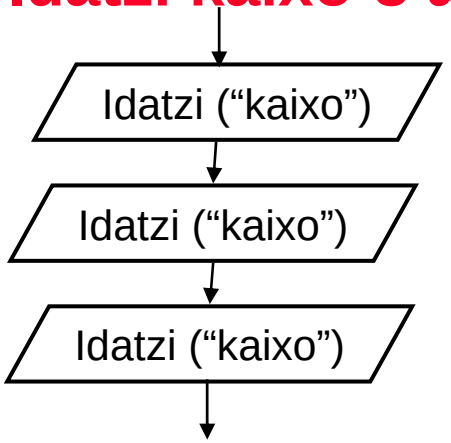
    - Ikasle baten bi noten batez bestekoa kalkulatzeko:

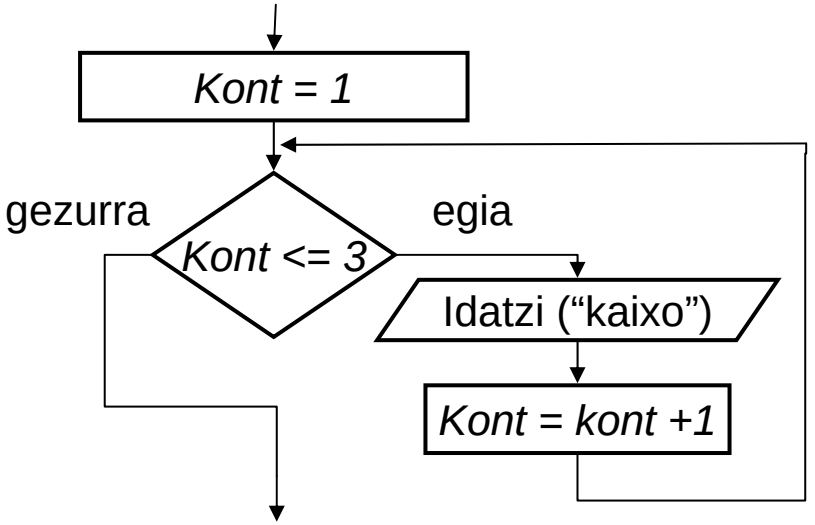
```
N1 = InputBox("Eman 1. ikasgaiaren nota: ")
N2 = InputBox("Eman 2. ikasgaiaren nota: ")
Print ( (N1+N2)/2 )
```

    - eta 100 ikasleren noten batez bestekoa kalkulatzeko?

