

3. gaia: Errekurtsibitatea

Bertol Arrieta eta **Koldo Gojenolaren** gardenkiak



Errekurtsibitatea

- Metodo (eragiketa) bat errekurtsiboa da, baldin eta bere buruari deitzen badio
- Diseinu errekurtsiboak oinarritzko kasuak eta kasu orokorrak hartu behar ditu kontuan

Adibidea: batukaria

$$\sum_{k=1}^N k = N + \sum_{k=1}^{N-1} k$$

```
public int suma (int num) {  
    int result;  
    if (num == 1)  
        result = 1;  
    else  
        result = num + suma (num-1);  
    return result;  
}
```

Errekurtsioa erabiltzeko klabeak

- Programa errekurtsibo batean derrigorrez egon behar da kasu ez errekurtsibo bat (gutxienez bat)
 - Kasu ez-errekurtsiboak (kasu nabariak): Emaizta dei errekurtsiborik egin gabe lortzen dutenak
 - Kasu errekurtsiboek kasu nabarietara hurbiltzeko balio behar dute, hau da, programak bukatzera jo behar du.
- Kasu nabaririk ez duen programa errekurtsibo baten exekuzioa ez da inoiz amaituko

Azpiprograma errekurtsiboen diseinurako pausoak

1. Espezifikazioa / parametrizazioa
2. Kasu nabarien (ez-errekurtsiboen) azterketa
3. Kasu errekurtsiboen (orokorren) azterketa
4. Algoritmoaren idazketa
5. Bukaeraren azterketa
6. Inplementazioa

1. Espezifikazioa / parametrizazioa

- 1.1. Egin beharreko azpiprogramaren xehetasunak argitu
→ Aurrebaldintzak, postbaldintzak.
- 1.2. Azpiprogramaren parametroak eta parametro horien mota finkatu (sarrera eta irteerakoak) → Kontuan izan dei errekurtsiboa parametro horien arabera egingo dela.
- 1.3. Parametroek bete behar dituzten mugak eta murrizketak adierazi.
- 1.4. Hasierako deia definitu
→ bereziki garrantzitsua da parametroren batek hasierako balio bat hartu behar badu.

2. Kasu nabarien azterketa

1. Zehaztu zein diren kasu nabariak
(**gutxienez kasu nabari bat !!**)
2. Kasu nabari horietarako adierazi zein den eman beharreko emaitza

3. Kasu orokorren azterketa

1. Zehaztu zein diren kasu orokorrak:
parametroen zein balioetarako egin behar diren dei errekurtsiboak.
2. Kasu orokorren tratamendua
 - Kasu nabarien eta orokorren artean gerta litezkeen kasu guztiak bildu behar dira.
 - **Dei errekurtsiboen parametro errealek kasu nabarietara hurbiltzeko balio behar dute.**
 - Dei errekurtsiboetan erabiltzen diren parametroek bat etorri behar dute parametro formalekin (parametrizazioan definitutakoak)
→ motan eta kopuruan

4. Bukaeraren azterketa

Egiaztatu dei errekurtsiboetan parametroak kasu nabarietara hurbiltzen doazela, eta, beraz, beti bukaerara iritsiko garela.

5. Algoritmoaren idazketa

- Algoritmoa idazten da kasu desberdin guztien definizioak bilduz, txukunduz eta, ahal bada, trinkotuz.
- Behar baldin bada, datu-egituraren diseinua ere pentsatu

6. Inplementazioa

1. Programazio-lengoaian idatzi algoritmoa
2. Datu-egituren xehetasunak finkatu
3. Eraginkortasunean irabazteko aldaketak egin

Iterazioaren eskema

algoritmo Iterazioa

hasiera

Hartu_lehenengo_osagaia (Osag)

bitartean ez (Azkeneko_osagaia_da (Osag))
egin

Tratatu_osagaia (Osag)

Hartu_hurrengo_osagaia (Osag)

ambitartean

Tratatu_osagaia (Osag)

amaia

Errekurtsioaren eskema (1)

algoritmo Errekurtsioa1

hasiera

baldin *bukaera-baldintza* orduan

azkeneko pausuak burutu

bestela

ekintza orokorrak egin

hurbildu bukaera-baldintzara

deitu berriro prozesuari parametro berriekin

ambaldin

amaia

Errekurtsioaren eskema (2)

algoritmo Errekurtsioa2

hasiera

baldin ez (*bukaera-baldintza*) **orduan**

ekintza orokorrak egin

hurbildu bukaera-baldintzara

deitu berriro prozesuari parametro berriekin

ambaldin

amaia

Hanoiko dorreak



Hanoiko dorreak

// k diska mugituko dira x dorretik y dorrera
// z dorrea laguntza moduan erabiliz.
// mugitu(x,y) = “diska bat mugitu x dorretik y dorrera”

```
public static void hanoi(int k, int x, int y, int z){  
    if (k==1)  
        mueve(x,y);  
    else{  
        hanoi(k-1, x, z, y);  
        mueve(x,y);  
        hanoi(k-1, z, y, x);  
    }  
}
```


Adibidez: Bilaketa dikotomikoa

(Bilaketa bitarra ere deitua)

- Array ordenatu bat izanik:

// taula[i..f] osokoen array ordenatu bat da (txikitik handira)

// x taula[i..f]n baldin badago, bere indizea itzultzen du.

// Bestela, -1 itzultzen du

```
public int bilaketaDikotomikoa(int i, int f, long x){  
    if ( i>f)  
        return -1; //x ez dago taulan  
    else{  
        int erdia = (i+f)/2;  
        if (taula[erdia]==x)  
            return erdia;  
        else if (taula[erdia]>x)  
            return bilaketaDikotomikoa(i, erdia-1, x);  
        else  
            return bilaketaDikotomikoa(erdia+1, f, x);  
    }  
}
```

Algoritmo errekurtsiboen analisia

- Kostu-funtzioaren kalkulua:
 - Batukaria:
 - $f(n) = O(1) + f(n-1)$
 - Bilaketa dikotomikoa:
 - $h(n) = O(1) + h(n/2)$

Irakurgaiak

- [Lewis, Chase 2010]
 - 7. kapitulua