M1 ENJMIN - MJV102

Conception et développement informatique

20-23 septembre 2021

Pierre Cubaud cubaud @ cnam.fr
Viviane Gal gal @ cnam.fr

le cnam
École nationale du jeu et des médias interactifs numériques enjmin

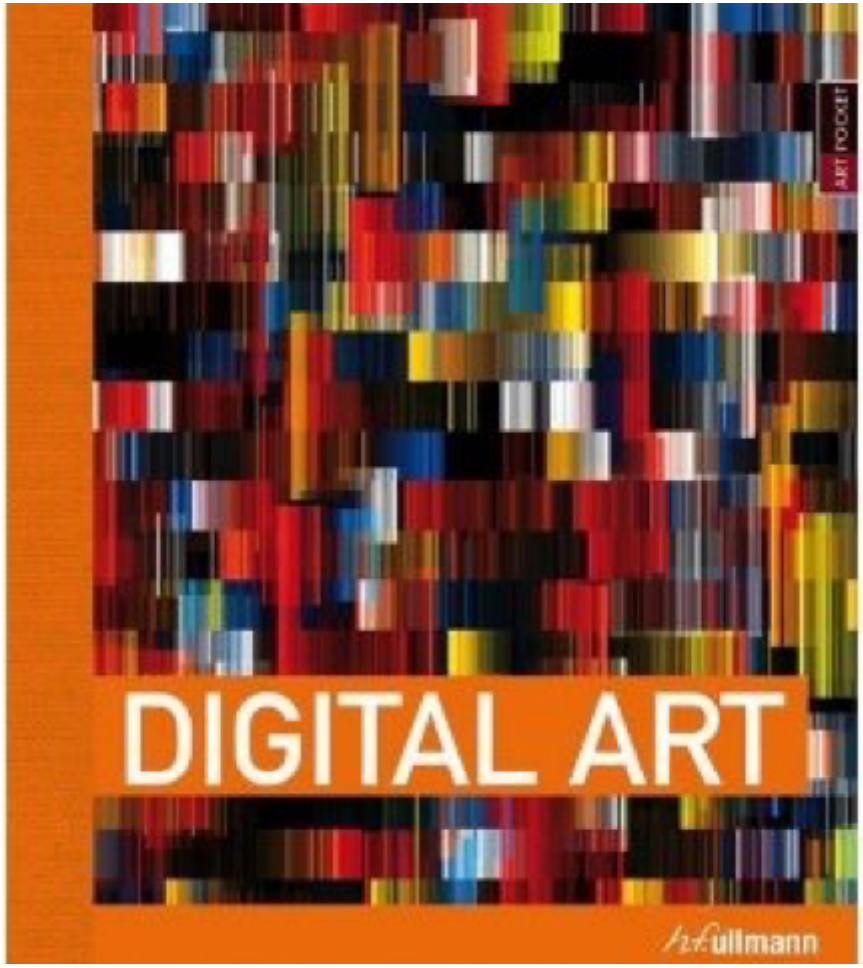
Support de cours disponible sur :

http://cedric.cnam.fr/~cubaud/mjv102.pdf

1. Programmer pour libérer sa créativité et conquérir le monde

Soyez un de ces hommes nouveaux qui savent parler aux ordinateurs.

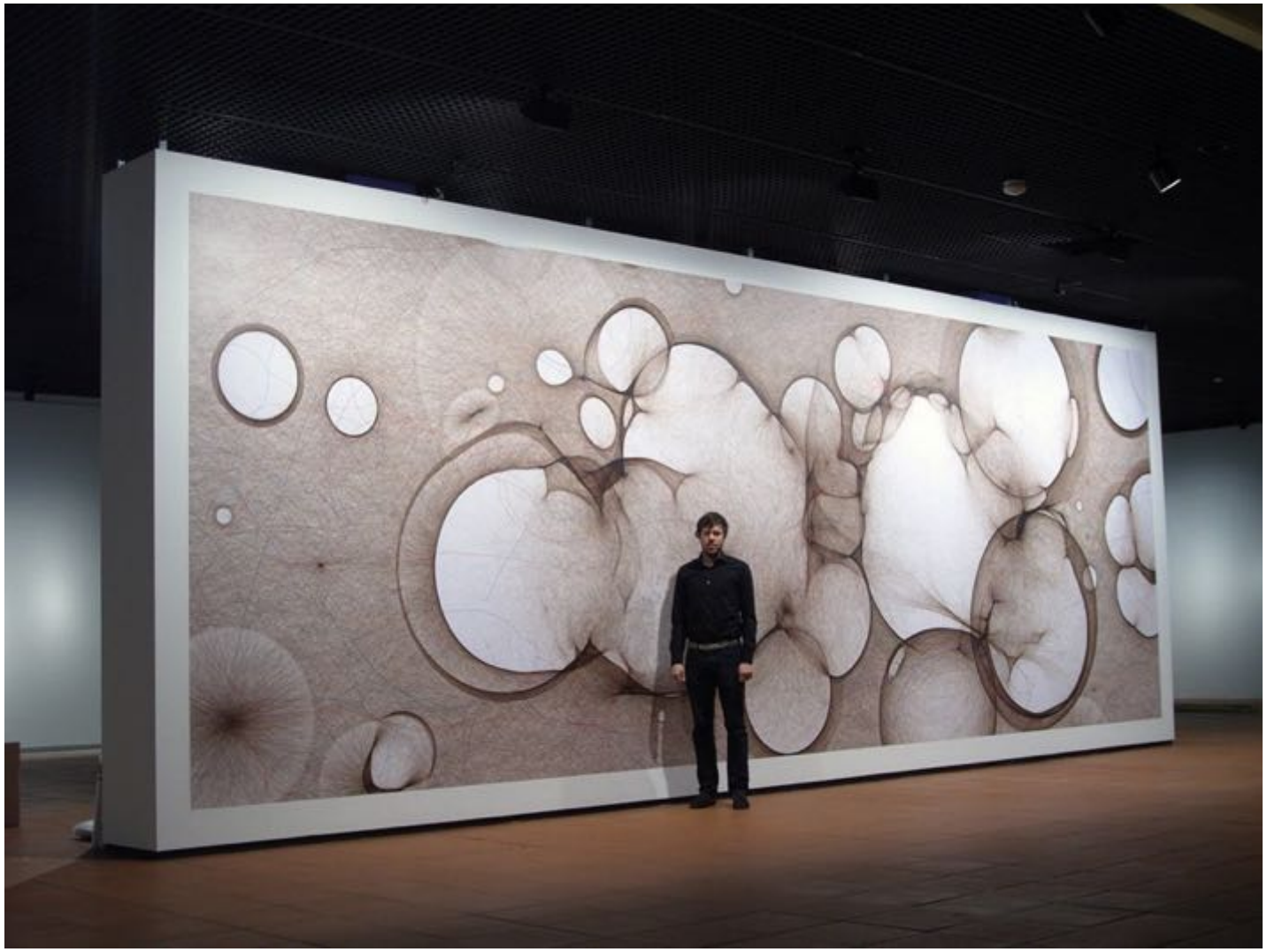
Pour "parler" à un calculateur électronique, il faut un langage spécial. Les programmeurs le connaissent. Voilà pourquoi le programmeur est un spécialiste



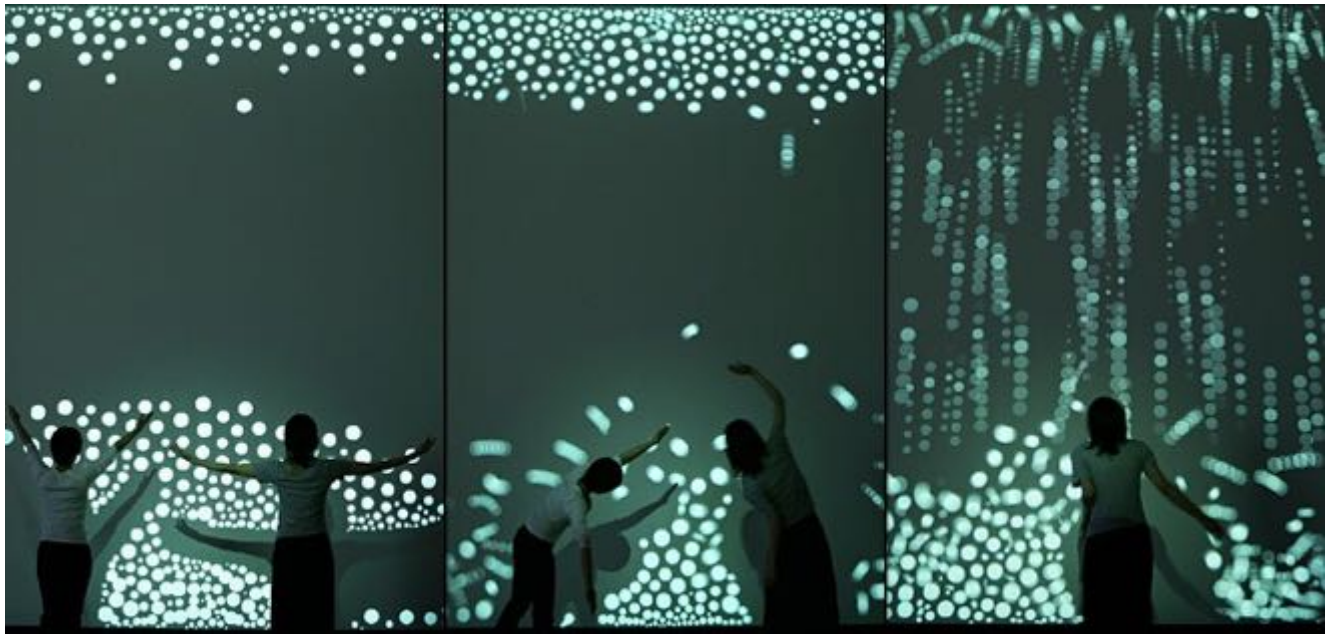
2009 editions h.f. ullmann
<10€

Robert SILVERS, Photomosaic, 1995

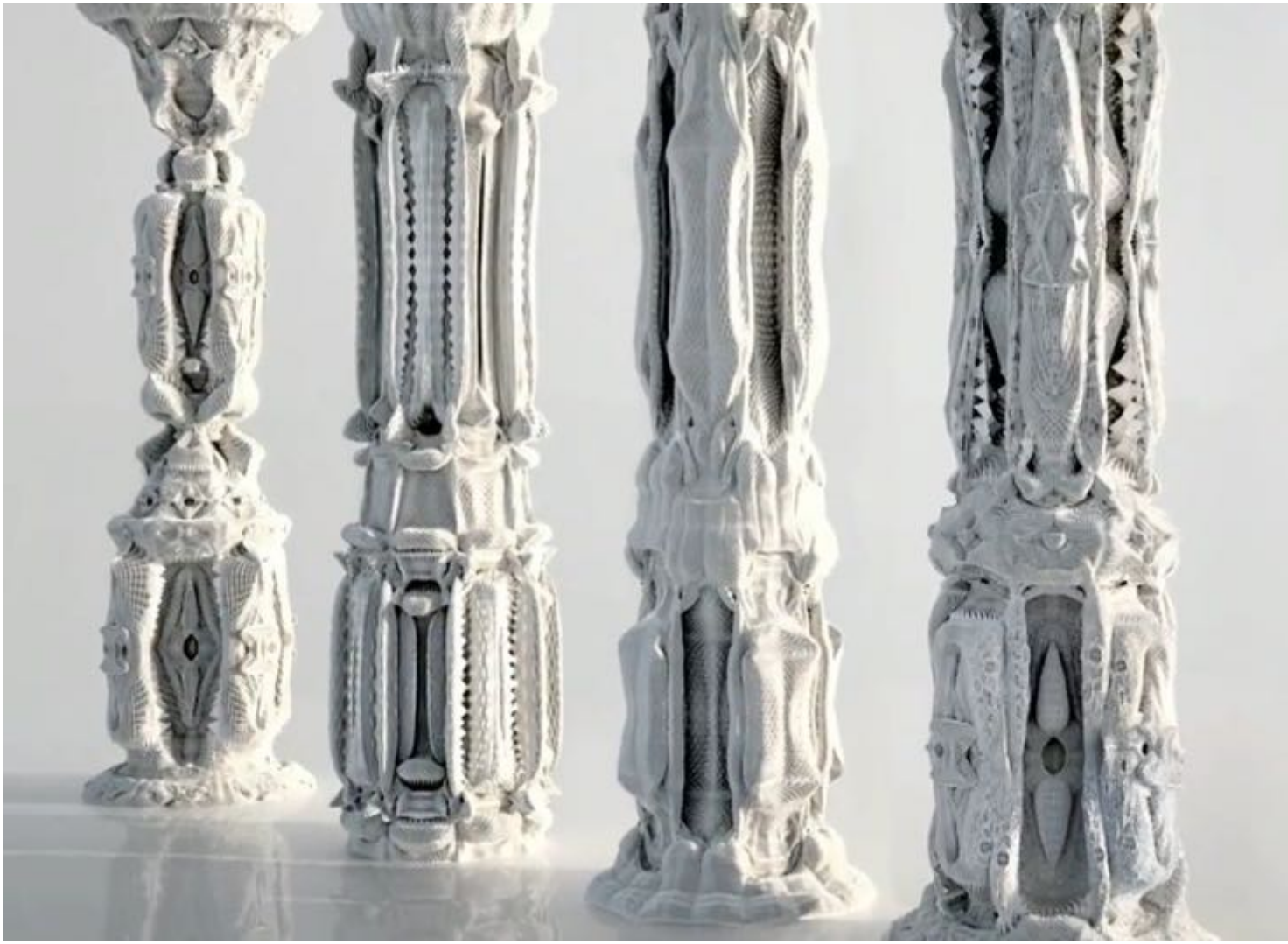
Jonathan PUCKEY
"Delaunay raster" 2008



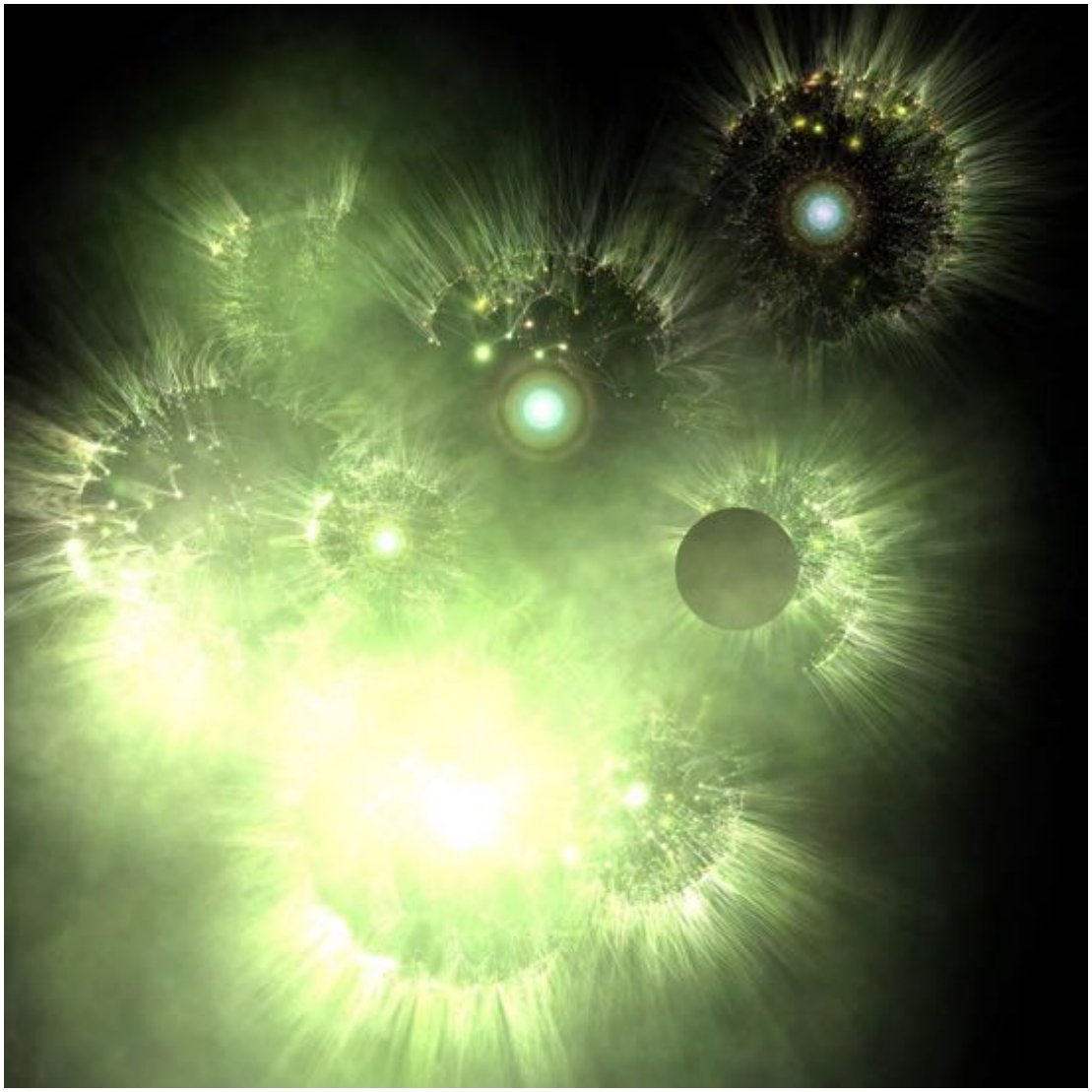
Eno HENZE
http://enohenze.de/



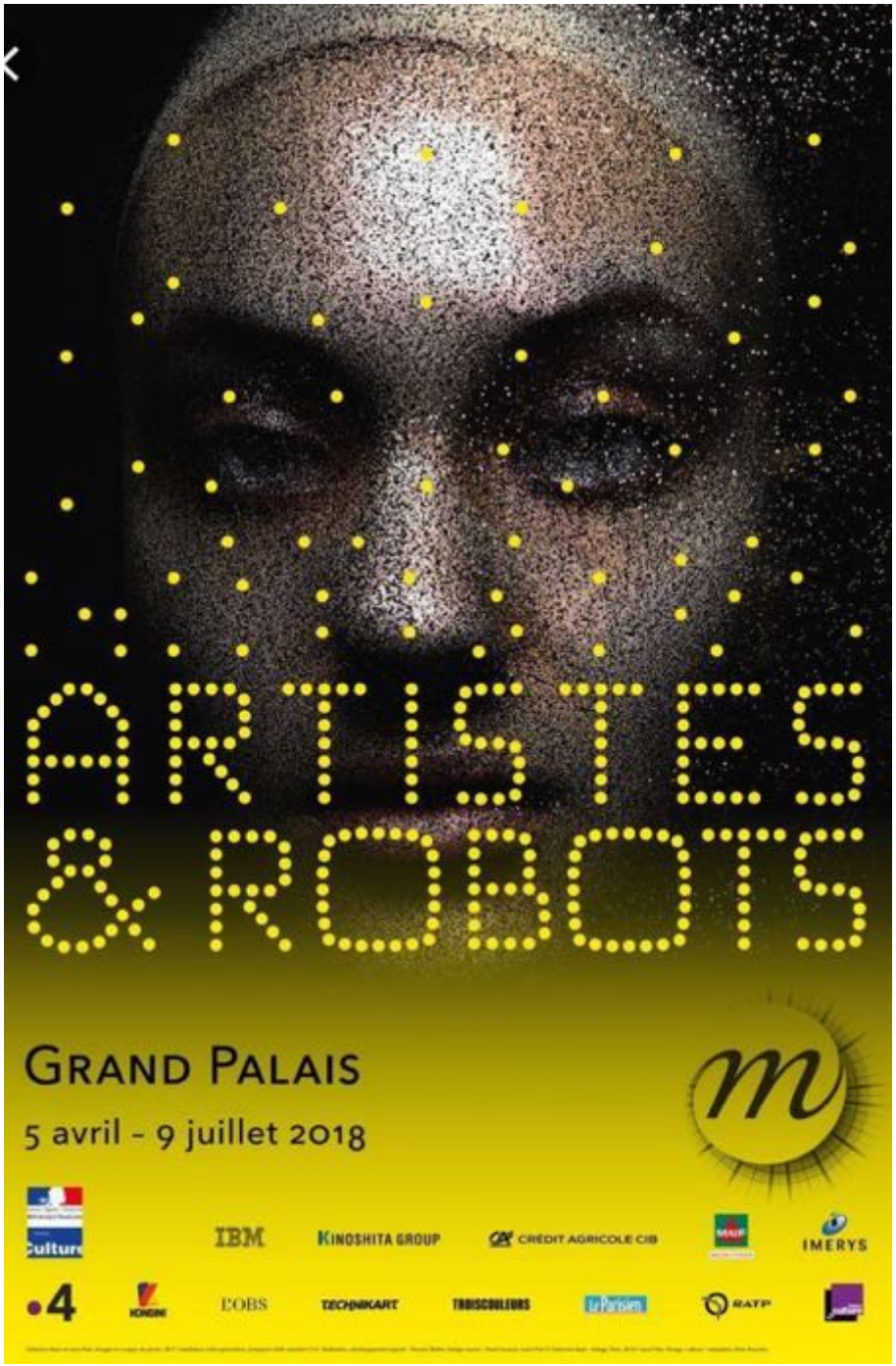
Golan LEVIN "reface" & "footfalls"
http://www.flong.com/

