```
module Murgilketa3 where
import Zerrendak2
import Zerrendak
import Murgilketa

--D ATALEKO ARIKETAK

--1
-- Funtzio honek x eta bid zenbaki osoak emanda, [1..x] tarteko
-- elementuen biderkadura bider bid itzuliko du emaitza bezala.
-- Biderkadura hori kalkulatzeko [1..x] tarteko zenbakiak atzeraka
-- zeharkatu beharko dira, hau da, x-tik hasi eta 1eraino.
-- Kasu berezi bezala, x parametroaren balioa 0 baldin bada,
-- emaitza bid izango da eta 0 baino txikiagoa baldin bada,
-- errore-mezua aurkeztuko da.

-- Bigarren parametroa (hau da, bid) [1..x] tarteko zenbakien biderkadura
-- gordetzeko erabiliko da.

fakt_be_lag:: Int -> Int -> Int
fakt_be_lag x bid
        | x < 0                 = error "Lehenengo datua 0 baino txikiagoa da."
        | x == 0                = bid
        | otherwise             = fakt_be_lag (x - 1) (bid * x)

-----------------------------------------------------------------

-- Funtzio honek 0 baino handiagoa edo berdina den x zenbakiaren faktoriala
-- kalkulatuko du. Beraz, x parametroaren balioa 1 edo handiagoa
-- baldin bada, [1..x] tarteko elementuen biderkadura itzuliko da eta, kasu
-- berezi bezala, x parametroaren balioa 0 baldin bada, emaitza 1 izango da.
-- Emandako x balioa 0 baino txikiagoa baldin bada, errore-mezua aurkeztuko da.

-- Murgilketa erabiliko da.

fakt_be:: Int -> Int
fakt_be x
        | x < 0         = error "Datua 0 baino txikiagoa da."
        | otherwise     = fakt_be_lag x 1

-----------------------------------------------------------------

--2
-- Funtzioa honek x, bm eta bid zenbaki osoak emanda [bm..x]
-- tarteko elementuen biderkadura bider bid itzuliko du emaitza bezala.
-- Biderkadura hori kalkulatzeko [bm..x] tarteko zenbakiak aurreraka
-- zeharkatuko dira, hau da, bm-tik hasi eta x balioraino.
```

```
-- Kasu berezi bezala, x parametroaren balioa 0 baldin bada edo bm
-- balioa x baino handiagoa bada, emaitza bid izango da.
-- Beste aldetik, x parametroaren balioa 0 baino txikiagoa
-- baldin bada edo bm parametroaren balioa 1 baino txikiagoa baldin bada,
-- errore-mezua aurkeztuko da.

-- Bigarren parametroa (hau da, bm) [1..x] tarteko zenbakiak zeharkatzeko
-- erabiliko da eta hirugarren parametroa (hau da, bid) [1..x] tarteko
-- zenbakien biderkadura gordetzeko erabiliko da.

fakt2_be_lag:: Int -> Int -> Int -> Int
fakt2_be_lag x bm bid
        | x < 0                                 = error "Lehenengo datua 0 baino txikiagoa da."
        | bm < 1                                = error "Bigarren datua 1 baino txikiagoa da."
        | x < bm                                = bid
        | otherwise                             = fakt2_be_lag x (bm + 1) (bid * bm)

-----------------------------------------------------------------

-- Funtzio honek 0 baino handiagoa edo berdina den x zenbakiaren
-- faktoriala kalkulatuko du. Beraz, x parametroaren balioa 1 edo
-- handiagoa baldin bada, [1..x] tarteko elementuen biderkadura
-- itzuliko da eta, kasu berezi bezala, x parametroaren balioa 0
-- baldin bada, emaitza 1 izango da.
-- Emandako x balioa 0 baino txikiagoa baldin bada, errore-mezua
-- aurkeztuko da.

-- Murgilketa erabiliko da.

fakt2_be:: Int -> Int
fakt2_be x
        | x < 0         = error "Datua 0 baino txikiagoa da."
        | otherwise     = fakt2_be_lag x 1 1

-----------------------------------------------------------------

--3
-- Funtzio honek t motako zerrenda bat eta zenbaki oso bat emanda,
-- zerrendaren luzera gehi bigarren parametroaren balioa itzuliko du.

-- Bigarren parametro hori berez zenbatzaile bezala erabiltzeko da.

luzera_be_lag:: [t] -> Int -> Int
luzera_be_lag [] zenb = zenb
luzera_be_lag (x:s) zenb = luzera_be_lag s (zenb + 1)

-----------------------------------------------------------------
```

```
-- Funtzio honek t motako elementuz osatutako zerrenda bat emanda,
-- zerrendako elementu-kopurua, hau da, zerrendaren luzera kalkulatuko
-- du. Zerrenda hutsa baldin bada, 0 itzuliko da emaitza bezala.
-- Horretarako murgilketa erabiliko da.