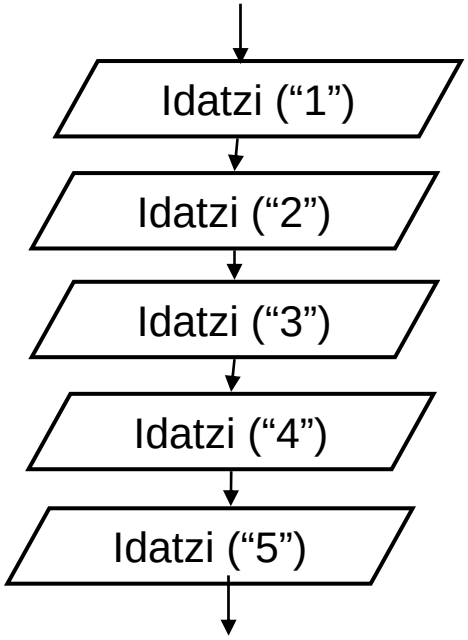
Errepikapena. Adibideak

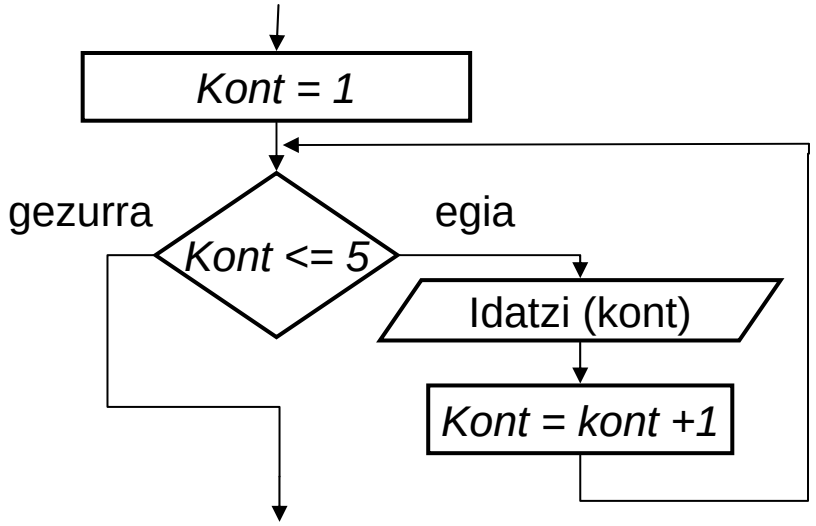
1. Idatzi kaixo 3 aldiz





2. Idatzi 1-etik 5-rako zenbakiak

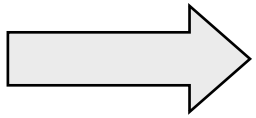




Errepikapena: kopuru finitua: For

1. **Idatzi kaixo 3 aldiz**

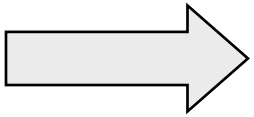
```
Print “kaixo”
Print “kaixo”
Print “kaixo”
```



```
For kont=1 to 3 step 1
    Print “kaixo”
Next kont
```

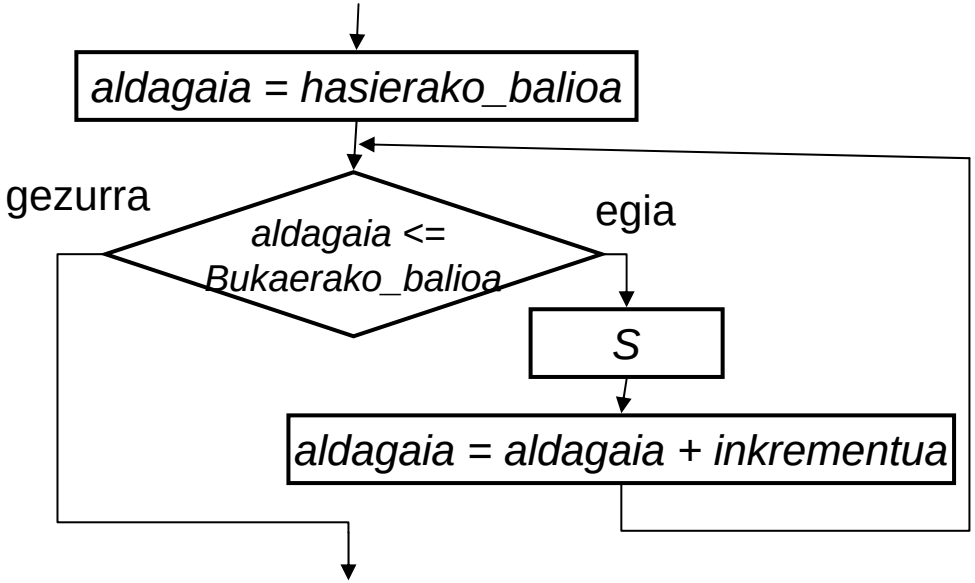
2. **Idatzi 1-etik 5-rako zenbakiak**

```
Print “1”
Print “2”
Print “3”
Print “4”
Print “5”
```



```
For kont=1 to 5 step 1
    Print kont
Next kont
```

Oinarrizko algoritmo egitura



for *aldagaia* **=** *hasierako_balioa* **to** *Bukaerako_balioa* **step** *inkrementua*

S

next aldagaia

- Ikusi 2.2.1. Algoritmoak
  - Oinarrizko algoritmo egitura (12)

2.4.1. For. Sintaxia

- **For** agindua::=

  for *<indizea>* **=** *<hasi_balio>* **to** *<amaia_balio>* [**step** *exp*]

  <agindua>

  ....

  next *<indizea>*

- Oharrak:

<indizea>: beti aldagai bat izango da

<hasi_balio>: zenbaki oso bat aldagai, literal edo adierazpen moduan

<amaia_balio>: zenbaki oso bat aldagai, literal edo adierazpen moduan

<exp>: zenbaki oso bat aldagai, literal edo adierazpen moduan

<agindua>: orain arte ikusitako edozein: esleipen-sententzia, baldintza egitura (if-then-else edo select case) edo

egitura errepikakorra (for, while, do..loop)