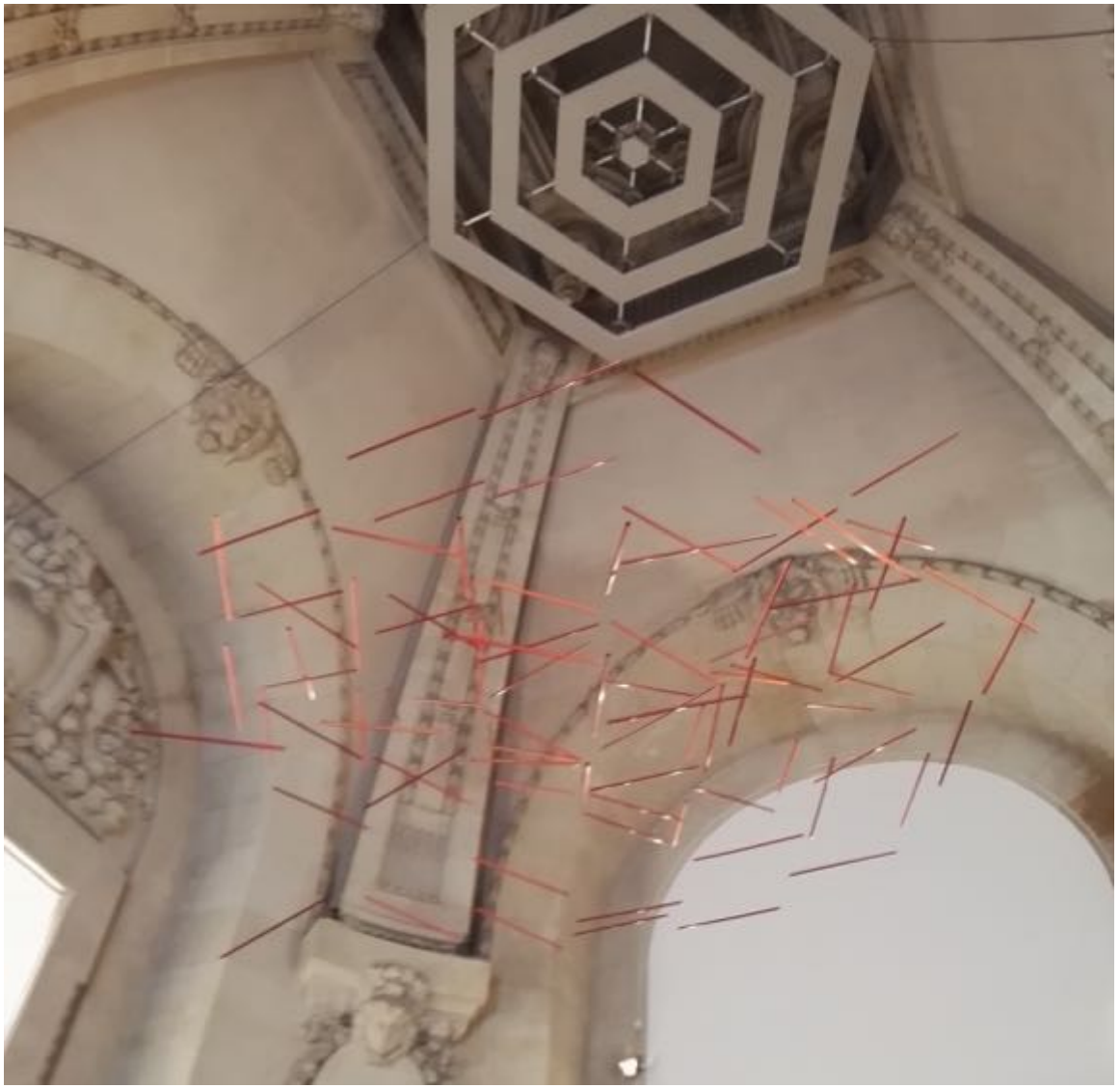


Michael HANSMEYER
http://www.michael-hansmeyer.com/

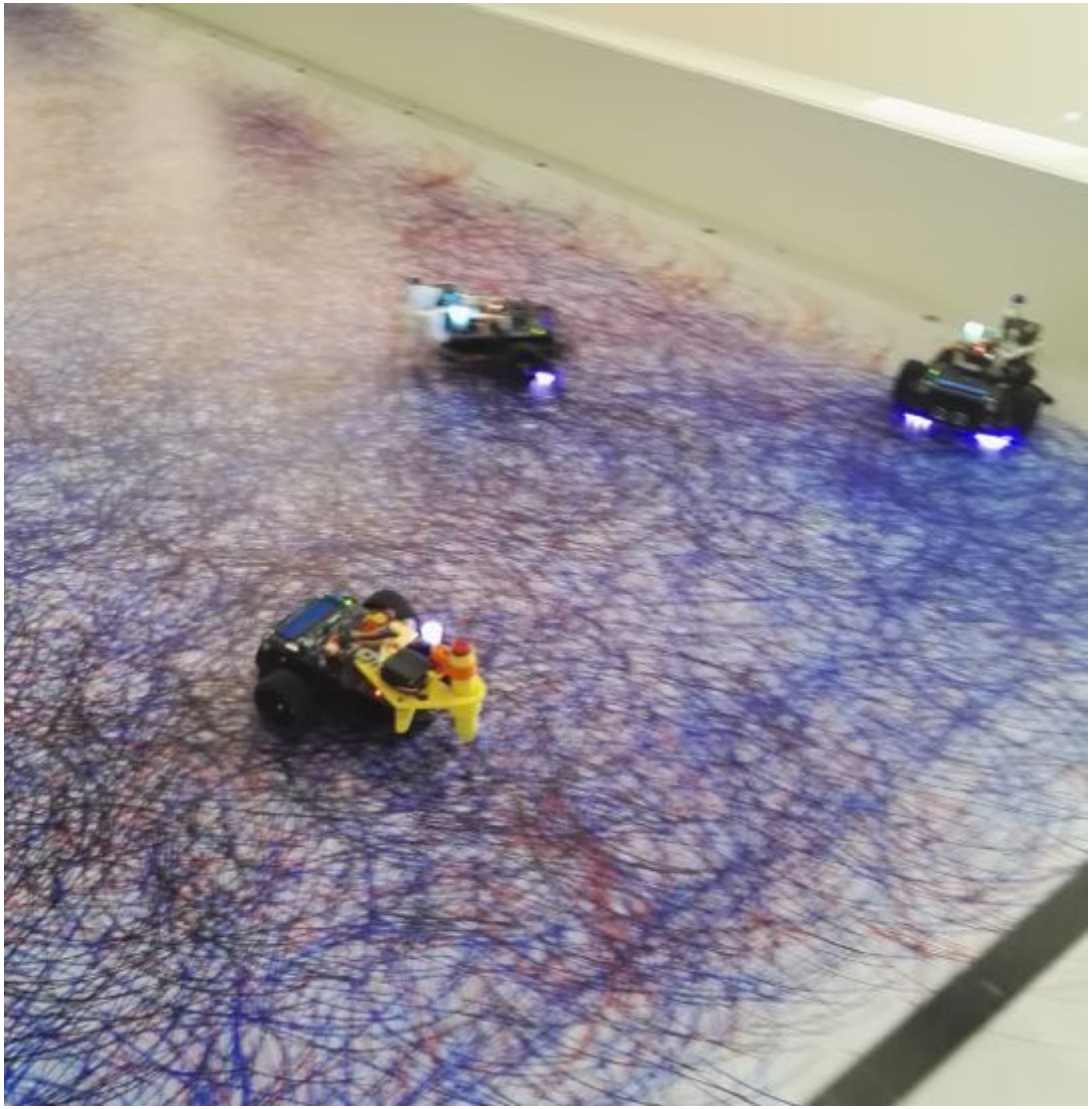


http://roberthodgin.com/


ARTISTES
&ROBOTS
GRAND PALAIS
5 avril - 9 juillet 2018
IBM
KINOSHITA GROUP
CRÉDIT AGRICOLE CIB
IMERYS
•4
ICHIGIN
TECHNIKART
TROISCOULEURS
RATP



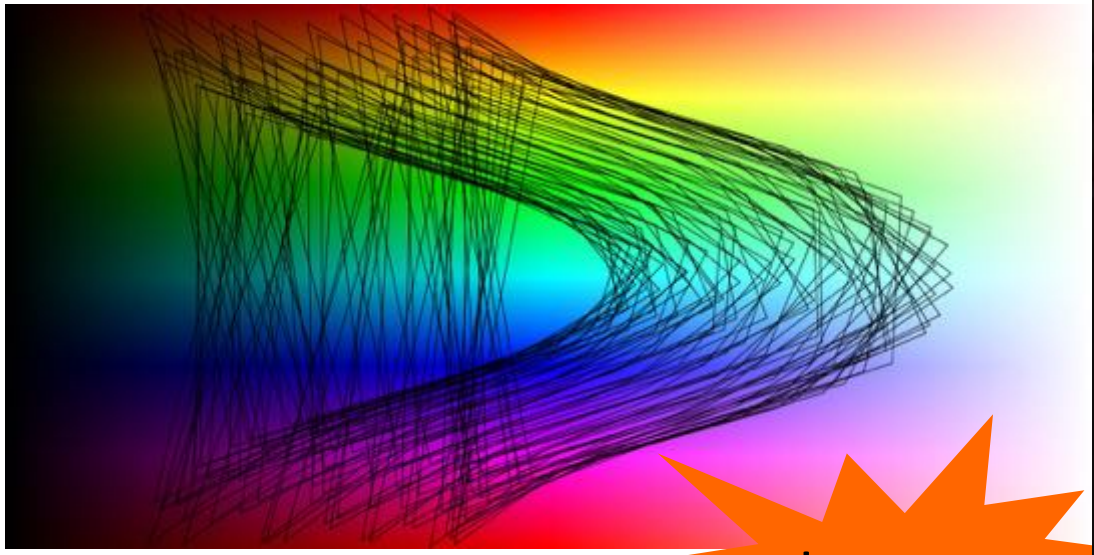
Elias Crespin – Grand HexaNet



Leonel Moura – Robot Art

Plus modestement :

installation P. Cubaud pour le photographe Thomas Paquet (sept-nov 2021)

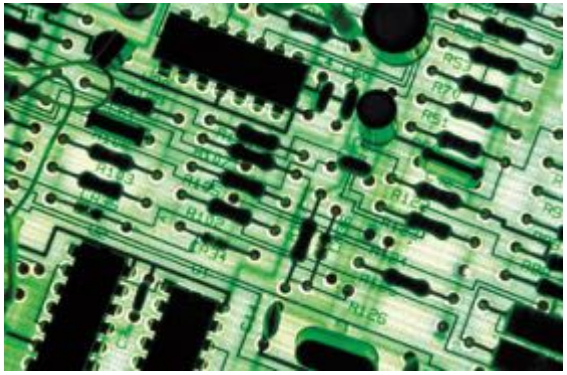


demo

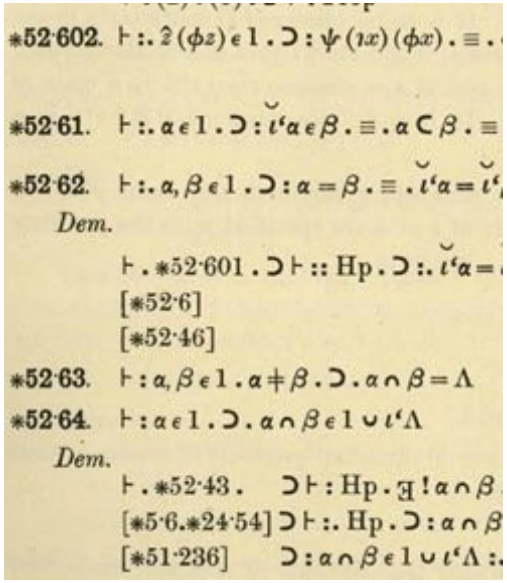
https://bigaignon.fr/thomas-paquet

Programmer c’est dur ?

le programmeur



l'ordinateur :
- fonctionnement ?
- potentiel ?
- contraintes ?



l'application :
- spécifications ?
- codage ?
- vérification ?

2012 Eyrolles 19€
pdf en ligne gratuit à l'INRIA

COMPUTER SCIENCE

Unplugged

L'informatique sans ordinateur

Programme d'activités d'éveil
pour les élèves à partir de l'école primaire

Créé par
Tim Bell, Ian H. Witten et Mike Fellows

Adapté à l'utilisation en classe par
Robyn Adams et Jane McKenzie

Illustré par Matt Powell

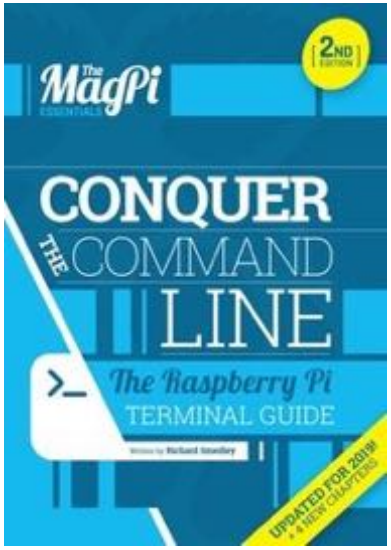
Version française coordonnée par l'équipe d'Interstices
(http://interstices.info)

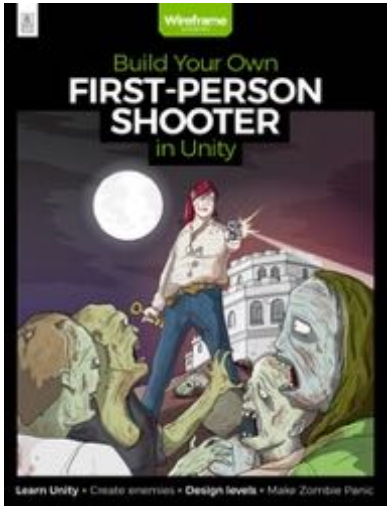
Septembre 2009 - 2e éd. juin 2014

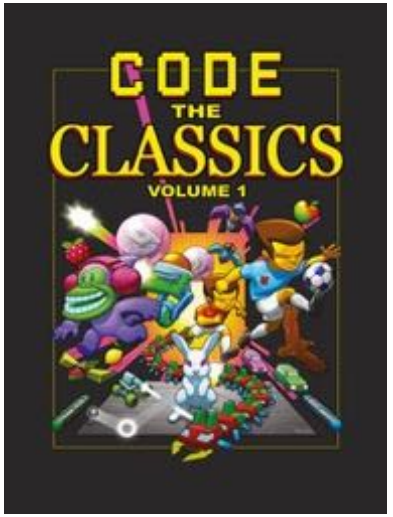
pdf
en ligne

+ la communauté Raspberry Pi

https://www.raspberrypi.org







(au passage)

Sir Clive Sinclair, 1940-2021

17th Sep 2021 Liz Upton 22 comments

It's an incredibly sad day for the British computing industry.

We're always going to be very grateful to Sir Clive for being one of the founding fathers of the UK home computing boom that helped so many of us at Raspberry Pi get hooked on programming as kids.

He was someone from whom the business behind Raspberry Pi has drawn great inspiration. He'll be very sadly missed.

et les cours en ligne !

