

# Sarrera

---

- Zein datu-egitura erabili dituzu programazioa-l irakasgaian?
- Zein Java klase erabili dituzu zure programetan?
- Zein da egin duzun programa zailena?
- Zein izan zen zailtasun handiena?

# Zertarako datu-egiturak?

**Computer Science Curriculum 2008:**

**Association for Computing Machinery / IEEE Computer Society**

<http://www.acm.org/education/curricula/ComputerScience2008.pdf>

*The thing that we can't afford to do [...] is teach candidates how to think critically, to be effective problem solvers, and to **have basic mastery of programming languages, data structures, algorithms**, concurrency, networking, computer architecture, and discrete math / probability / statistics. **I can't begin to emphasize the importance of algorithms and data structures to the work we do here** [...]. With multi-terabyte disks, bigger broadband pipes, etc. on the way, the big data problems that demand these skills [...] are quickly going to be in need in a huge number of programming contexts.*

# Zertarako datu-egiturak?

- Datuak **modu egokian gordetzeko**
  - Nola antolatzen dituzu datuak memorian?
  - Antolaketa hori egokia al da 100 elementurentzat? Eta 1000 elementurentzat? Eta milioi bat elementurentzat? Datuen kudeaketa errazten al du?
  - Elementuak gehitzea eta kentzea modu efizientean egiten al da?
  - Eta elementu baten bilaketa?
- “Bizi dugun mundua” **eredu batean errepresentatzeko**

# Egiturekin erabiltzen diren algoritmo ohikoenak

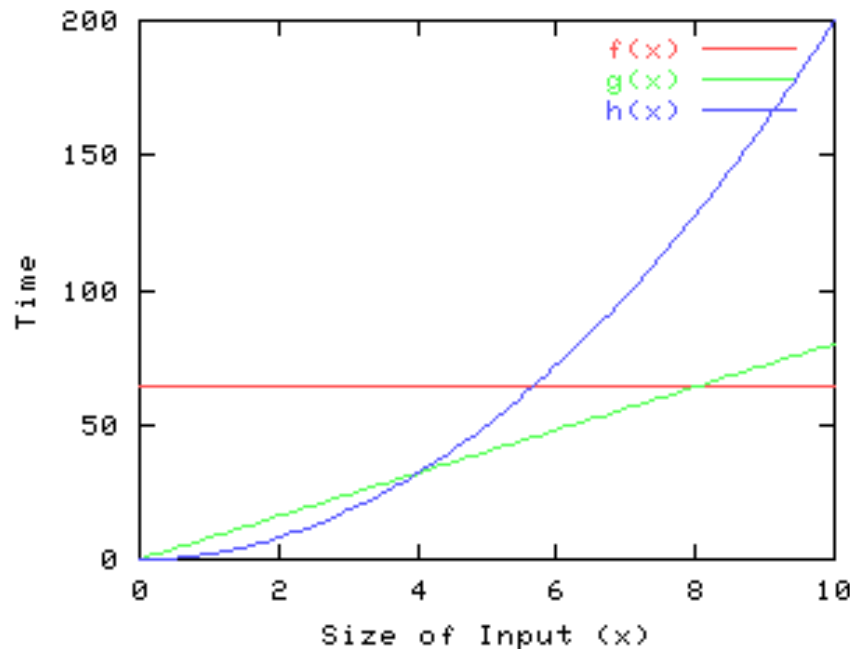
- Egitura bat **sortu**
- Elementu berri bat **txertatu**
- Elementu bat **bilatu**
- Elementu bat **kendu**
- Egitura bateko elementu **guztiak korritu**

# Adibidea: Zereginen ilara, lehentasunekin

- Zeregin bat txertatzea ilaran, lehenetasun konkretu batekin
- Zein da lehenetasun handieneko zeregina?
- Lehenetasun handieneko zeregina kentzea ilaratik
- Zeregin konkretu bati lehenetasuna aldatzea

# Algoritmoen konplexutasunaren analisia

- Nola neurtzen dugu algoritmoen efizientzia?
  - Kostu-funtzioekin (denborakoak edo espaziokoak)



# ¿Garrantzitsua al da algoritmoen efizientzia?

Fibonacciren seriea: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

```
function Fib (N: Natural) return Natural is
begin
  if N <= 1 then return N;
  else return Fib(N-1) + Fib(N-2);
  end if;
end
```



# Zenbat batuketa egiten ditu Fib(N) deiak?

- $\text{batuketak}(N) = 1 + \text{batuketak}(N-1) + \text{batuketak}(N-2)$   
 $\forall N > 1$
- $\text{batuketak}(N-1) = 1 + \text{batuketak}(N-2) + \text{batuketak}(N-3)$
- $\text{batuketak}(N) > 2 \times \text{batuketak}(N-2)$
- $2 \times \text{batuketak}(N-2) > 2 \times 2 \times \text{batuketak}(N-4) > \dots > 2^k \times \text{batuketak}(N-2k)$
- $\text{batuketak}(N) > 2^{N/2} = (\sqrt{2})^N$