For-to ren Semantika

- Indize bat erabili eta honek begiztaren pasada bakoitzean zein balio hartuko dituen adierazi behar da
- *<hasi_balio>* ≤ *<amaia_balio>* , orduan step-en dagoen balioa handituz joan behar du indizea. Begiztaren pasada bakoitzean indizearen balioa automatikoki “handitzen” da, *exp* -en adierazten den moduan hurrengo balioa hartuz
- *<hasi_balio>* **>** *<amaia_balio>*, orduan for-agindua amaitzen da eta ondorengo agindua exekutatzera goaz. Hots, for gorputzeko aginduak ez dira behin ere egikaritu

KONTUZ! Indizearen balioak automatikoki aldatzen dira for-aren gorputzean. Erabiltzaileak EZIN DITU aldatu

For. Adibideak

- N zenbaki oso bat irakurri ondoren, 1 + 2 + 3 +...+ N = $\left\{\sum_{K=1}^{N} K\right\}$ kalkulatzeko programa egin

```
N = InputBox("Eman N-ren balioa")
Batukaria = 0;
for  K = 1 to N step 1
    Batukaria = Batukaria + K
next K
```

$$\left\{\text{Batukaria} = \sum_{K=1}^{N} K\right\}$$

```
N = InputBox("Eman N-ren balioa")
Batukaria= 0;
for  K = N to 1 step -1
        Batukaria= Batukaria + K;
next K
```

$$\left\{\text{Batukaria} = \sum_{K=N}^{1} K\right\}$$

```
N = InputBox("Eman N-ren balioa")
Batuz = 0;
for  K  = 1 to N step 1
        Batuz = Batuz + K
        print K, " pausoan "
        print "progresio "
        print "aritmetikoa: "
        print Batuz
 next K
```

For. Ariketa

- Teklatu bidez sartuko diren noten batez bestekoa kalkulatu nahi da programa baten bidez. Erabiltzaileak hasiera zenbat nota sartuko dituen adieraziko du. Programa interaktiboa izango da eta honelako itxura eduki lezake:

Exekuzioaren adibidea:

Zenbat nota sartuko dituzu? **7**

1 nota: **7.50**

2 nota: **6.40**

3 nota: **4.35**

...

7 nota: **9.75**

Aurreko noten batez bestekoa: **6.80**

Sekuentzia baten minimoaren kalkulua

- Zenbaki osoen sekuentzia baten luzera N izanik, sekuentzia horretako balio minimoa kalkulatzen eta pantailaratzen duen programa bat diseinatu eta idatzi.

2.4.2. While sententzia

- **Errepikapen kopurua finitua ez denean, zer?**

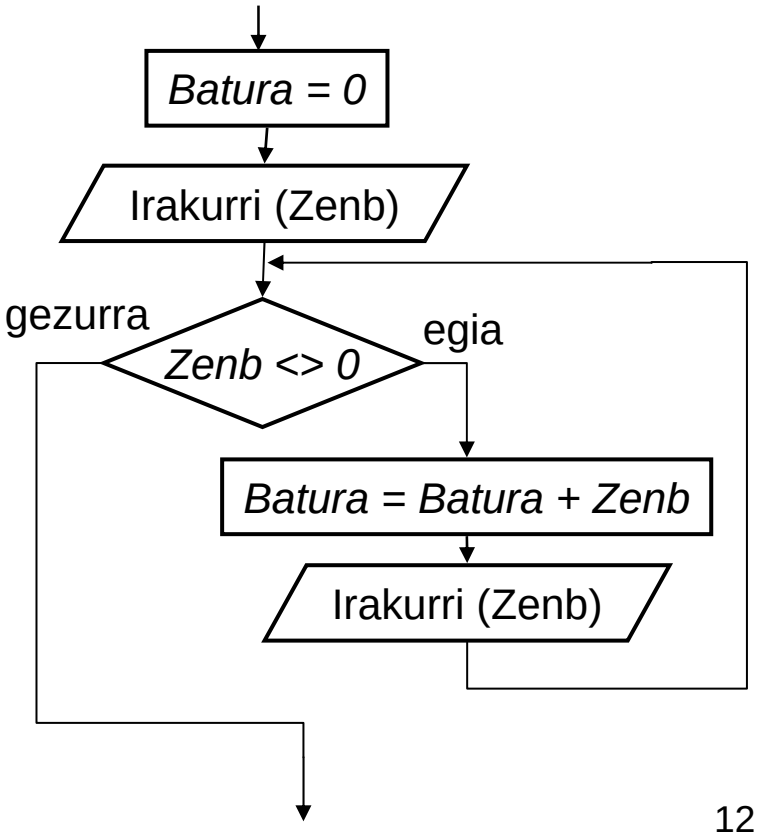
Orduan, bestelako baldintza bat asmatu behar