MOOC médias interactifs (2014-19)
https://www.fun-mooc.fr
+20K inscrits au total

Suite du cours :

- l’ordinateur
- l’environnement Processing
- l'itération
- l'interaction
- les objets

du TP en 2 groupes
(débutants ≠ initiés)

+ des exos libres plus durs

- conclusion de l’atelier (+ evaluation)

2. L’ordinateur

1.**ORDINATEUR, TRICE** [ɔRdinatœR, tRIS] adj. et n. m. — 1491 ; lat. *ordinator, trix* **1.** DIDACT. Qui ordonne, met en ordre. *Cause ordinatrice.* **2.** N. m. RELIG. Celui qui confère un ordre ecclésiastique. ⇒ **ordinant.**

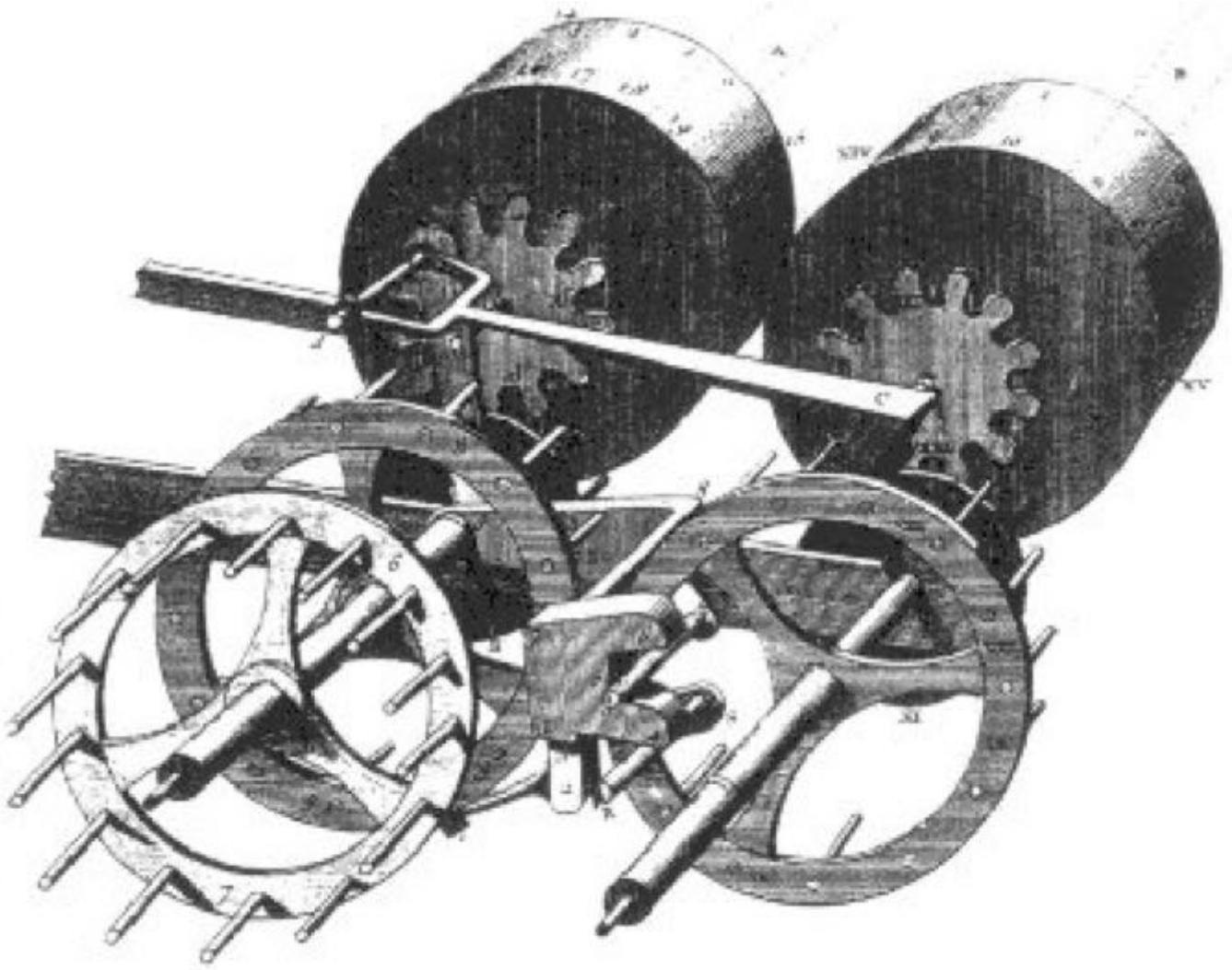
2.**ORDINATEUR** [ɔRdinatœR] n. m. — 1951 ; du lat. *ordo, ordinis* → ordre* (encadré) ; remplaçant l'anglic. *computer* → compter* (encadré) ♦ INFORM. Machine électronique de traitement numérique de l'information, exécutant à grande vitesse les instructions d'un programme enregistré. *Le clavier, la souris, l'écran, la console, la mémoire, l'unité de traitement d'un ordinateur.* ⇒ **matériel.** *Le compilateur, le langage, le progiciel, le système d'exploitation d'un ordinateur.* ⇒ **logiciel.** *Ordinateur frontal. Vitesse d'un ordinateur* (⇒ **inférence**). *Ordinateur spécialisé* (⇒ 2. **calculateur, supercalculateur**), *individuel, de bureau* (⇒ **micro-ordinateur,** 2. **P. C.** ; aussi **mini-ordinateur**). *Ordinateur portable. Travailler sur ordinateur. Conception, enseignement assisté* par ordinateur.* ⇒ **informatique.** — ABRÉV. FAM. ORDI. *Brancher des ordis en réseau.*

ORDINATION [ɔRdinasjɔ̃] n. f. — 1190 ; lat. chrét. *ordinatio* → ordre* (encadré) **1.** LITURG. CATHOL. Acte par lequel es-

Pr. Jacques Perret, 16/5/55, pour IBM France (elec. data processing machines)

Machine de Pascal (ca 1645) – musée des arts et métiers

999999 + 1 = 000000



Pascaline "nature et découverte"

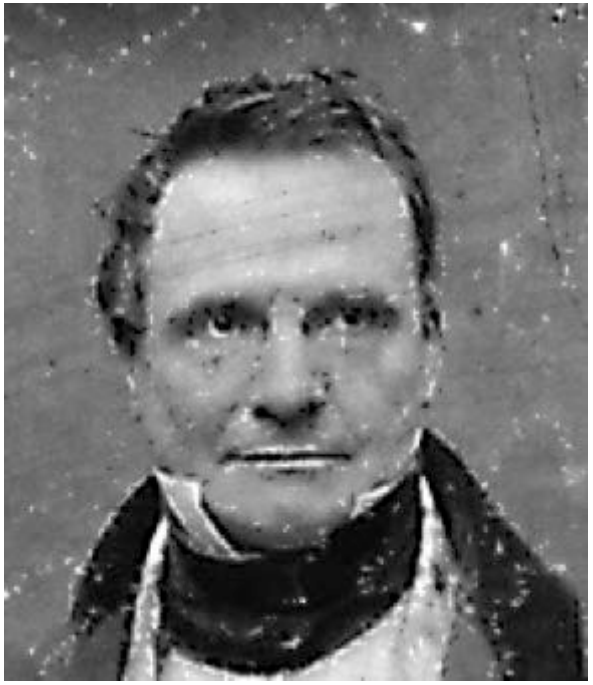
en ce moment en solde <10€

Charles Babbage (1791-1871)

TABLE I.

	N	Logarith. Hyp.
8042	541	6.29341.92788.46481.52157
0488	542	6.29526.60014.39646.20951
8731	543	6.29710.93199.33635.43823
0696	544	6.29894.92468.55942.62734
9672	545	6.30078.57946.63244.07498
6639	546	6.30261.89757.44905.04197
9073	547	6.30444.88024.21981.20563
7700	548	6.30627.52869.48015.53415
2546	549	6.30809.84415.09530.63154

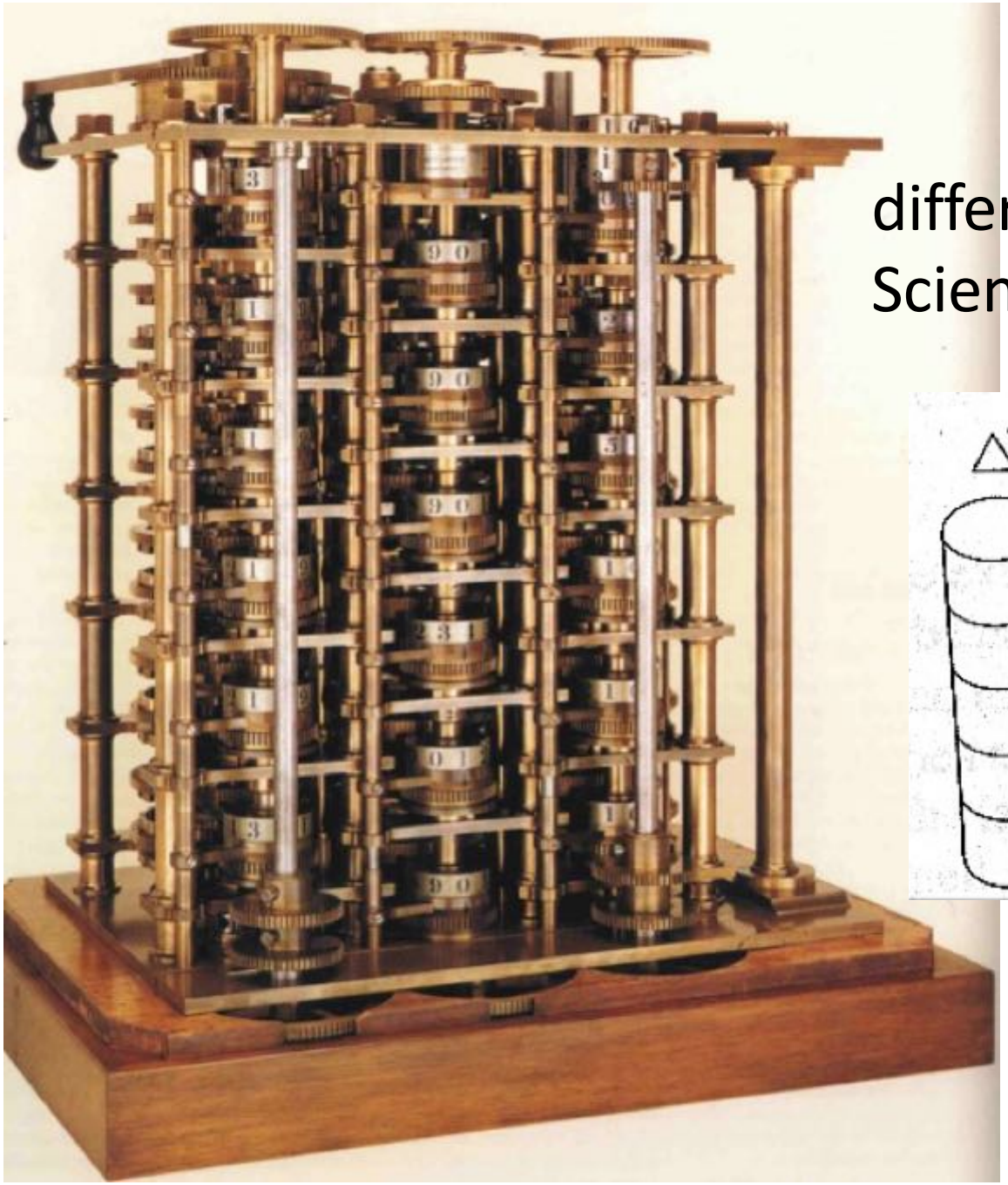
33935.



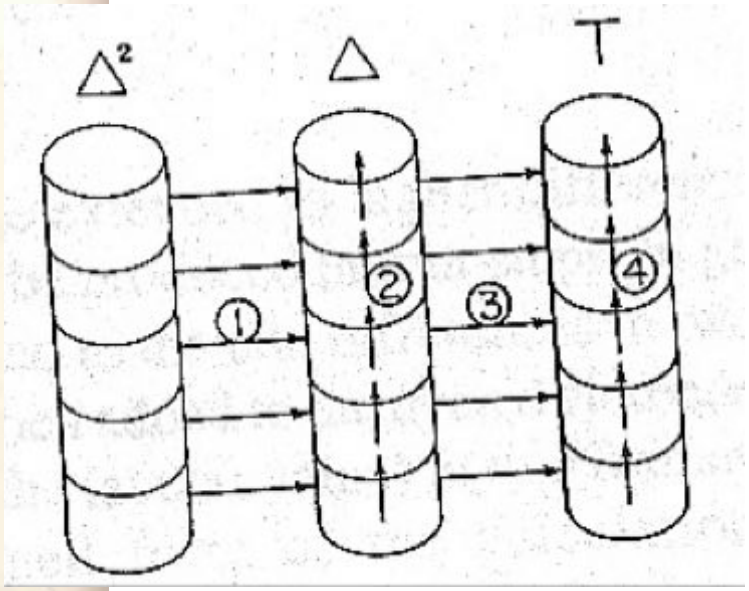
"I wish to God these calculations had been executed by steam " (1820)

TABLE II.

Nom.	Log. Hyp. 0,0	Différence. I.	I I.	III.
101000	0995.03208.53168.08286	99009.41084.53096	98027.66380	194110
01	0996.02317.94252.61382	99008.43056.86716	98025.72270	194104
02	0997.01326.37909.48098	99007.45031.14446	98023.78166	194098
03	0998.00333.82340.62544	99006.47007.36280	98021.84068	194091
04	0998.99340.29347.98824	99005.48985.52212	98019.89977	194087



difference engine n°1 (1832)
Science Museum, Londres



The Analytical Engine

1833 : "The engine eating its own tail"

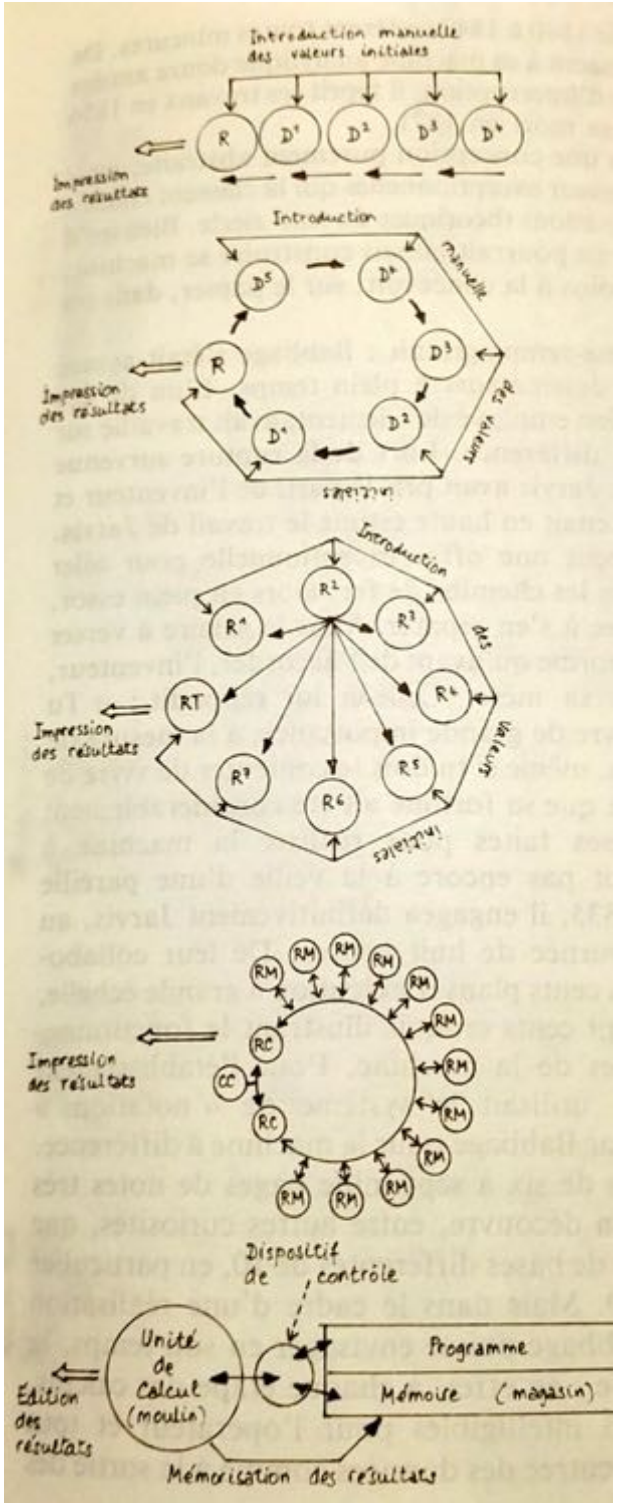
1834 - 36 :

Séparation entre le "store" (magasin des nombres) et le "mill" (moulin, pour le calcul)

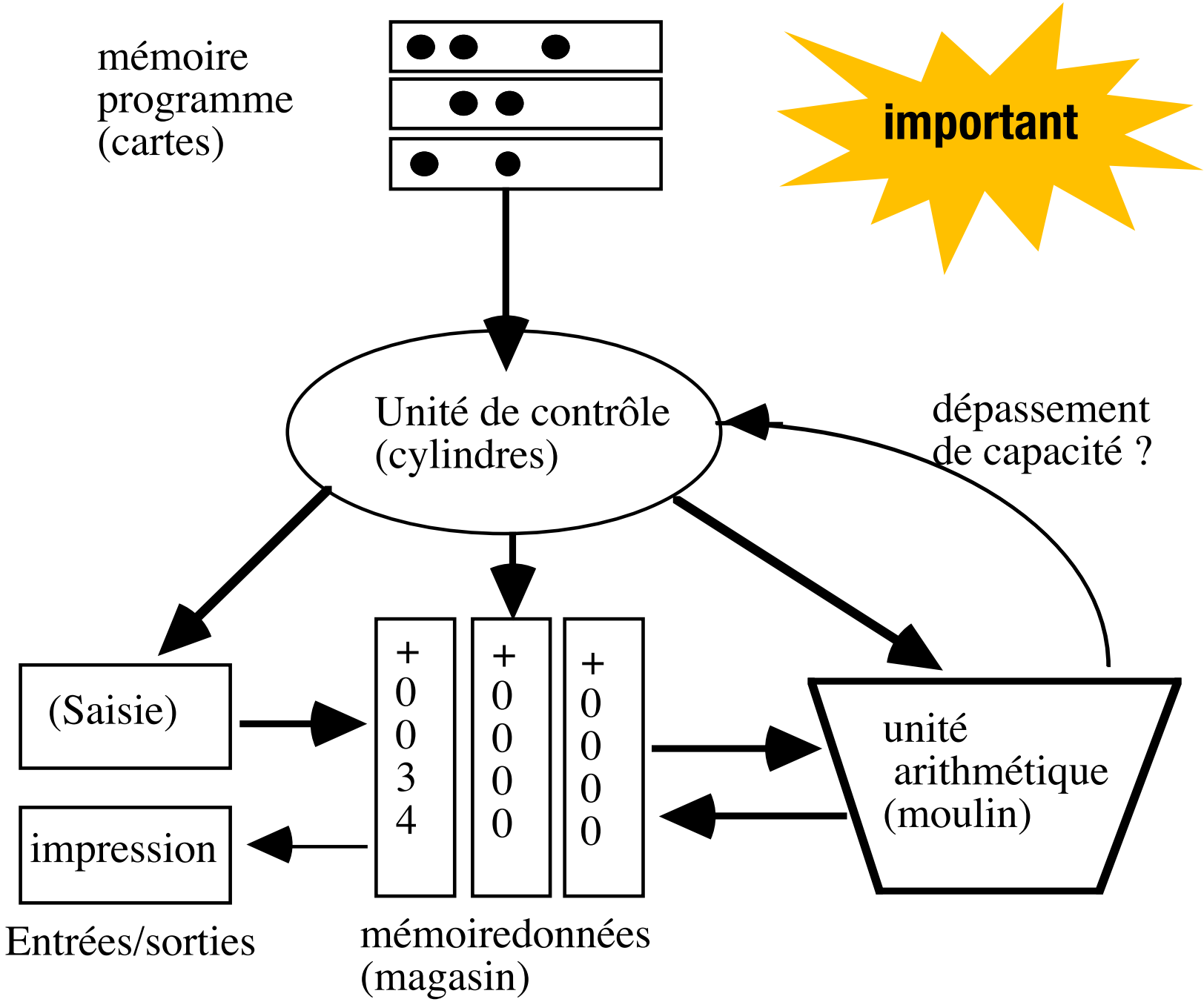
La circulation de l'information et la répétition des calculs est controlée par des "barrels" (cylindres à picot)