luzera_be:: [t] -> Int
luzera_be s = luzera_be_lag s 0

-----------------------------------------------------------------

--4
-- Funtzio honek t motako elementu bat, t motako zerrenda bat
-- eta Int motako elementu bat emanda, lehenengo parametroak adierazten
-- duen balioa zerrendan zenbat aldiz agertzen den gehi hirugarren
-- parametroaren balioa itzuliko du.

-- Hirugarren parametroa berez agerpen-kopurua zenbatzeko da.

aldiz_be_lag:: (Eq t) => t -> [t] -> Int -> Int
aldiz_be_lag x [] zenb = zenb

aldiz_be_lag x (y:s) zenb
        | x == y                        = aldiz_be_lag x s (zenb + 1)
        | otherwise                     = aldiz_be_lag x s zenb

-----------------------------------------------------------------

-- Funtzio honek t motako elementu bat eta t motako zerrenda bat emanda,
-- elementu hori zerrendan zenbat aldiz agertzen den zenbatuko du.
-- Horretarako murgilketa erabiliko da.

aldiz_be:: (Eq t) => t -> [t] -> Int
aldiz_be x r = aldiz_be_lag x r 0

-----------------------------------------------------------------

--5
-- Funtzio honek osoa eta positiboa den n zenbaki bat, zenbaki osozko
-- zerrenda bat eta beste bi zenbaki oso emanda,
-- bigarren parametrotzat emandako zerrendari dagokionez, n gainditzeko
-- hirugarren parametroari batu beharreko aurrizki laburrenaren luzera
-- kalkulatuko du eta luzera hori gehi laugarren parametroaren balioa
-- itzuliko du. Aurrizki hutsaren, hau da, zerrenda hutsaren batura 0
-- dela kontuan hartu beharko da.

-- Zerrendako elementuak eta hirugarren parametroa batuz n ezin bada
```

```
-- gainditu, -1 itzuliko da emaitza bezala.

-- Hirugarren parametroa berez batura kalkulatuz joateko da eta laugarren
-- parametroa zenbatzaile gisa erabiltzeko da.

gainditu_be_lag:: Int -> [Int] -> Int -> Int -> Int
gainditu_be_lag n [] batura zenb
        | batura > n    = zenb
        | otherwise     = -1

gainditu_be_lag n (x:s) batura zenb
        | batura > n            = zenb
        | otherwise             = gainditu_be_lag n s (x + batura) (zenb + 1)

-----------------------------------------------------------------------

-- Funtzio honek osoa eta positiboa den n zenbaki bat eta zenbaki osozko
-- zerrenda bat emanda,
-- zerrenda horri dagokionez, n balioa gainditzeko adineko batura duen
-- aurrizki laburrenaren luzera kalkulatuko du.

-- Aurrizki hutsaren, hau da, zerrenda hutsaren batura 0 dela kontuan
-- hartu beharko da.
-- Zerrendako elementuak batuz n ezin bada gainditu, -1 itzuliko da
-- emaitza gisa.

gainditu_be:: Int -> [Int] -> Int
gainditu_be n r = gainditu_be_lag n r 0 0

-----------------------------------------------------------------------

--6
-- Funtzio honek osoa den n zenbaki bat, zenbaki osozko zerrenda bat
-- eta beste zenbaki oso bat emanda, lehenengo parametroak (hau
-- da n datuak) adierazten duen balioaren lehenengo agerpenaren posizioa
-- (ezkerretik hasita) gehi hirugarren parametroaren balioa itzuliko du.
-- Kasu berezi gisa, lehenengo
-- parametroak adierazten duen n balioa zerrendan ez bada agertzen,
-- 0 balioa itzuliko da.

-- Hirugarren parametroa berez zenbatzaile gisa erabiltzeko da.


lehenpos_be_lag:: Int -> [Int] -> Int -> Int

lehenpos_be_lag n [] zenb = 0
```

```
lehenpos_be_lag n (x:s) zenb
        | n == x                = 1 + zenb
        | otherwise             = lehenpos_be_lag n s (zenb + 1)

-----------------------------------------------------------------

-- Osoa den n zenbaki bat eta zenbaki osoz osatutako zerrenda bat emanda,
-- n-ren lehenengo agerpenaren posizioa (ezkerretik hasita) itzuliko du
-- funtzio honek. Zenbakia zerrendan ez bada agertzen, 0 balioa itzuliko da.

lehenpos_be:: Int -> [Int] -> Int
lehenpos_be n r = lehenpos_be_lag n r 0

-----------------------------------------------------------------

--7
-- Funtzio honek, t motako bi zerrenda emanda, honako beste bi zerrenda
-- hauek elkartuz lortuko den zerrenda itzuliko du:
--      1. Datu bezala emandako bigarren zerrenda
--      2. Datu bezala emandako lehenengo zerrendatik azkeneko elementua
--         (eskuineko ertzean dagoena) kezduz geratzen den zerrenda
-- Datu bezala emandako lehenengo zerrenda hutsa bada, errore-mezua
-- aurkeztuko da.