- Adibidea: Teklatu bidez irakurritako zenbaki osoen sekuentzia baten batura kalkulatu. Sekuentziaren amaiera 0 zenbakiarekin adierazten da.
  - Analisia:
    - **Sarrerako datuak:** zenbaki sekuentzia bat
    - **Irteerako datuak:** batura, sekuentziako zenbakiena
    - **Prozesua:** Zenbaki bat irakurri eta aurreko baturari batu, beste zenbaki bat irakurri eta aurrekoari batu, …
    - **Zenbat aldiz??:** EZ DAGO KOPURU FINITURIK!! sarreran irakurritako **zenbakia <> 0** den bitartean

Adibidearen soluzioa algoritmikoki

- Sarrerako datuen kasuak aztertu:
  - Sekuentzia hutsa izan daiteke (zenbaki bakarra eta hura 0)
  - Sekuentzia ez-hutsa, gutxienez zenbaki bat 0-ren ezberdina
  - Beraz, prozesua 0 edo n aldiz errepikatu daiteke (n sekuentziaren luzera izanik, 0 kontutan hartu gabe)

```
Batura = 0
Irakurri Zenb
bitartean Ez da sekuentziaren amaiera egin
    Batura = Batura + Zenb
    Irakurri Zenb
ambitartean
```



2.4.2. Do... Loop Sententzia

- **Errepikapen kopurua finitua ez denean, zer?**

Orduan, bestelako baldintza bat asmatu behar

- Adibidea: Teklatu bidez zenbaki bat sartu eta errepikatu datu-sarrera sartutako zenbakia 1 eta 10 artean ez badago. Amaieran, emandako zenbaki ona idatzi.
  - Analisia:
    - **Sarrerako datuak:** zenbaki bat
    - **Irteerako datuak:** sarrerako zenbaki ona
    - **Prozesua:** Zenbaki bat irakurri, begiratu onargarria den, onargarria ez bada aurreko bi pausoak errepikatu, onaragarria bada prozesua amaitu pantailan zenbakia idatziz
    - **Noiz arte??:** sarreran irakurritako zenbakia 1 eta 10 artean egon arte

Adibidearen soluzioa algoritmikoki

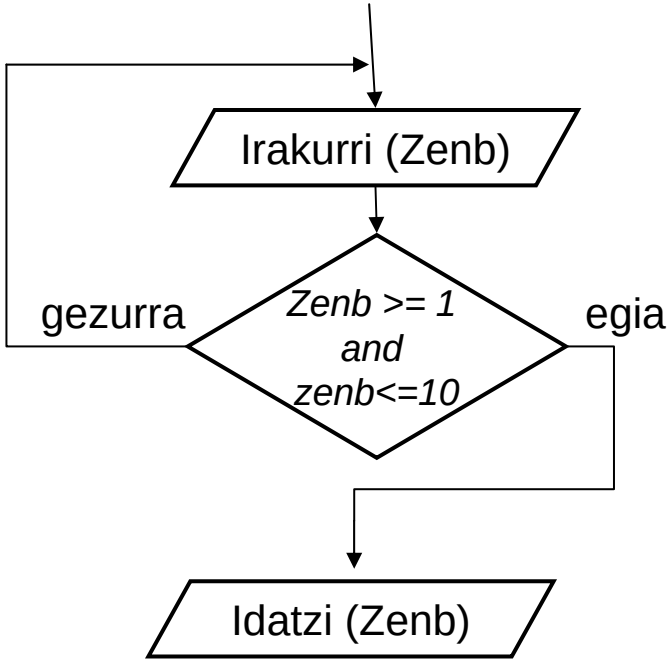
- Sarrerako datuen kasuak aztertu:
  - Zenbakia ez da zuzena (negatiboak, 0 eta 10 baino handiagoak)
  - Zenbakia zuzena da (1 eta 10 artekoak)

errepikatu

Irakurri Zenb

zenb >=1 and zenb <= 10 **bete arte**

Idatzi Zenb



Oinarrizko algoritmo egiturak

- Ikusi 2.2.1. Algoritmoak
  - Oinarrizko algoritmo egitura (10)
  - Oinarrizko algoritmo egitura (11)