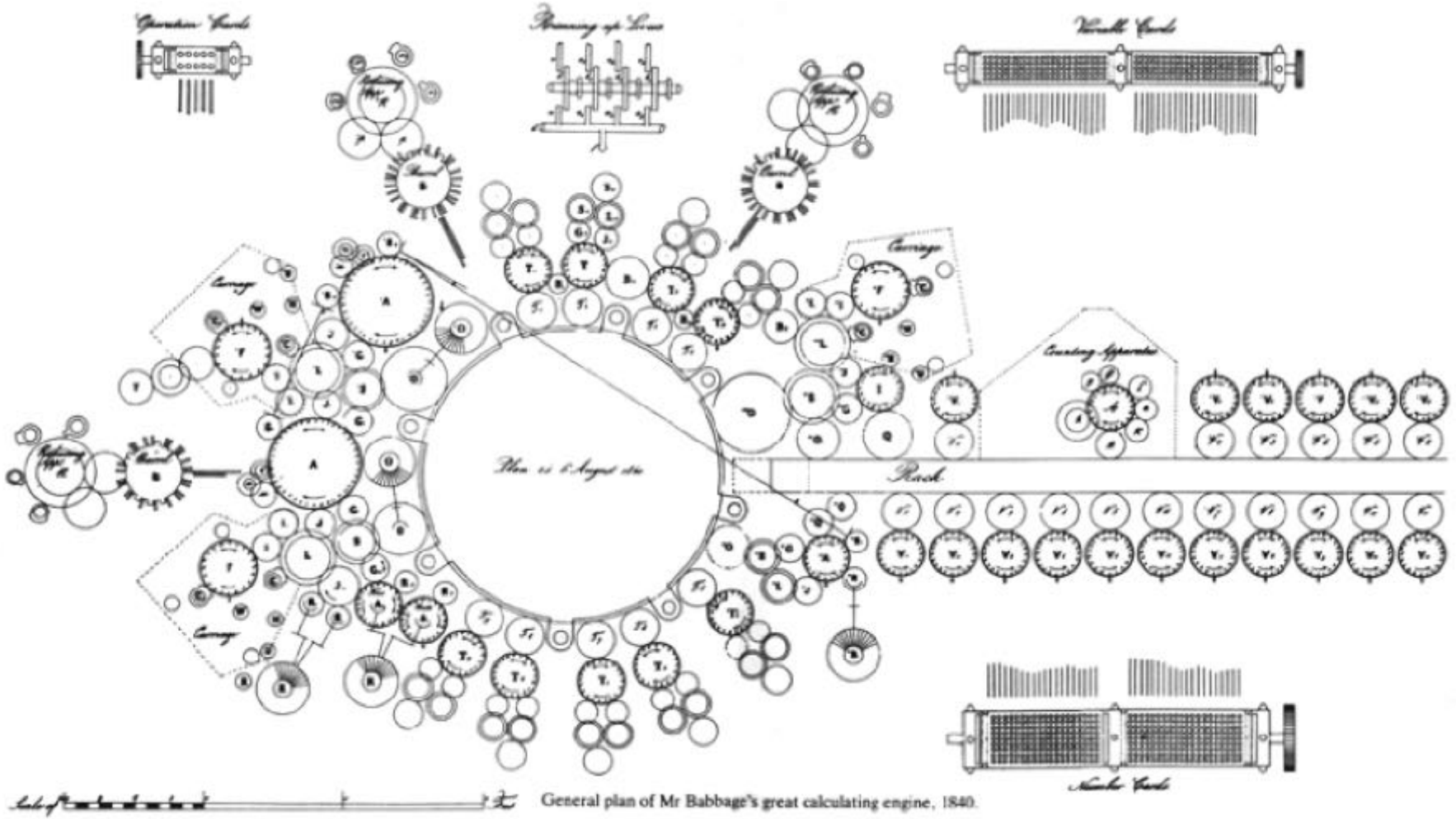
Contrôle de l'execution (programme) par des cartes perforées (Jacquard)

1838 : Design général finalisé



(R. Ligonnière. Préhistoire et histoire des ordinateurs. Robert Laffont, 1987)

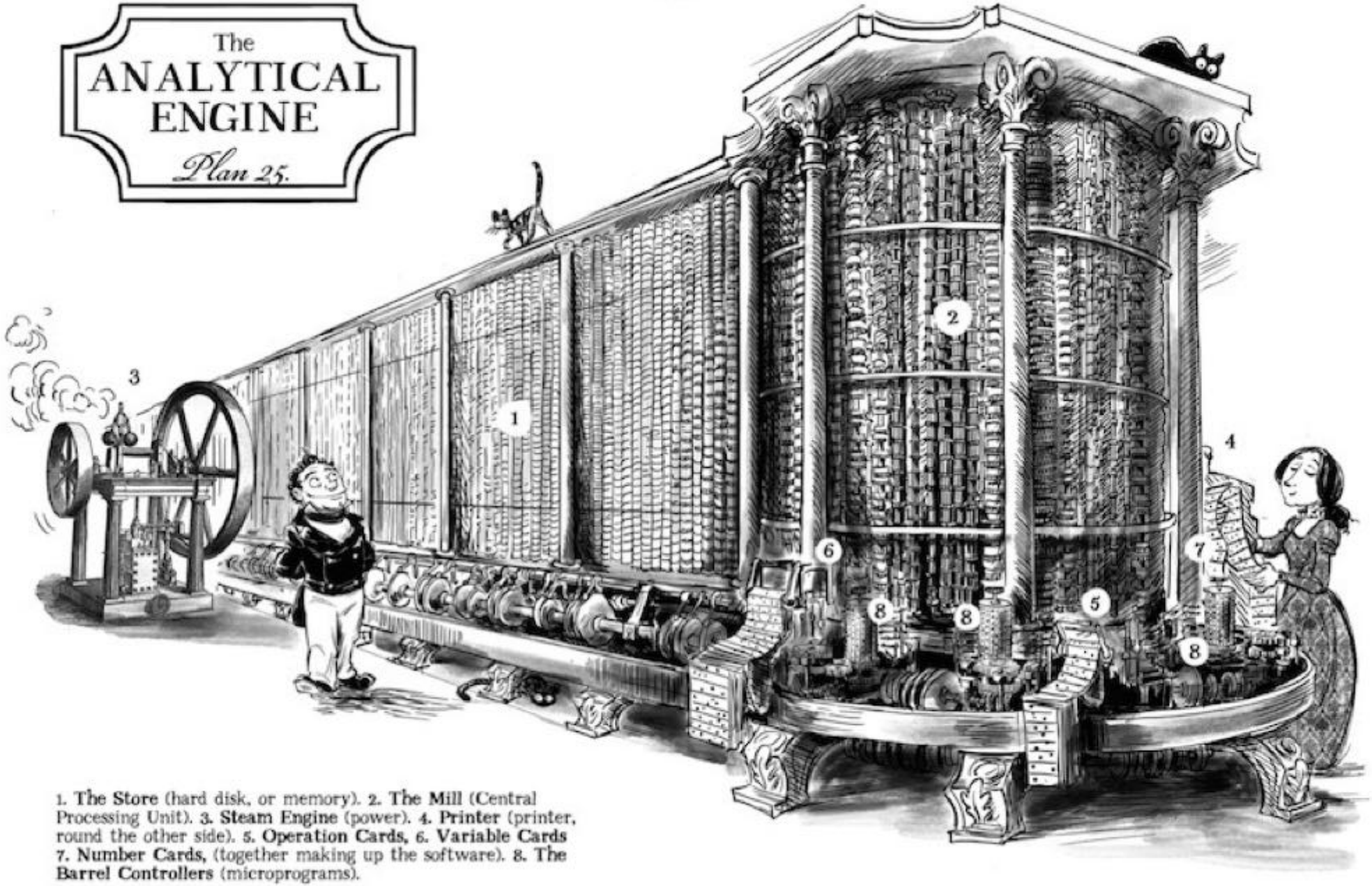

mémoire
programme
(cartes)
important
Unité de contrôle
(cylindres)
dépassement
de capacité ?
(Saisie)
impression
+
0
0
3
4
+
0
0
0
0
+
0
0
0
0
unité
arithmétique
(moulin)
Entrées/sorties
mémoiredonnées
(magasin)



General plan of Mr Babbage's great calculating engine, 1840.

A.G. BROMLEY, IEEE Annals of the history of comp. 20(4), 1998, pp. 29-45.

Hauteur 4.60 m Longeur 7.65 m Diamètre moulin 1.85m
200 colonnes (40 chiffres+signe) dans le magasin
200 colonnes de rouages dans le moulin



1. **The Store** (hard disk, or memory). 2. **The Mill** (Central Processing Unit). 3. **Steam Engine** (power). 4. **Printer** (printer, round the other side). 5. **Operation Cards**, 6. **Variable Cards** 7. **Number Cards**, (together making up the software). 8. **The Barrel Controllers** (microprograms).

https://sydneypadua.com/2dgoggles/the-marvellous-analytical-engine-how-it-works/

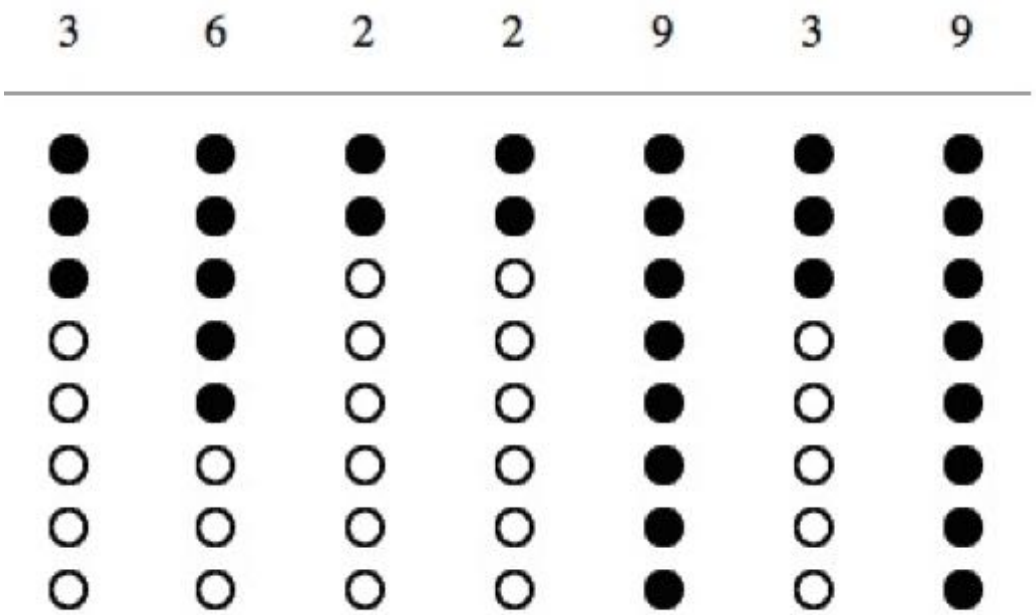
Une autre piste ? Panizzi et le British Museum

Salle de lecture circulaire

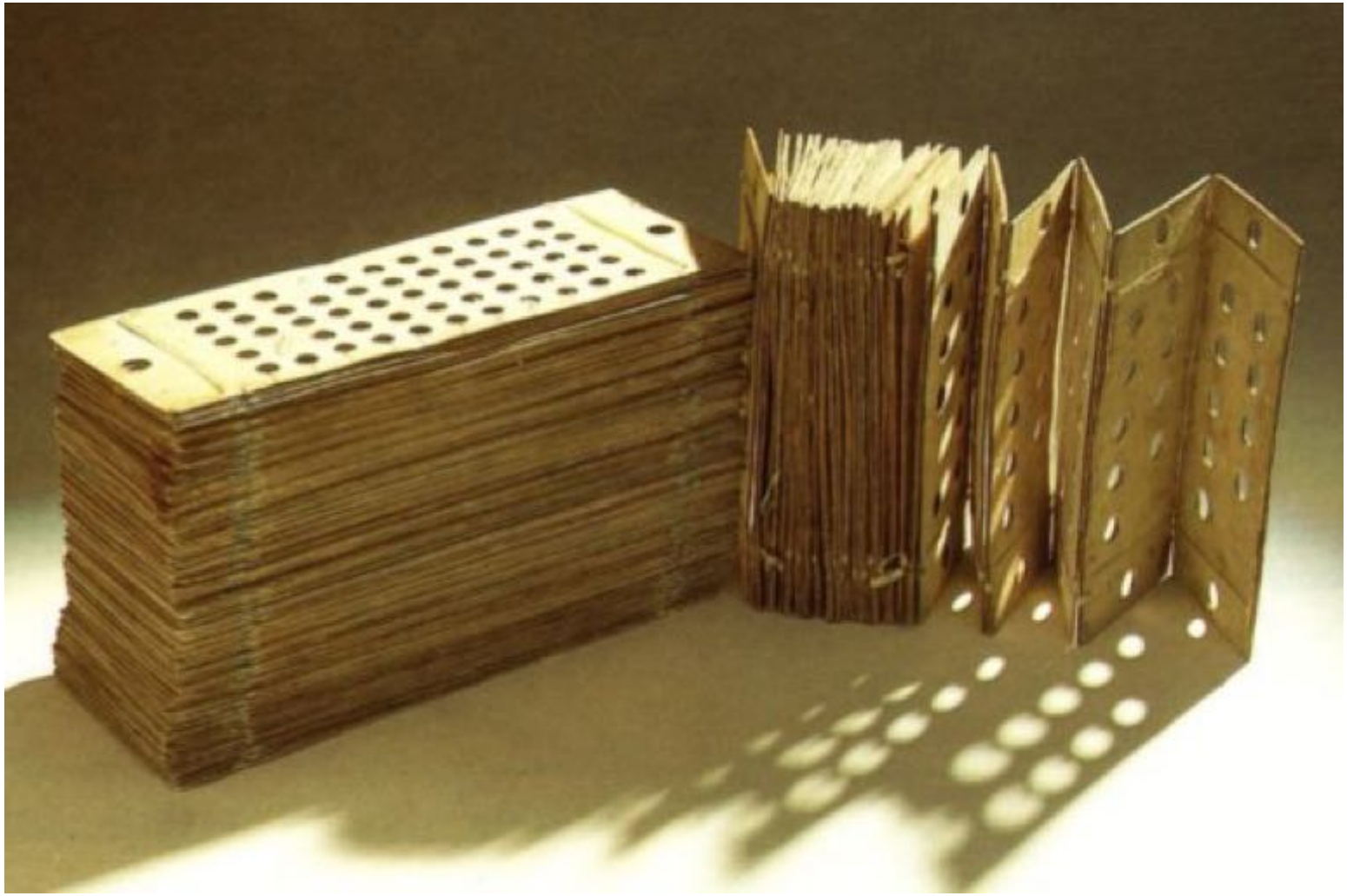
Magasins de stockage rectilignes

Métier à tisser de Jacquard

Cartes pour représenter des nombres ?

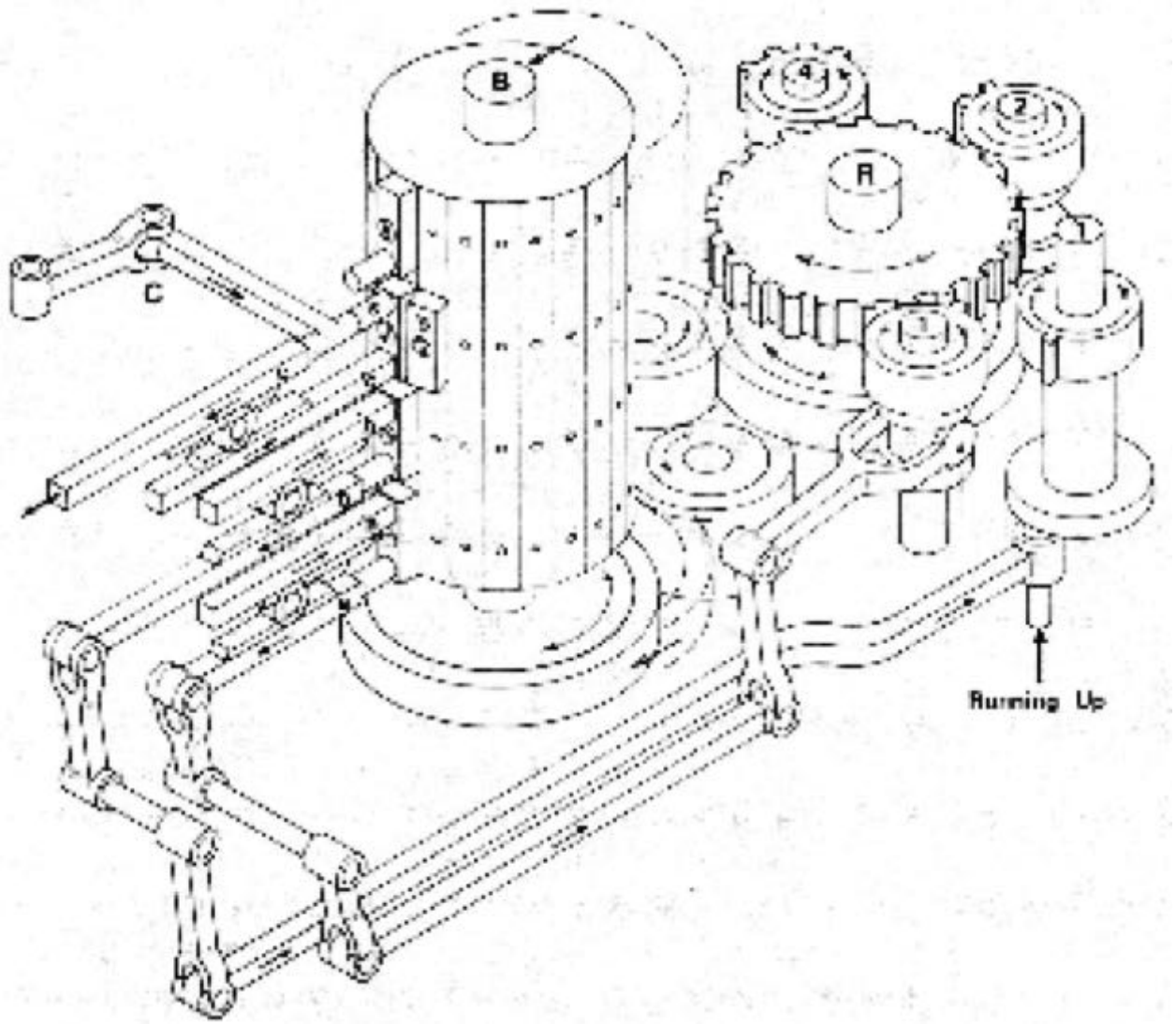


Cartes pour représenter des opérations ?