-- Bigarren parametroa emaitza (zerrenda berria) gordez joateko da.


ak_be_lag:: [t] -> [t] -> [t]
ak_be_lag [] q = error "La primera lista es vacia"

ak_be_lag (x:s) q
        | hutsa_da s            = q
        | otherwise             = ak_be_lag s (q ++ (x:[]))

-----------------------------------------------------------------

-- t motako elementuz osatutako zerrenda bat emanda, azkeneko elementua
-- (eskuineko ertzean dagoena) kenduz gelditzen den zerrenda itzuliko du
-- funtzio honek. Zerrenda hutsa bada, errore-mezua aurkeztuko da.

ak_be:: [t] -> [t]
ak_be r = ak_be_lag r []

-----------------------------------------------------------------

--8
-- Datu bezala zenbaki osozko hiru zerrenda emanda, honako beste bi
```

```
-- zerrenda hauek elkartuz lortzen den zerrenda
-- itzuliko du funtzio honek:
--      1. Datu bezala emandako hirugarren zerrenda eta
--      2. Datu bezala emandako lehenengo bi zerrendatan posizio berean
--         dauden elementuen baturez osatutako zerrenda

-- Datu bezala emandako lehenengo bi zerrenden luzera desberdina bada,
-- errore-mezua aurkeztuko da.

-- Hirugarren parametroa emaitza (zerrenda berria) gordez joateko da.


bz_be_lag:: [Int] -> [Int] -> [Int] -> [Int]
bz_be_lag [] h q
        | not (hutsa_da h)      = error "Lehenengo bi zerrendek luzera desberdina dute."
        | otherwise             = q

bz_be_lag (x:s) h q
        | (luzera (x:s)) /= (luzera h)  = error "Lehenengo bi zerrendek luzera desberdina dute."
        | otherwise                             = bz_be_lag s (hond h) (q ++ ((x + (leh h)):[]))

-----------------------------------------------------------------

-- Zenbaki osoz osatutako bi zerrenda emanda, posizio berean dauden
-- elementuen baturez osatutako zerrenda itzuliko du funtzio honek.
-- Zerrenden luzera desberdina bada, errore-mezua aurkeztuko da.

bz_be:: [Int] -> [Int] -> [Int]
bz_be r u = bz_be_lag r u []

-----------------------------------------------------------------

--9
-- Datu bezala n zenbaki oso bat eta zenbaki osozko hiru zerrenda emanda,
-- honako beste lau zerrenda hauek elkartuz lortzen den zerrenda
-- itzuliko du:
--      1. Hirugarren parametro bezala emandako zerrenda.
--      2. Bigarren parametro bezala emandako zerrendan n baino
--         txikiagoak edo berdinak diren elementuez osatutako zerrenda.
--         Elementu horiek beraien arteko hasierako ordenari eutsiko
--         diote.
--      3. Laugarren parametro bezala emandako zerrenda.
--      4. Bigarren parametro bezala emandako zerrendan n baino
--         handiagoak diren elementuez osatutako zerrenda. Elementu
--         horiek beraien arteko hasierako ordenari eutsiko diote.

-- Bigarren parametro bezala emandako zerrenda hutsa baldin bada,
```

```
-- hirugarren eta laugarren parametroak elkartuz lortzen den zerrenda
-- itzuliko da.

-- Bigarren zerrenda (hirugarren parametroa) n baino txikiagoak edo
-- berdinak diren elementuak gordetzeko erabiliko da eta hirugarren
-- zerrenda (laugarren parametroa) n baino handiagoak diren elementuak
-- gordetzeko erabiliko da.


th_be_lag:: Int -> [Int] -> [Int] -> [Int] -> [Int]
th_be_lag n [] h q = h ++ q

th_be_lag n (x:s) h q
        | x <= n                = th_be_lag n s (h ++ (x:[])) q
        | otherwise             = th_be_lag n s h (q ++ (x:[]))

-----------------------------------------------------------------

-- Osoa den n zenbaki bat eta zenbaki osozko zerrenda bat emanda, n baino
-- txikiagoak edo berdinak direnak ezkerraldean eta n baino handiagoak
-- direnak eskuinaldean ipiniz osatutako zerrenda itzuliko du funtzio honek.
-- Txikiagoak edo berdinak direnek beraien artean hasierako ordenari
-- eutsiko diote eta handiagoak direnek ere beraien artean
-- hasierako ordenari eutsiko diote. Gerta daiteke n balioa
-- zerrendan ez agertzea. Datu bezala emandako
-- zerrenda hutsa bada, zerrenda hutsa aurkeztuko da.

th_be:: Int -> [Int] -> [Int]
th_be n r = th_be_lag n r [] []

-----------------------------------------------------------------
```