2.4.2. WHILE + DO... LOOP (Sintaxia: Vbasic)

- **WHILE** agindua::=

  while <baldintza>

  <agindua>

  wend

- **do ... loop** agindua::=

  do

  <agindua>

  loop {while/until} <baldintza> **;**

WHILE+ DO...LOOP. SEMANTIKA

WHILE:

- Baldintza ebaluatu
  - ❖emaitza faltsua bada (False), bukatu eta jarraitu while aginduaren ondoren dagoen aginduarekin
  - ❖emaitza egiazkoa bada (True), orduan while aginduaren barneko agindu-multzoa exekutatu

DO...LOOP WHILE:

- Do...loop-aginduaren gorputzeko agindu multzoa exekutatu
- Baldintza ebaluatu
  - ❖emaitza faltsua bada (False), bukatu eta do...loop aginduaren ondoren dagoen aginduarekin jarraitu
  - ❖emaitza egiazkoa bada (True), orduan do...loop aginduaren barneko agindu-multzoa egikaritu

WHILE+ DO...LOOP. SEMANTIKA

DO...LOOP UNTIL:

- Do...loop-aginduaren gorputzeko agindu multzoa exekutatu
- Baldintza ebaluatu
  - ❖emaitza egiazkoa bada (True), bukatu eta do...loop aginduaren ondoren dagoen aginduarekin jarraitu
  - ❖emaitza faltsua bada (False), orduan do...loop aginduaren barneko agindu-multzoa egikaritu

DO..LOOP Adibidea

- **Ariketa**: Diseinatu eta idatzi programa bat, lehenengo 100 zenbakiak 20-naka pantailaratzen dituena. Gainera, 20 zenbakiak pantailaratu bakoitzeko jarrian ondorengo mezua: “PROZESUA AMAITU NAHI ZUZU, (Bai/Ez)?” idazten du. 100. zenbakia pantailaratzen denean, prozesua amaitutzat emango da egindako galderaren erantzuna kontutan hartu gabe.

Adibidea:

1

2

….

20

Prozesua Amaitu nahi al duzu, (Bai/Ez): ez

BEGIZTEN DISEINUA

- Begiztaren prozesua
  - Zer errepikatu behar da?
  - Nola hasieratu?

- Begiztaren kontrola
  - Noiz bukatuko da errepikapena? Bukaerako baldintza
  - Nola eguneratuko da baldintza?
  - Nola hasieratu baldintza

- Diseinuaren zuzentasunaren konprobazioa trazen bidez

WHILE. ADIBIDEAK

Zenbaki bat irakurri eta kalkulatu 2^K moduko lehen berredura, zenbakia-ren berdina edo handiagoa dena.

```
REM Zenbakia > 1
Berredura = 1;        ‘ 2^0 = 1
while Berredura < Zenbakia
        Berredura = Berredura*2
wend
print “Lehen berredura handiena: “
print Berredura
```

Puntu karaktereaz amaitzen den karaktere sekuentzia bat irakurri eta kontatu zenbat karaktere dituen sekuentziak

```
Kop =0
input Kar
while Kar <> ‘.’
    Kop = Kop + 1
    input Kar
wend
print “karaktereak “
print “guztira: “, Kop
```

DO...LOOP. ADIBIDEAK

Zenbaki bat irakurri eta kalkulatu 2^K moduko lehen berredura, zenbakia-ren berdina edo handiagoa dena.

```
REM Zenbakia > 1
Berredura=1        ‘ 2^0 = 1
do
      Berredura = Berredura*2
loop until Berredura>=Zenbakia
print “Berredura hurbilena: “
print Berredura
```

Idatzi programa bat prozesua errepikatu nahi duen galdetzen duena, erantzuna BAI den kasuan prozesua errepikatzen da eta bestela, programa amaitzen da.

```
Do
    erantzuna = InputBox _
          (“errepikatu nahi duzu?”)
loop while erantzuna = “BAI”
```

KONTROL-EGITURA ERREPIKAKORRAK

Noiz zein egitura erabili?

- Zenbat aldiz errepikatu behar den ezaguna denean **for** erabili.

  Kontrolerako aldagaiak zenbaki oso bat izan behar du

- **Do-loop** egitura begizta gutxienez behin errepikatu behar denean

- Gainontzeko kasuetan **while** egitura erabili