Science Museum - Londres

Unité de contrôle : les cylindres à picot



A.G. BROMLEY, IEEE Annals of the history of comp. 20(4), 1998, pp. 29-45.

le premier programme

NOMBRE des OPÉRATIONS.	CARTONS des OPÉRATIONS. — Signes indiquant la nature des opérations.	CARTONS DES VARIABLES. Colonne soumise aux opérations.	Colonne recevant le résultat des opérations.	MARCHE des OPÉRATIONS.
1	×	$V_2 \times V_4 =$	$V_8 \ldots\ldots$	$= dn'$
2	×	$V_5 \times V_1 =$	$V_9 \ldots\ldots$	$= d'n$
3	×	$V_4 \times V_0 =$	$V_{10} \ldots\ldots$	$= n'm$
4	×	$V_1 \times V_3 =$	$V_{11} \ldots\ldots$	$= nm'$
5	—	$V_8 - V_9 =$	$V_{12} \ldots\ldots$	$= dn' - d'n$
6	—	$V_{10} - V_{11} =$	$V_{13} \ldots\ldots$	$= n'm - n'm'$
7	:	$\frac{V_{12}}{V_{13}} =$	$V_{14} \ldots\ldots$	$= x = \frac{dn' - d'n}{nn' - m'n}$

NOTIONS SUR LA MACHINE ANALYTIQUE DE M. CHARLES BABBAGE, par Mr. L.-F. MENABREA, capitaine du génie militaire.

Les travaux qui appartiennent à plusieurs branches des sciences mathématiques, quoique paraissant, au premier abord, être uniquement du ressort de l'esprit, peuvent néanmoins se diviser en deux parties distinctes : l'une qu'on peut appeler mé-

ne résolvaient point le double problème que présente la question, celui d'obtenir l'exactitude des résultats unie à l'économie du temps.

Frappé de ces réflexions, Mr. Charles Babbage a consacré plusieurs années à réaliser une pensée gigantesque. Il ne s'est proposé rien moins que de construire une machine capable d'exécuter, non-seulement les calculs arithmétiques ; mais encore les calculs analytiques, dont les lois seraient connues. L'imagination est d'abord effrayée d'une telle entreprise, mais à mesure que l'on réfléchit avec plus de calme, le succès en paraît moins impossible, et l'on sent qu'il peut dépendre de la

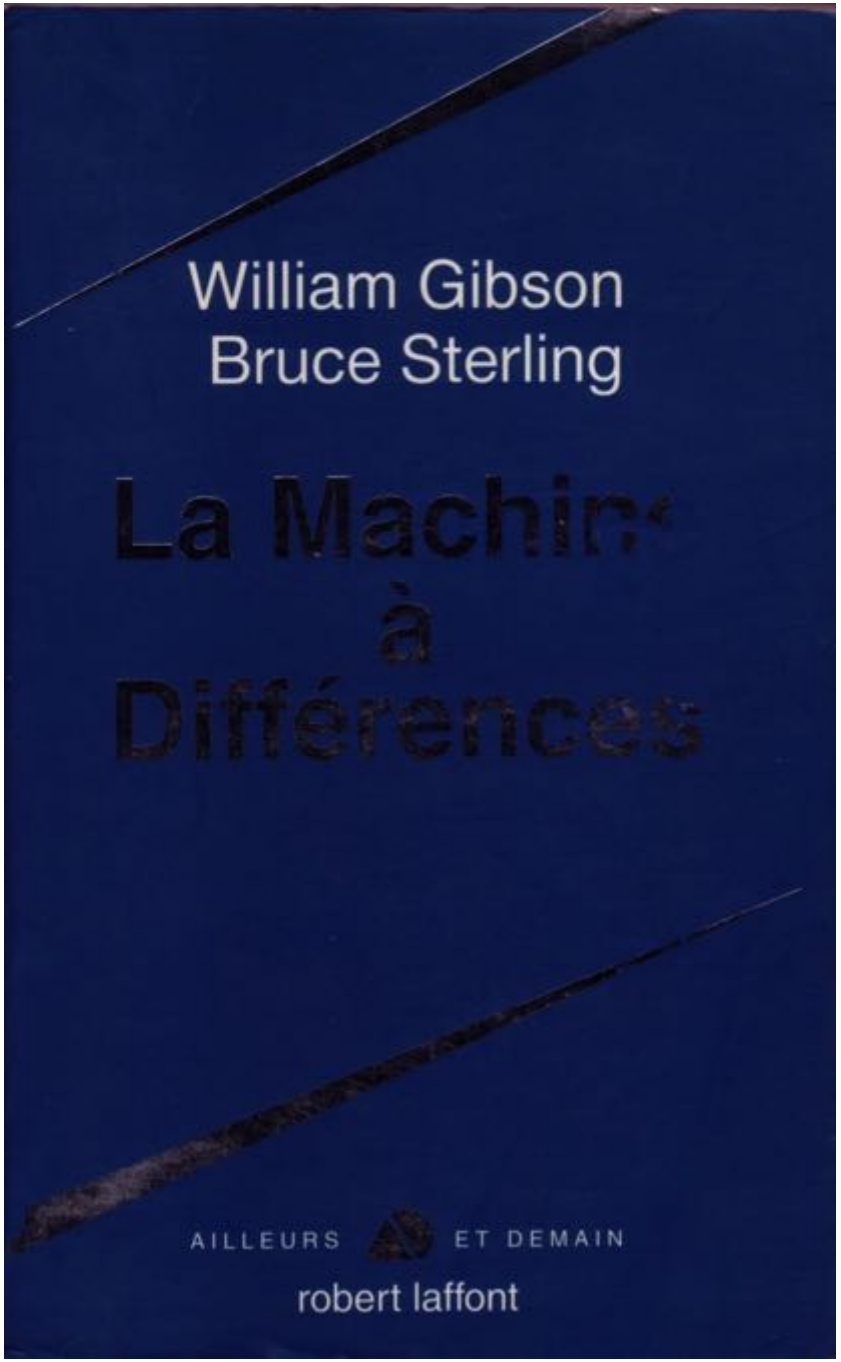
machine les ordres nécessaires pour agir. Une fois que la machine sera construite, la difficulté se reportera donc sur la confection des cartons ; mais comme ceux-ci ne sont que la traduction de formules algébriques, par le moyen de simples notations il sera facile d'en confier l'exécution à un ouvrier. Ainsi tout le travail intellectuel se bornera à la préparation des formules, qui devront être aptes à être calculées par la machine.

l'annexe G de la version anglaise

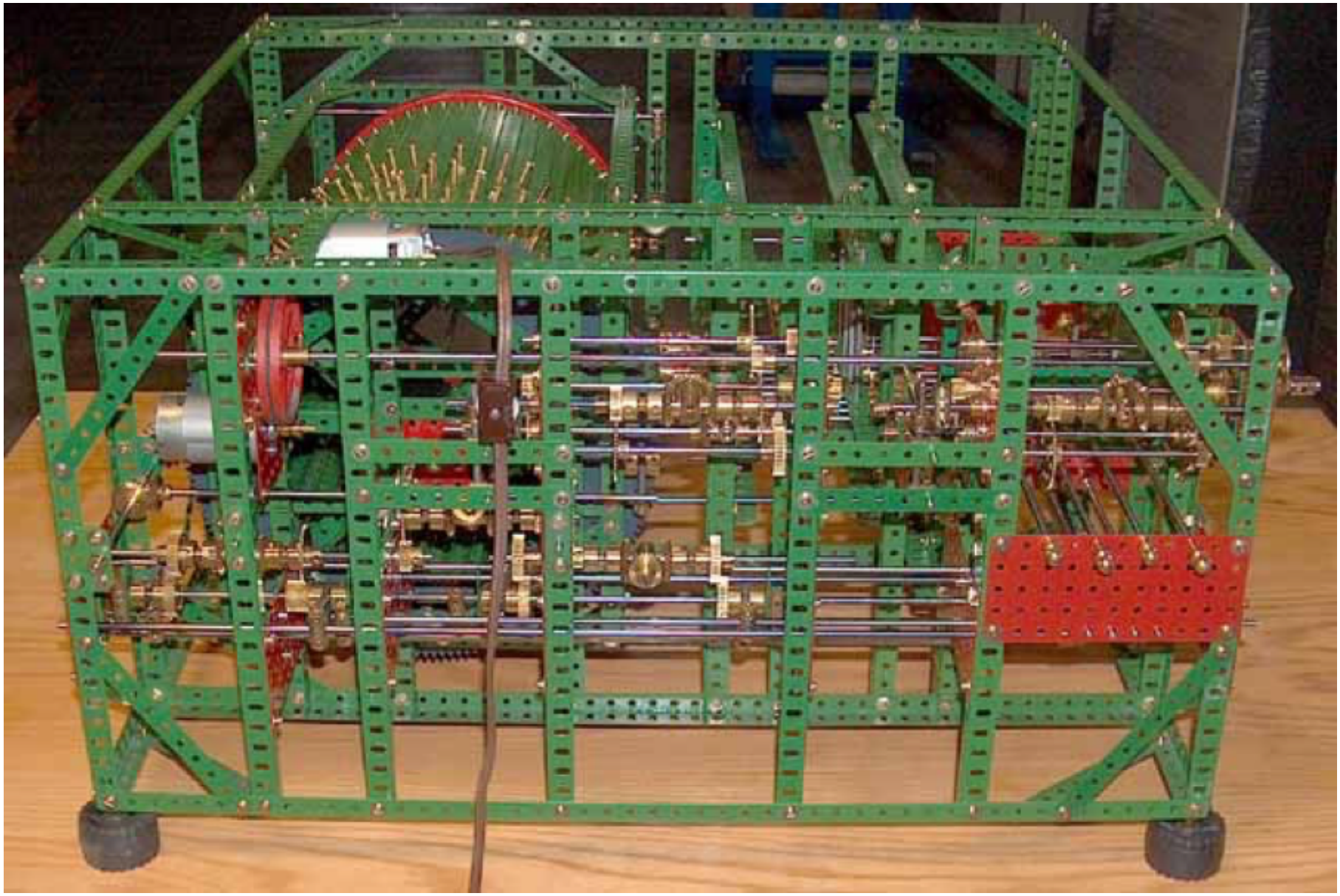
$$\left|12\right| - \left|{}^{1}V_{10} - {}^{1}V_{1}\right| {}^{2}V_{10}$$

comtesse Ada Lovelace, née Byron (1815-52)

Number of Operation	Nature of Operation	Variables acted upon	Variables receiving results	Indication of change in the value on any Variable	Statement of Results
1	×	${}^{1}V_{2} \times {}^{1}V_{3}$	${}^{1}V_{4}, {}^{1}V_{5}, {}^{1}V_{6}$	$\left\{\begin{matrix} {}^{1}V_{2} = {}^{1}V_{2} \\ {}^{1}V_{3} = {}^{1}V_{3} \end{matrix}\right\}$	$= 2n$
2	−	${}^{1}V_{4} - {}^{1}V_{1}$	${}^{2}V_{4}$	$\left\{\begin{matrix} {}^{1}V_{4} = {}^{2}V_{4} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= 2n-1$
3	+	${}^{1}V_{5} + {}^{1}V_{1}$	${}^{2}V_{5}$	$\left\{\begin{matrix} {}^{1}V_{5} = {}^{2}V_{5} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= 2n+1$
4	÷	${}^{2}V_{5} \div {}^{2}V_{4}$	${}^{1}V_{11}$	$\left\{\begin{matrix} {}^{2}V_{5} = {}^{0}V_{5} \\ {}^{2}V_{4} = {}^{0}V_{4} \end{matrix}\right\}$	$= \frac{2n-1}{2n+1}$
5	÷	${}^{1}V_{11} \div {}^{1}V_{2}$	${}^{2}V_{11}$	$\left\{\begin{matrix} {}^{1}V_{11} = {}^{2}V_{11} \\ {}^{1}V_{2} = {}^{1}V_{2} \end{matrix}\right\}$	$= \frac{1}{2} \cdot \frac{2n-1}{2n+1}$
6	−	${}^{0}V_{13} - {}^{2}V_{11}$	${}^{1}V_{13}$	$\left\{\begin{matrix} {}^{2}V_{11} = {}^{0}V_{11} \\ {}^{0}V_{13} = {}^{1}V_{13} \end{matrix}\right\}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$
7	−	${}^{1}V_{3} - {}^{1}V_{1}$	${}^{1}V_{10}$	$\left\{\begin{matrix} {}^{1}V_{3} = {}^{1}V_{3} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= n-1 (=3)$
8	+	${}^{1}V_{3} + {}^{0}V_{7}$	${}^{1}V_{7}$	$\left\{\begin{matrix} {}^{1}V_{3} = {}^{1}V_{3} \\ {}^{0}V_{7} = {}^{1}V_{7} \end{matrix}\right\}$	$= 2+0 = 2$
9	÷	${}^{1}V_{4} \div {}^{1}V_{7}$	${}^{2}V_{11}$	$\left\{\begin{matrix} {}^{1}V_{4} = {}^{1}V_{4} \\ {}^{0}V_{11} = {}^{2}V_{11} \end{matrix}\right\}$	$= \frac{2n}{2} = A_1$
10	×	${}^{1}V_{21} \times {}^{2}V_{11}$	${}^{1}V_{12}$	$\left\{\begin{matrix} {}^{1}V_{21} = {}^{1}V_{21} \\ {}^{2}V_{11} = {}^{2}V_{11} \end{matrix}\right\}$	$= B_1 \cdot \frac{2n}{2} = B_1 A_1$
11	+	${}^{1}V_{12} + {}^{1}V_{13}$	${}^{2}V_{13}$	$\left\{\begin{matrix} {}^{1}V_{12} = {}^{0}V_{12} \\ {}^{1}V_{13} = {}^{2}V_{13} \end{matrix}\right\}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}$
12	−	${}^{1}V_{10} - {}^{1}V_{1}$	${}^{2}V_{10}$	$\left\{\begin{matrix} {}^{1}V_{10} = {}^{2}V_{10} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= n-2 (=2)$
13	−	${}^{1}V_{6} - {}^{1}V_{1}$	${}^{2}V_{6}$	$\left\{\begin{matrix} {}^{1}V_{6} = {}^{2}V_{6} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= 2n-1$
14	+	${}^{1}V_{1} + {}^{1}V_{7}$	${}^{2}V_{7}$	$\left\{\begin{matrix} {}^{1}V_{1} = {}^{1}V_{1} \\ {}^{1}V_{7} = {}^{2}V_{7} \end{matrix}\right\}$	$= 2+1 = 3$
15	÷	${}^{2}V_{6} \div {}^{2}V_{7}$	${}^{1}V_{8}$	$\left\{\begin{matrix} {}^{2}V_{6} = {}^{2}V_{6} \\ {}^{2}V_{7} = {}^{2}V_{7} \end{matrix}\right\}$	$= \frac{2n-1}{3}$
16	×	${}^{2}V_{8} \times {}^{2}V_{11}$	${}^{4}V_{11}$	$\left\{\begin{matrix} {}^{1}V_{8} = {}^{0}V_{8} \\ {}^{2}V_{11} = {}^{4}V_{11} \end{matrix}\right\}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3}$
17	−	${}^{2}V_{6} - {}^{1}V_{1}$	${}^{3}V_{6}$	$\left\{\begin{matrix} {}^{2}V_{6} = {}^{3}V_{6} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= 2n-2$
18	+	${}^{1}V_{1} + {}^{2}V_{7}$	${}^{3}V_{7}$	$\left\{\begin{matrix} {}^{2}V_{7} = {}^{3}V_{7} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= 3+1 = 4$
19	+	${}^{2}V_{6} \div {}^{2}V_{7}$	${}^{1}V_{9}$	$\left\{\begin{matrix} {}^{2}V_{6} = {}^{2}V_{6} \\ {}^{2}V_{7} = {}^{2}V_{7} \end{matrix}\right\}$	$= \frac{2n-2}{4}$
20	×	${}^{1}V_{9} \times {}^{4}V_{11}$	${}^{2}V_{11}$	$\left\{\begin{matrix} {}^{1}V_{9} = {}^{0}V_{9} \\ {}^{4}V_{11} = {}^{5}V_{11} \end{matrix}\right\}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4} = A_2$
21	×	${}^{1}V_{22} \times {}^{5}V_{11}$	${}^{0}V_{12}$	$\left\{\begin{matrix} {}^{1}V_{22} = {}^{1}V_{22} \\ {}^{0}V_{12} = {}^{5}V_{12} \end{matrix}\right\}$	$= B_2 \cdot \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = B_2 A_2$
22	+	${}^{5}V_{12} + {}^{5}V_{13}$	${}^{5}V_{13}$	$\left\{\begin{matrix} {}^{5}V_{12} = {}^{0}V_{12} \\ {}^{5}V_{13} = {}^{5}V_{13} \end{matrix}\right\}$	$= A_0 + B_1 A_1 + B_2 A_2$
23	−	${}^{5}V_{10} - {}^{1}V_{1}$	${}^{5}V_{10}$	$\left\{\begin{matrix} {}^{5}V_{10} = {}^{5}V_{10} \\ {}^{1}V_{1} = {}^{1}V_{1} \end{matrix}\right\}$	$= n-3 (=1)$
24	+	${}^{4}V_{13} + {}^{0}V_{24}$	${}^{1}V_{24}$	$\left\{\begin{matrix} {}^{4}V_{13} = {}^{0}V_{13} \\ {}^{0}V_{24} = {}^{1}V_{24} \end{matrix}\right\}$	$= B_2$
25	+	${}^{1}V_{1} + {}^{1}V_{2}$	${}^{1}V_{2}$	$\left\{\begin{matrix} {}^{1}V_{1} = {}^{1}V_{1} \\ {}^{1}V_{2} = {}^{1}V_{2} \\ {}^{5}V_{6} = {}^{0}V_{6} \\ {}^{5}V_{7} = {}^{0}V_{7} \end{matrix}\right\}$	$= n+1 = 4+1 = 5$ by a Variable-card. by a Variable-card.


William Gibson
Bruce Sterling
La Machine
à
Différences
AILLEURS ET DEMAIN
robert laffont

Tim Robinson http://www.meccano.us/



Calcul binaire : c’est beaucoup plus simple !

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 10

faux ET faux = faux
faux ET vrai =
vrai ET faux =
vrai ET vrai =

faux OU faux = faux
faux OU vrai =
vrai OU faux =
vrai OU vrai =

faux OU-EX faux = faux
faux OU-EX vrai =
vrai OU-EX faux =
vrai OU-EX vrai =

lois de Morgan : non(A ET B) = non(A) OU non(B)
non(A OU B) = non(A) ET non(B)

Avec des billes

http://woodgears.ca/marbleadd/

video

Evil Mad Scientist

Search...