`function` Fib (N: Natural) `return` Natural `is`

`begin`

`if` N<=1 `then return` N;


`else return` Fib(N-1) + Fib(N-2);

`end if;`

`end`

- Probatu zenbaki hauekin: 30, 40, 43, 45, 50, 55

# Denboren taula

N	$\sim 2^{n/2}$	$2^{n/2} \times 10^{-9}$ segundu
50	33.554.432	33 mseg
60	$1,1 \times 10^9$	1 seg
70	$3,4 \times 10^{10}$	34 seg
80	$1,1 \times 10^{12}$	18 min
90	$3,5 \times 10^{13}$	9 ordu
100	$1,1 \times 10^{15}$	
110	$3,6 \times 10^{16}$	
120	$1,2 \times 10^{18}$	
130	$3,6 \times 10^{19}$	

# Irakasgaiaren ikuspuntua

- DMAen eta hauek inplementatzen dituzten datu-egituren eta algoritmoen ikasketa.
- Konplexutasunaren analisisia.
- Inplementazioak Java lengoaiarekin.

# Datu mota abstraktuak (DMA)

- **Datuen eta** datu horien gainean egin daitezkeen **eragiteken** espezifikazioa.
- **Abstraktua**: **Zein** eragiketa egin daitezkeen azpimarratzen da, eta ez **nola** egin daitezkeen.
- Izan ditzakeen inplementazio posibleekiko independentea da DMAa.

## DMAen abantailak:

*Gogoratu Programazioaren metodologia irakasgaian ikasitakoa!*

- ➡ Abstrakzio maila handiagoa.
- ➡ DMAa darabilten programen eta DMAa inplementatzen dutenen egiaztapen independentea.
- ➡ DMAaren inplementazioan aldaketak, DMAa darabilten programetan eraginik izan gabe.

## Ekuzazio-espezifikazioa (algebraikoa):

- Signatura : datu-motak eta bere eragiketak definitzen ditu
- Ekuzazioak : signaturako eragiketen eragina deskribatzen du

*(Gogoratu Programazioaren  
Metodologia irakasgaia)*

Adibidea: mota kontagailua  
eragiketak

hasieratu:  $\rightarrow$  kontagailua

inkr, dekr : kontagailua  $\rightarrow$  kontagailua

ekuzazioak

dekr (inkr (x)) = x

# DMAen adibideak

- Bildumak:
  - Elementuen listak
  - Elementuen pilak
  - Elementuen multzoak
  - .....
- Arlo batera lotutakoak:
  - Banku-kontua
  - Espresio aritmetikoak
  - Hiztegia
  - .....

# Objektuei zuzendutako programazioa

- vs. prozedurei zuzendutakoa
- Objektu batean aldagaiak eta prozedurak kapsulatuta daude
- **Klase** bat objektuak definitzeko mekanismo bat da

*(Gogoratu lehen kurtsoko irakasgaiak)*



---

# Tips to be a competitive programmer

- Competitive Programming 3 [Steven & Felix Halim 2013]:
  - Type code faster!
  - Quickly identify problem types
  - Do algorithm analysis
  - Master programming languages
  - Master the art of testing code
  - Practice and more practice
  - Team work

# Adibidea: Klasearen definizioa

```
class BankuKontu
```

```
{
```

```
    private double zenbatekoa; // kontuan dagoen diru kopurua
```

```
    public BankuKontu(double hasierakoKopurua) // ERAIKITZAILEA  
    { zenbatekoa = hasierakoKopurua; }
```

```
    public void sartu(double kopurua) // dirua sartu  
    { zenbatekoa = zenbatekoa + kopurua; }
```

```
    public void atera(double kopurua) // dirua atera  
    { zenbatekoa = zenbatekoa - kopurua; }
```

```
    public void display() // zenbatekoa erakutsi  
    { System.out.println("zenbatekoa=" + zenbatekoa); }
```

```
} // end class BankuKontu
```

Atributua

Klasearen EGOERA  
definitzen du

Metodoak

Klasearen PORTAERA  
definitzen du

# Adibidea: klasearen erabilera

```
class ApliBanku
{
    public static void main(String[] args)
    {
        BankuKontu bk1 = new BankuKontu(100.00); // kontu bat sortu

        System.out.print("Eragiketen aurretik kontuan dagoen diru kopurua: ");
        bk1.display();                          // zenbatekoa erakutsi

        bk1.sartu(74.35);                        // dirua sartu
        bk1.atera(20.00);                       // dirua atera

        System.out.print("Eragiketen ondoren kontuan dagoen diru kopurua: ");
        bk1.display();                          // zenbatekoa erakutsi
    } // end main()
} // end class ApliBanku
```

# Kontu hauetan sakontzeko irakurketa

- [Lewis & Chase 2010]
  - 2. kapitulua
- *Algorithms, 4th Edition*,  
Robert Sedgewick and Kevin Wayne  
<http://algs4.cs.princeton.edu/home/>