PRODUCTS BLOG SUPPORT ABOUT CONTACT







Digi-Comp II: First Edition

Rolling-Ball Binary Digital Mechanical Computer

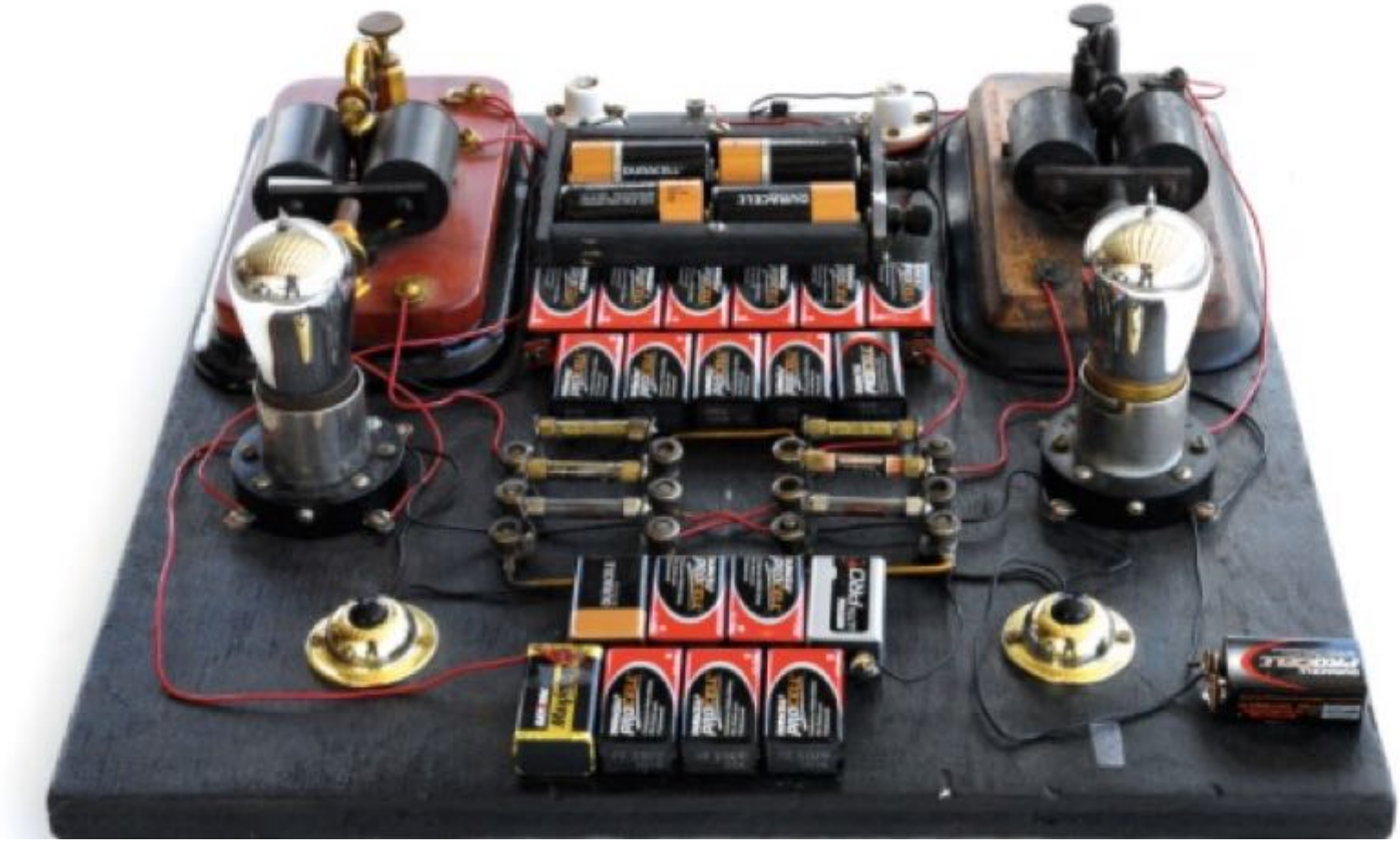
In stock

$349, with free shipping to US addresses

Add to Cart

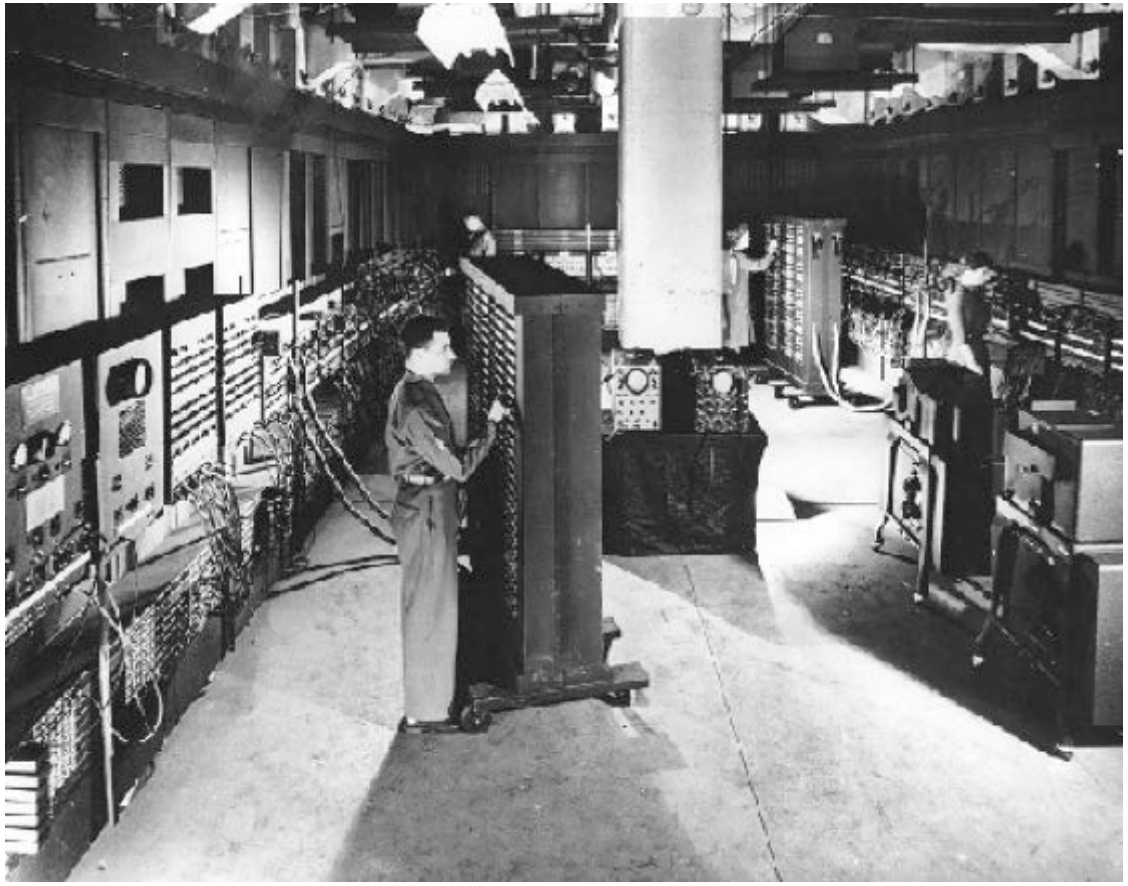
SKU: 970

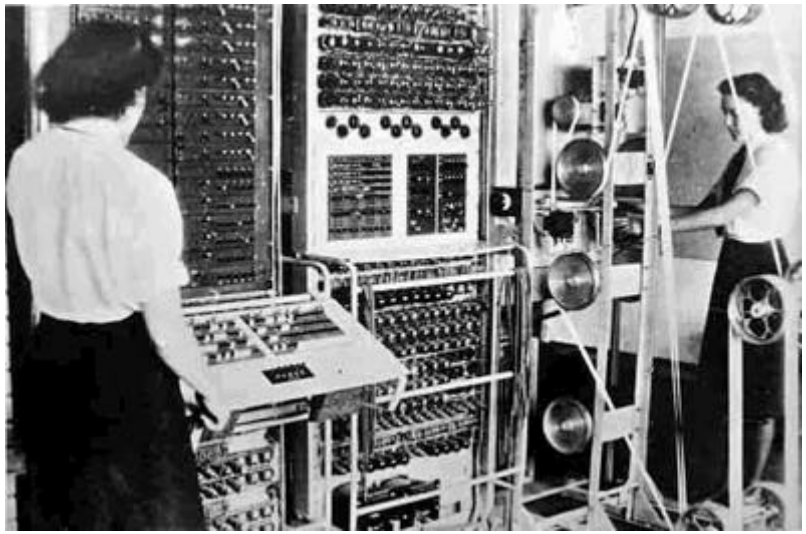
Mieux : avec l’électricité



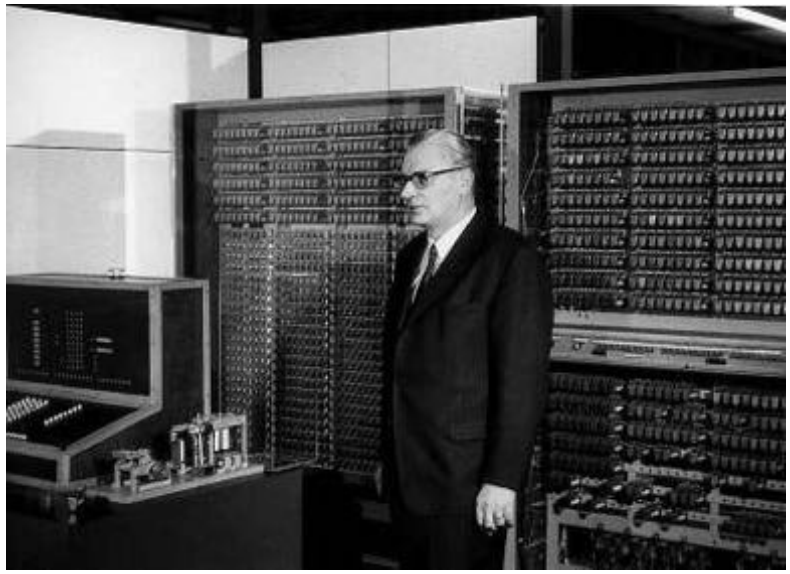
https://spectrum.ieee.org/geek-life/hands-on/recreating-the-first-flipflop

(2018 : les 100 ans du flip-flop)





Colossus (A. Turing etc.)



Z3 de K. Zuse

ENIAC : electronic numerical integrator and computer (Eckert & Mauchly, etc.)

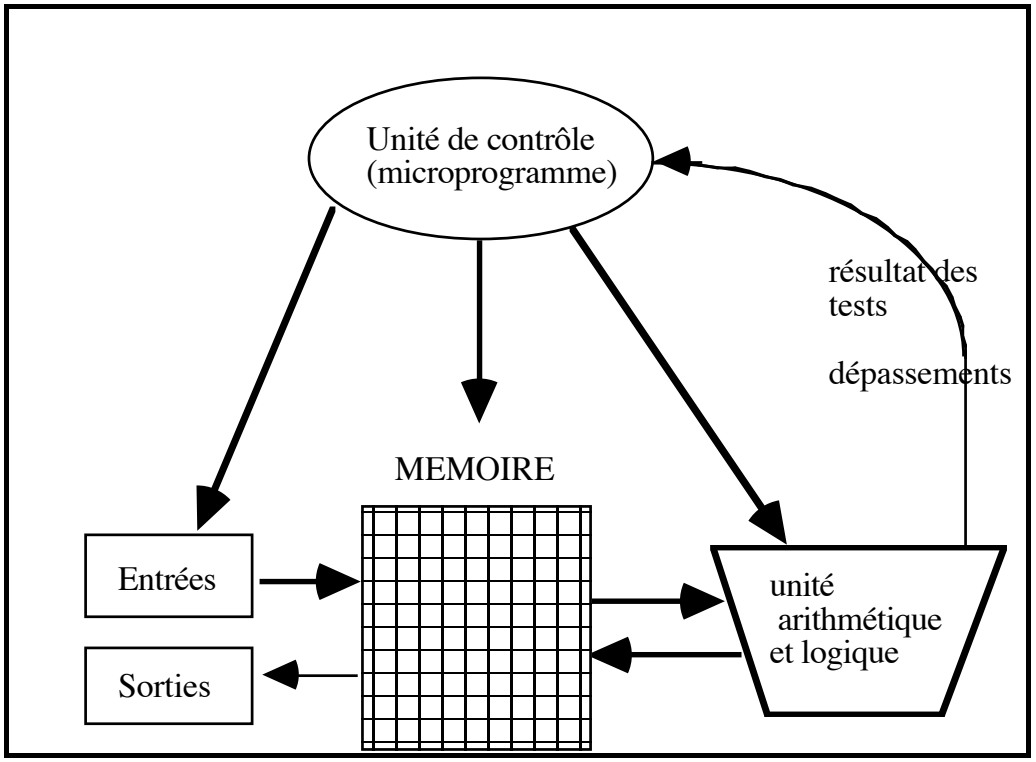
Cadencé par une horloge à 0.1 MHz
Addition : 0.2 ms
Multiplication : 3 ms
Division : 30 ms
20 cases mémoires de 10 chiffres
18000 tubes
70000 résistances, 10000 condensateurs, 6000 interupteurs
Consommation de 140 KW
35 m de long, 3 m de haut, 12 cm de profondeur, 30 tonnes

EDVAC

" if the machine can in some fashion distinguish a number form an order, the memory organ can be used to store both numbers and orders. "

Preliminary discussion of the logical design of an electronic computing instrument
A.W. Burks, H.H. Goldstine, J. von Neumann

J. von Neumann (1903-57)



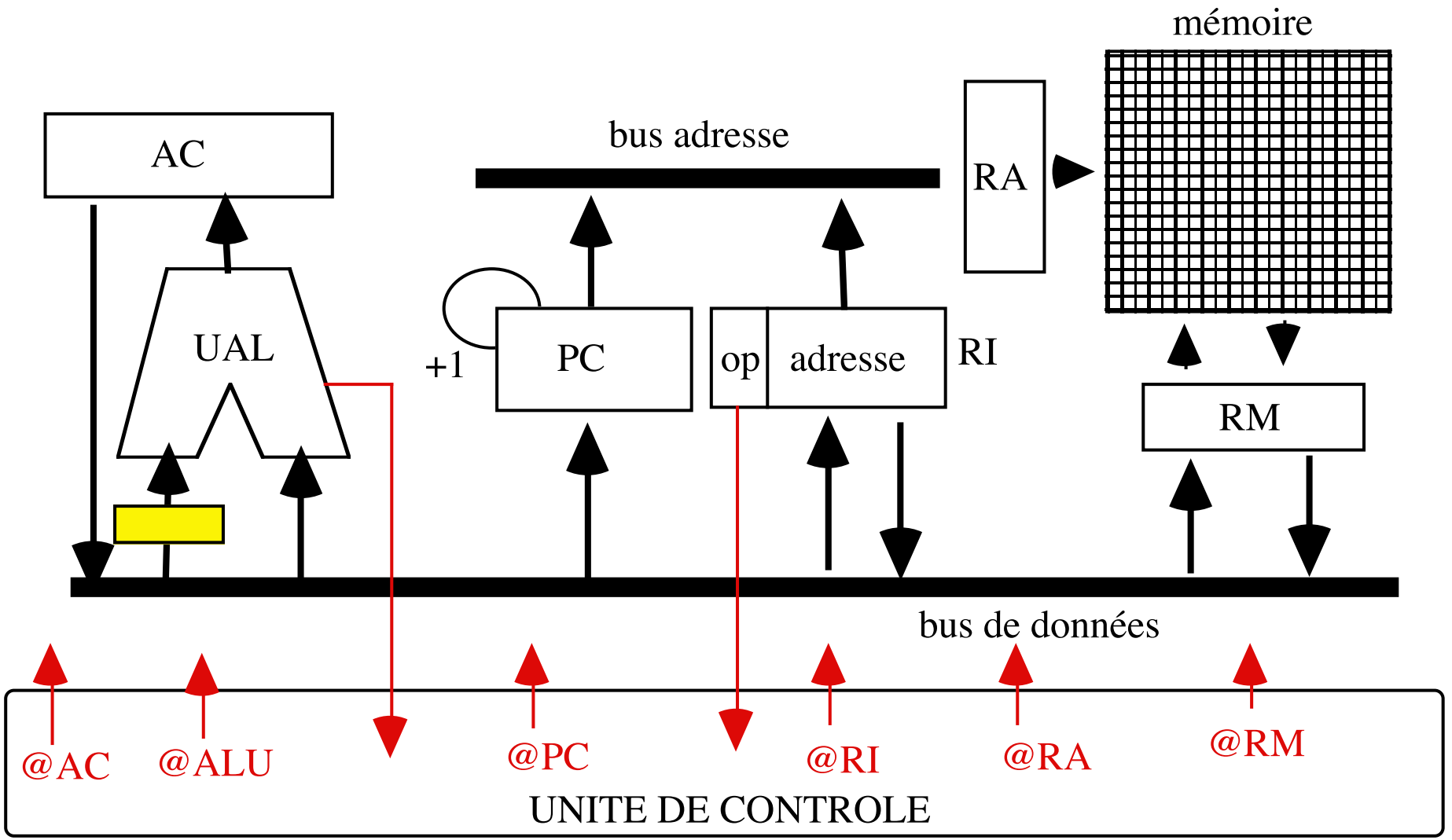
If people do not believe that mathematics is simple, it is only because they do not realize how complicated life is

Le programme enregistré !

Pouvoir assimiler des instructions à des données permet d'écrire :

- des programmes qui produisent d'autres programmes (compilateurs)

- des programmes qui lancent/arrêtent/gèrent d'autres programmes (systèmes d'exploitation, debogueurs, ...)



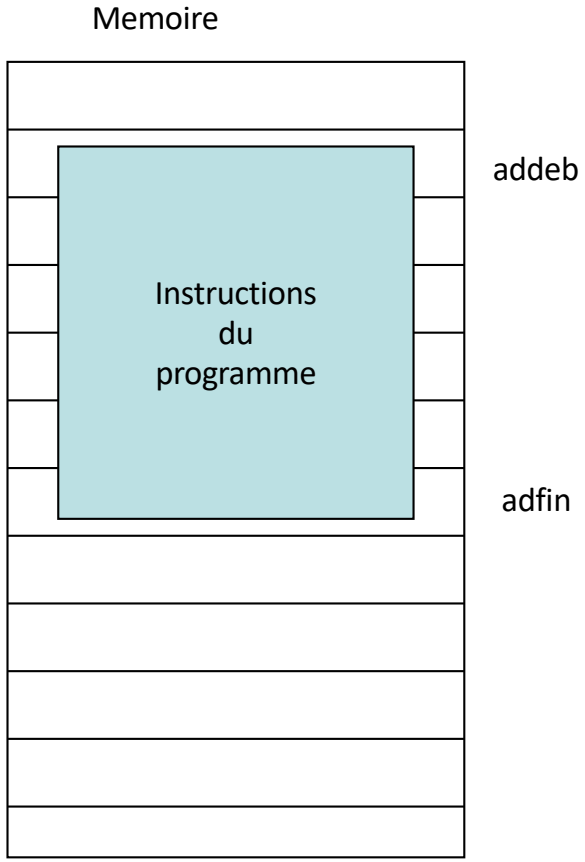
important

AC : registre accumulateur
UAL / ALU : unité arithmétique et logique
PC : Compteur de programme
RI : registre d'instruction
RA : registre d'adresse
RM : registre de mot

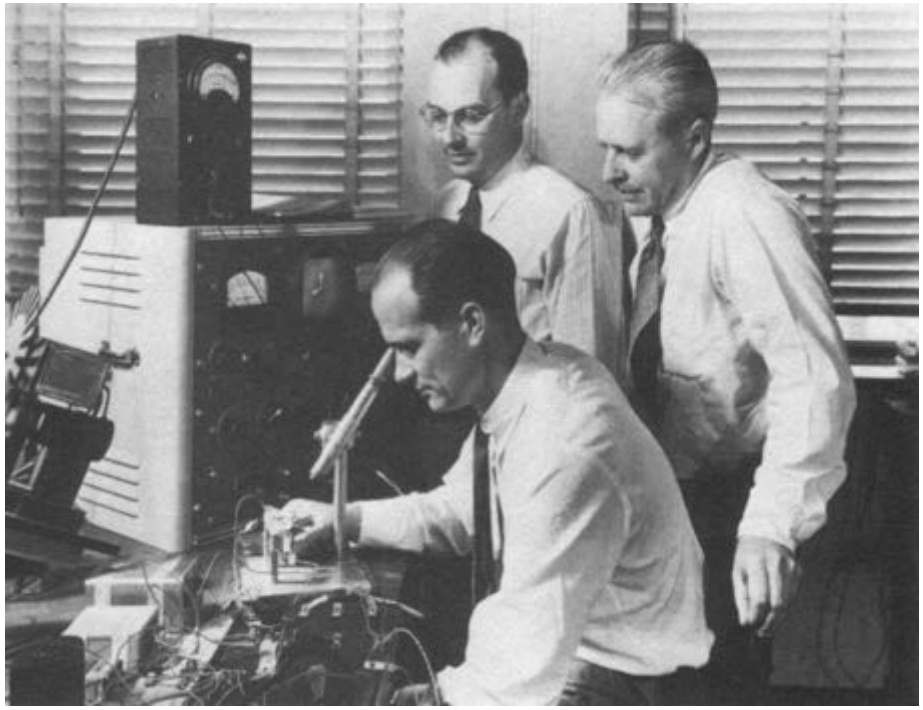
Un programme type : le "virus"

On veut écrire un programme dont la seule fonction est de se dupliquer dans la mémoire, par saut de 1000 cases. On supposera que la position en mémoire de la première instruction est connue. Le programme s'arrête quand toute la mémoire est parcourue.

```
for k := addeb to adfin loop
    Mem(k+1000) := Mem(k)
end loop
addeb := addeb + 1000
adfin := adfin + 1000
if adfin < Mem'size then goto addeb
```

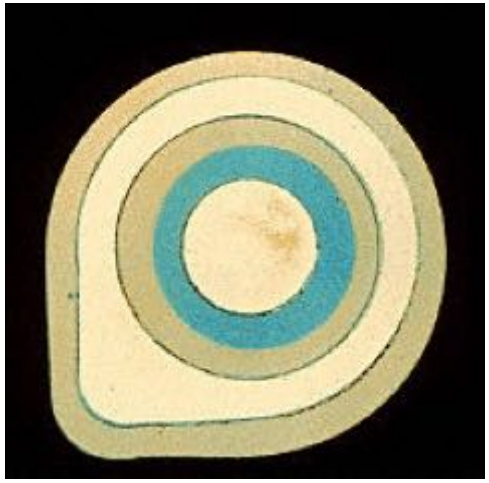


Le transistor et les circuits intégrés

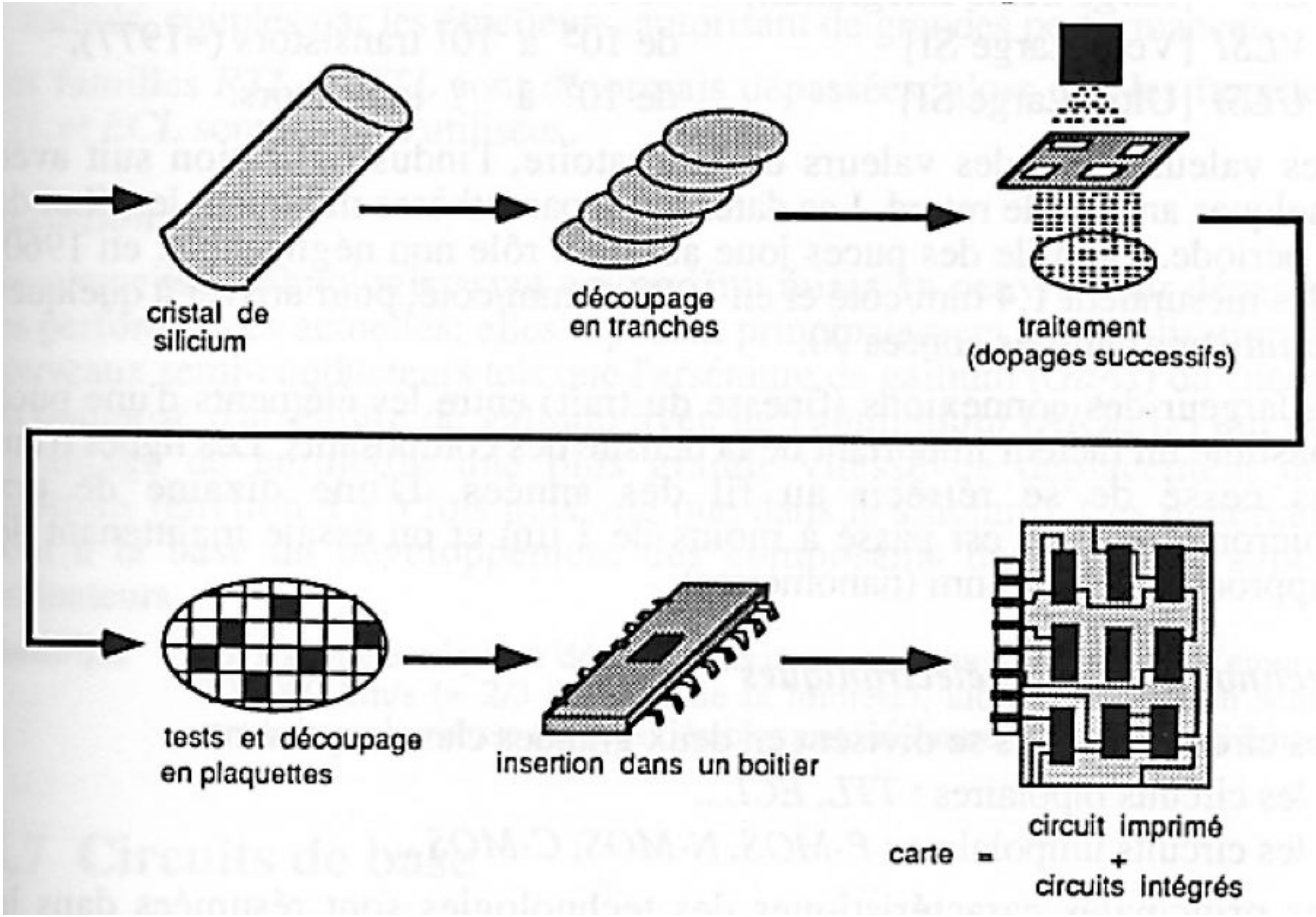


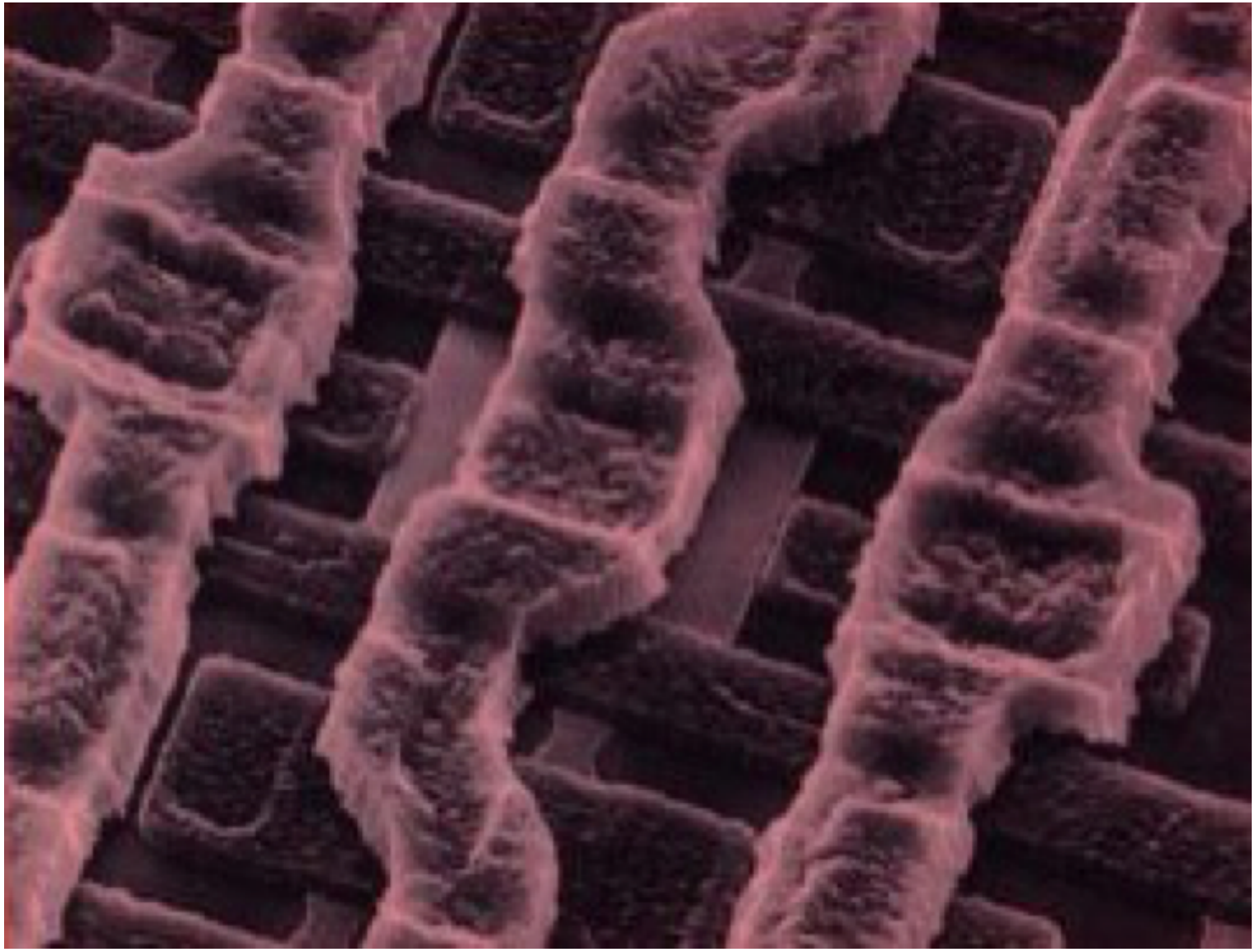
Shockley (assis), Bardeen (gauche) et Brattain (droite
- Bell Labs, Murray Hill, , New Jersey (1947)

Shocley (assis en bout de table) au moment
du prix Nobel (1956)
Robert Noyce debout au centre. Moore ?



Progrès de l'intégration

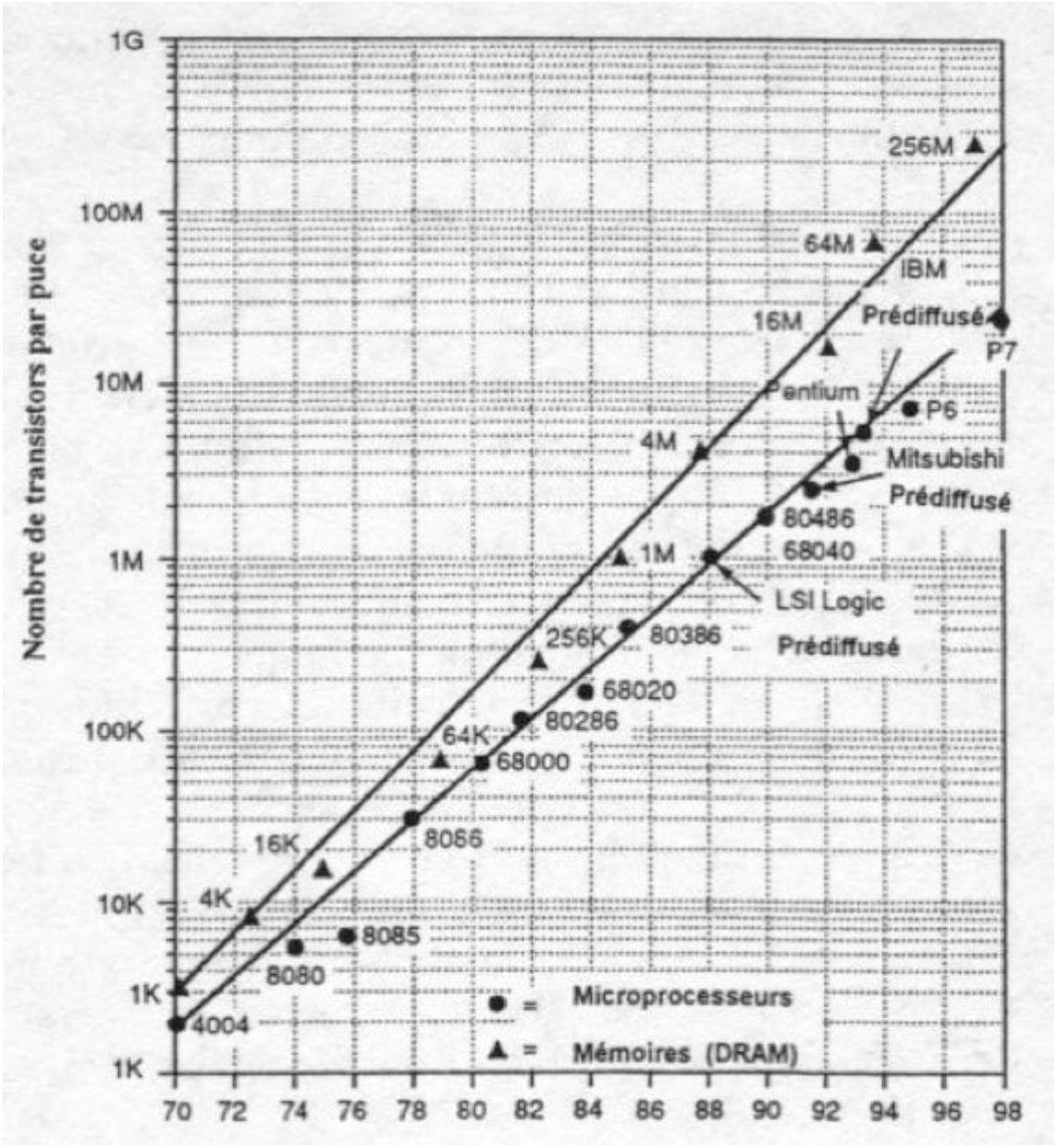


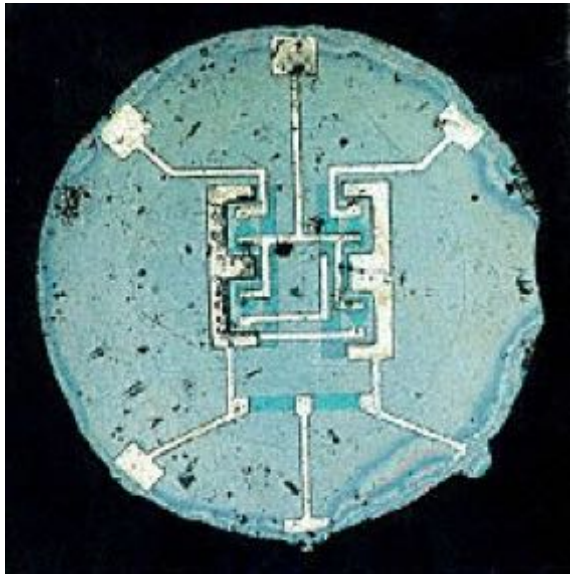


La « loi » de Gordon Moore

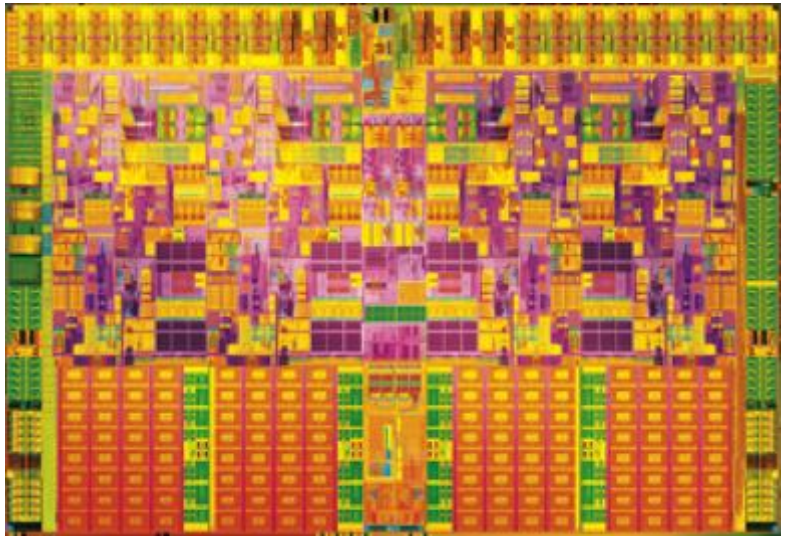
Densité mémoire : x 1.5 / an

Densité microprocesseur : x 1.35 / an





1962



2008

Préfixes du Système international d'unités — Wikipédia

1000^m	10^n	Préfixe français	Symbole	Depuis note 1	Nombre décimal	Désignation note 2
1000^8	10^{24}	yotta	Y	1991	1 000 000 000 000 000 000 000 000	Quadrillion
1000^7	10^{21}	zetta	Z	1991	1 000 000 000 000 000 000 000	Trilliard
1000^6	10^{18}	exa	E	1975	1 000 000 000 000 000 000	Trillion
1000^5	10^{15}	péta	P	1975	1 000 000 000 000 000	Billiard
1000^4	10^{12}	téra	T	1960	1 000 000 000 000	Billion
1000^3	10^9	giga	G	1960	1 000 000 000	Milliard
1000^2	10^6	méga	M	1960	1 000 000	Million
1000^1	10^3	kilo	k	1795	1 000	Millier
$1000^{2/3}$	10^2	hecto	h	1795	100	Centaine
$1000^{1/3}$	10^1	déca	da	1795	10	Dizaine
1000^0	10^0	(aucun)	—	—	1	Unité
$1000^{-1/3}$	10^{-1}	déci	d	1795	0,1	Dixième
$1000^{-2/3}$	10^{-2}	centi	c	1795	0,01	Centième
1000^{-1}	10^{-3}	milli	m	1795	0,001	Millième
1000^{-2}	10^{-6}	micro	μ	1960 note 3	0,000 001	Millionième
1000^{-3}	10^{-9}	nano	n	1960	0,000 000 001	Milliardième
1000^{-4}	10^{-12}	pico	p	1960	0,000 000 000 001	Billionième
1000^{-5}	10^{-15}	femto	f	1964	0,000 000 000 000 001	Billiardième
1000^{-6}	10^{-18}	atto	a	1964	0,000 000 000 000 000 001	Trillionième
1000^{-7}	10^{-21}	zepto	z	1991	0,000 000 000 000 000 000 001	Trilliardième
1000^{-8}	10^{-24}	yocto	y	1991	0,000 000 000 000 000 000 000 001	Quadrillionième

Quelques ordres de grandeurs

Capacité mémoire : en bits

1 octet = 8 bits = 1 byte (pas toujours)
1 o = 1 caractère (table d'encodage ISOLatin1)
3 o = un pixel sur un écran "million de couleur"
4 o = 1 nombre réel dans la norme IEEE-754

2 Ko (kilo) = 10^3 o = une page de texte

1Mo (mega) = 10^6 o = Un roman en mode texte
20 Mo = Un roman en fac-similé = Un logiciel
500 Mo = Texte de l'Encyclopedia Universalis (CDROM)

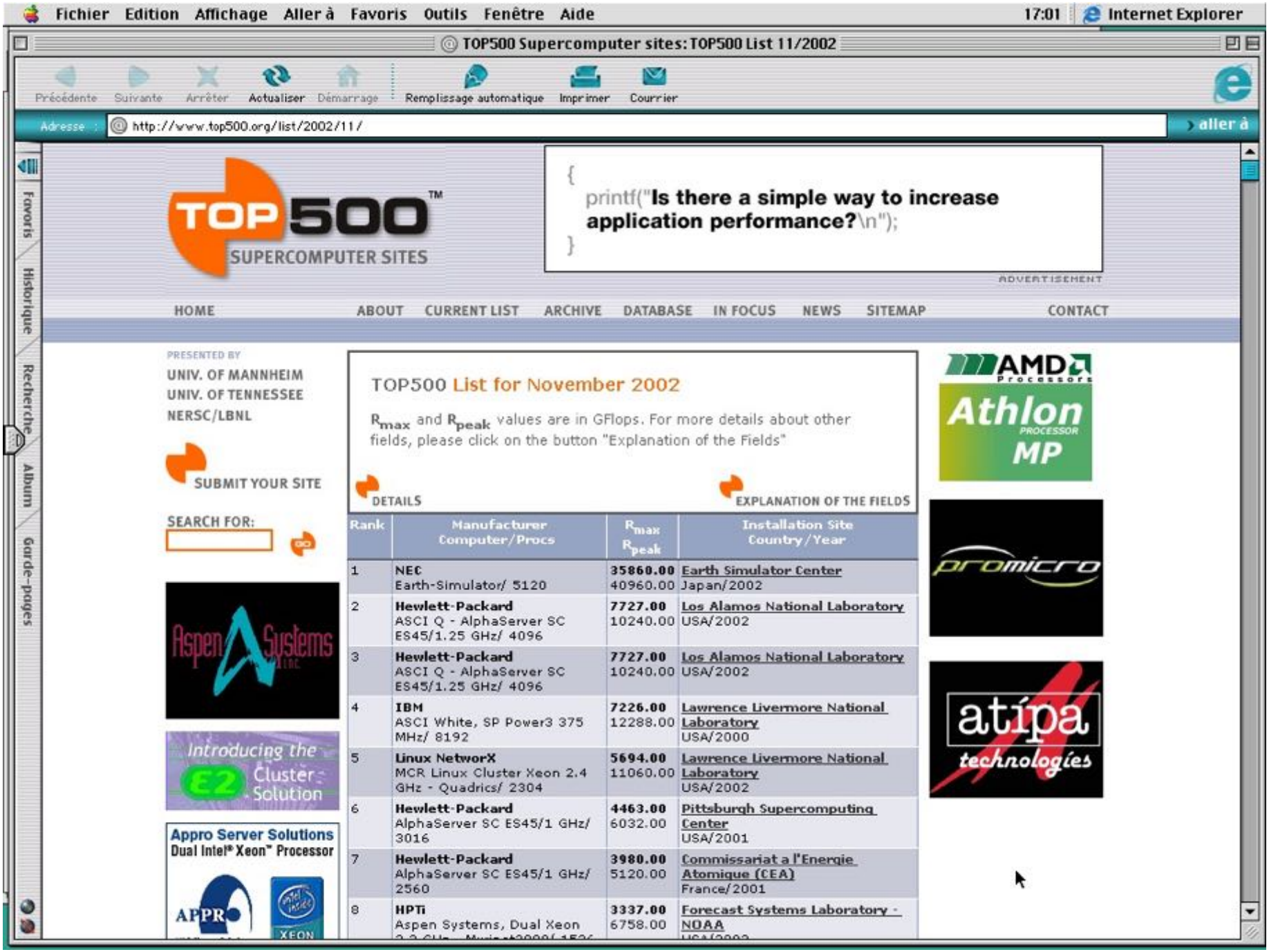
1 Go (giga) = 10^9 o
= Une bibliothèque personnelle en mode texte
10 Go = Un film compressé (DVD)
600 Go = Une librairie en fac-similé (300 Kvol.)

1 To (tera) = 10^12 o = Une cinémathèque personnelle
20 To = plus grand assemblage de disques en 1996 (LNB)
= la "library of congress" en mode texte (50 Mvol)

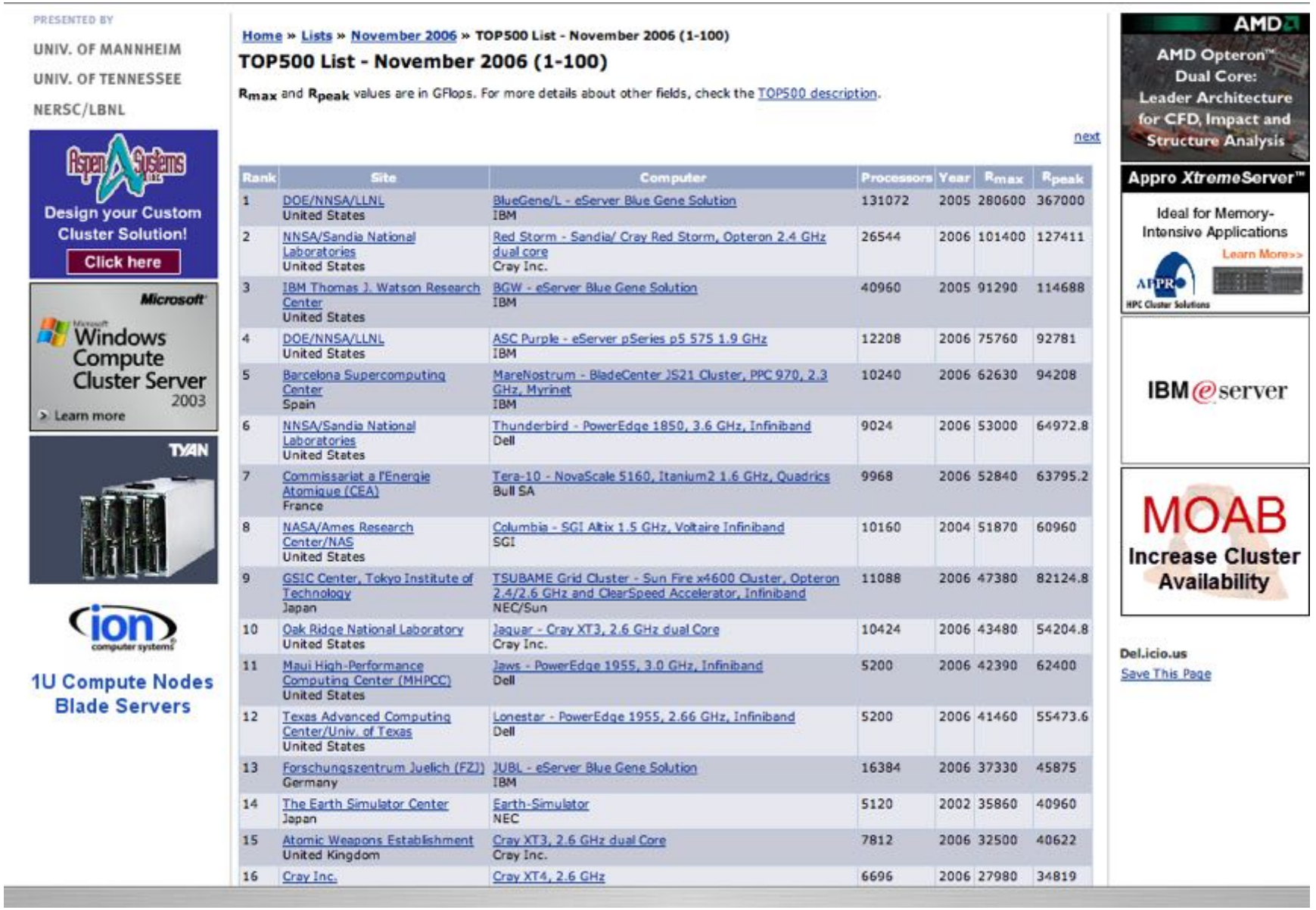
1 Po (peta) = 10^15 o
= Une bibliothèque nationale en mode image
15 Po = production mondiale de disques en 1995

Temps d'accès (≠ tps de cycle) : en secondes

1 ms (milli) = 10^-3 s
20 ms : tps accès maximum pour un relais
1 µs (micro) = 10^-6 s = tps accès maximum pour un tore
1 ns (nano) = 10^-9s
100 ns = temps d'accès DRAM
5 ns = temps d'accès bascule élementaire
1 ps (pico) = 10^-12s
temps pour qu'un signal lumineux ds le vide parcoure 3 cm


Fichier Edition Affichage Aller à Favoris Outils Fenêtre Aide
17:01
Internet Explorer
TOP500 Supercomputer sites: TOP500 List 11/2002
Précédente Suivante Arrêter Actualiser Démarrage Remplissage automatique Imprimer Courrier
Adresse : http://www.top500.org/list/2002/11/
aller à
Favoris Historique Recherche Album Garde-pages
TOP500 SUPERCOMPUTER SITES
{
printf("Is there a simple way to increase application performance?\n");
}
ADVERTISEMENT
HOME ABOUT CURRENT LIST ARCHIVE DATABASE IN FOCUS NEWS SITEMAP CONTACT
PRESENTED BY
UNIV. OF MANNHEIM
UNIV. OF TENNESSEE
NERSC/LBNL
SUBMIT YOUR SITE
SEARCH FOR:
TOP500 List for November 2002
Rmax and Rpeak values are in GFlops. For more details about other fields, please click on the button "Explanation of the Fields"
DETAILS
EXPLANATION OF THE FIELDS
Rank | Manufacturer Computer/Procs | Rmax Rpeak | Installation Site Country/Year
1 | NEC Earth-Simulator/ 5120 | 35860.00 40960.00 | Earth Simulator Center Japan/2002
2 | Hewlett-Packard ASCI Q - AlphaServer SC ES45/1.25 GHz/ 4096 | 7727.00 10240.00 | Los Alamos National Laboratory USA/2002
3 | Hewlett-Packard ASCI Q - AlphaServer SC ES45/1.25 GHz/ 4096 | 7727.00 10240.00 | Los Alamos National Laboratory USA/2002
4 | IBM ASCI White, SP Power3 375 MHz/ 8192 | 7226.00 12288.00 | Lawrence Livermore National Laboratory USA/2000
5 | Linux NetworX MCR Linux Cluster Xeon 2.4 GHz - Quadrics/ 2304 | 5694.00 11060.00 | Lawrence Livermore National Laboratory USA/2002
6 | Hewlett-Packard AlphaServer SC ES45/1 GHz/ 3016 | 4463.00 6032.00 | Pittsburgh Supercomputing Center USA/2001
7 | Hewlett-Packard AlphaServer SC ES45/1 GHz/ 2560 | 3980.00 5120.00 | Commissariat a l'Energie Atomique (CEA) France/2001
8 | HPTi Aspen Systems, Dual Xeon | 3337.00 6758.00 | Forecast Systems Laboratory - NOAA
Aspen Systems
Introducing the E2 Cluster Solution
Appro Server Solutions
Dual Intel® Xeon™ Processor
APPRO
XEON
AMD Processors
Athlon PROCESSOR MP
promicro
atipa technologies

fin 2006 : le 1er est devenu 14ème...



PRESENTED BY

UNIV. OF MANNHEIM

UNIV. OF TENNESSEE

NERSC/LBNL

TOP500 List - November 2006 (1-100)

R_{max} and R_{peak} values are in GFlops. For more details about other fields, check the TOP500 description.

next

Rank	Site	Computer	Processors	Year	R_{max}	R_{peak}
1	DOE/NNSA/LLNL United States	BlueGene/L - eServer Blue Gene Solution IBM	131072	2005	280600	367000
2	NNSA/Sandia National Laboratories United States	Red Storm - Sandia/ Cray Red Storm, Opteron 2.4 GHz dual core Cray Inc.	26544	2006	101400	127411
3	IBM Thomas J. Watson Research Center United States	BGW - eServer Blue Gene Solution IBM	40960	2005	91290	114688
4	DOE/NNSA/LLNL United States	ASC Purple - eServer pSeries p5 575 1.9 GHz IBM	12208	2006	75760	92781
5	Barcelona Supercomputing Center Spain	MareNostrum - BladeCenter JS21 Cluster, PPC 970, 2.3 GHz, Myrinet IBM	10240	2006	62630	94208
6	NNSA/Sandia National Laboratories United States	Thunderbird - PowerEdge 1850, 3.6 GHz, Infiniband Dell	9024	2006	53000	64972.8
7	Commissariat a l'Energie Atomique (CEA) France	Tera-10 - NovaScale 5160, Itanium2 1.6 GHz, Quadrics Bull SA	9968	2006	52840	63795.2
8	NASA/Ames Research Center/NAS United States	Columbia - SGI Altix 1.5 GHz, Voltaire Infiniband SGI	10160	2004	51870	60960
9	GSIC Center, Tokyo Institute of Technology Japan	TSUBAME Grid Cluster - Sun Fire x4600 Cluster, Opteron 2.4/2.6 GHz and ClearSpeed Accelerator, Infiniband NEC/Sun	11088	2006	47380	82124.8
10	Oak Ridge National Laboratory United States	Jaguar - Cray XT3, 2.6 GHz dual Core Cray Inc.	10424	2006	43480	54204.8
11	Maui High-Performance Computing Center (MHPCC) United States	Jaws - PowerEdge 1955, 3.0 GHz, Infiniband Dell	5200	2006	42390	62400
12	Texas Advanced Computing Center/Univ. of Texas United States	Lonestar - PowerEdge 1955, 2.66 GHz, Infiniband Dell	5200	2006	41460	55473.6
13	Forschungszentrum Juelich (FZJ) Germany	JUBL - eServer Blue Gene Solution IBM	16384	2006	37330	45875
14	The Earth Simulator Center Japan	Earth-Simulator NEC	5120	2002	35860	40960
15	Atomic Weapons Establishment United Kingdom	Cray XT3, 2.6 GHz dual Core Cray Inc.	7812	2006	32500	40622
16	Cray Inc.	Cray XT4, 2.6 GHz	6696	2006	27980	34819

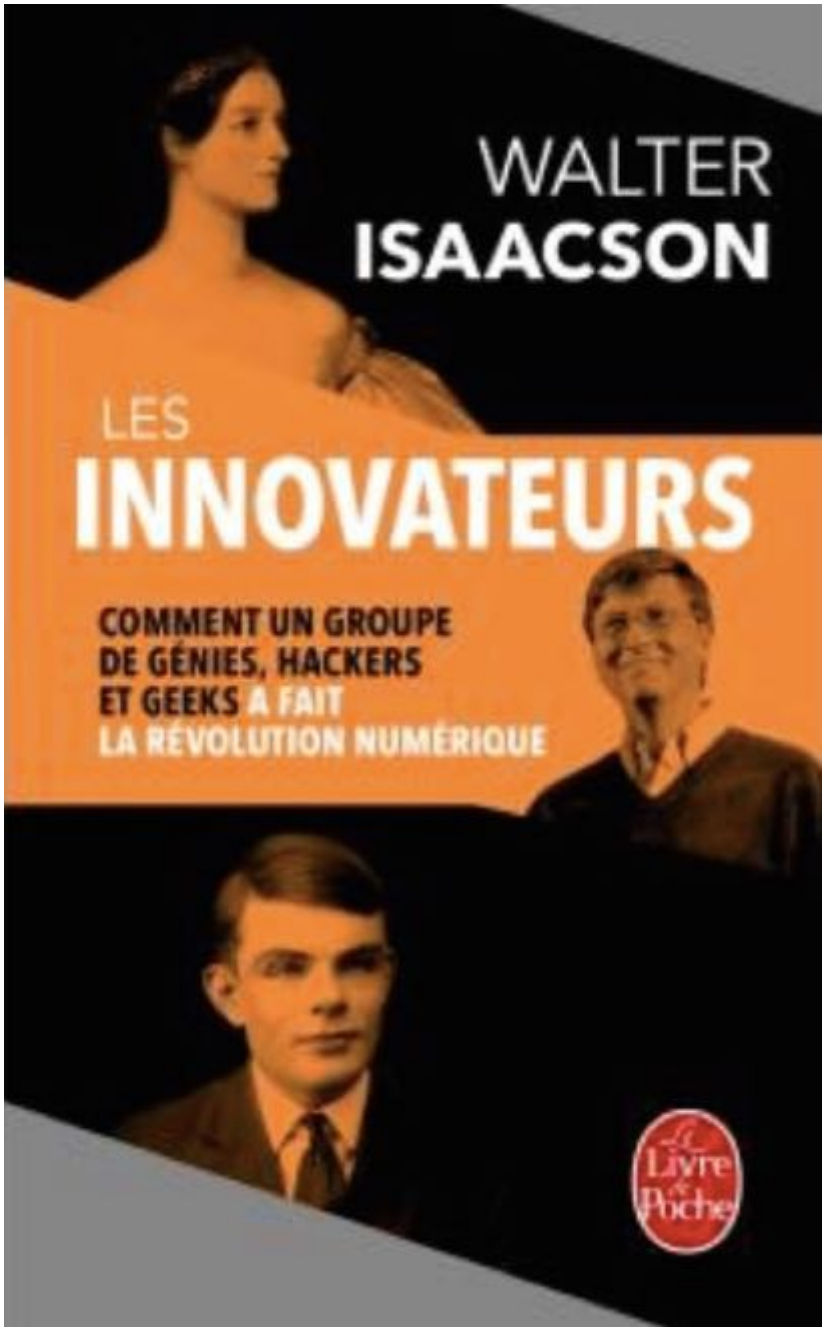
Juin 2018 : du GFlop au TFlop (et KW...)

June 2018 | TOP500 Supercomputer Sites

1-100 | 101-200 | 201-300 | 301-400 | 401-500

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , **IBM** DOE/SC/Oak Ridge National Laboratory **United States**	2,282,544	122,300.0	187,659.3	8,806
2	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway , **NRCPC** National Supercomputing Center in Wuxi **China**	10,649,600	93,014.6	125,435.9	15,371
3	**Sierra** - IBM Power System S922LC, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , **IBM** DOE/NNSA/LLNL **United States**	1,572,480	71,610.0	119,193.6	
4	**Tianhe-2A** - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000 , **NUDT** National Super Computer Center in Guangzhou **China**	4,981,760	61,444.5	100,678.7	18,482
5	**AI Bridging Cloud Infrastructure (ABCI)** - PRIMERGY CX2550 M4, Xeon Gold 6148 20C 2.4GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR , **Fujitsu** National Institute of Advanced Industrial Science and Technology (AIST) **Japan**	391,680	19,880.0	32,576.6	1,649
6	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 , **Cray Inc.** Swiss National Supercomputing Centre (CSCS) **Switzerland**	361,760	19,590.0	25,326.3	2,272
7	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x , **Cray Inc.** DOE/SC/Oak Ridge National Laboratory **United States**	560,640	17,590.0	27,112.5	8,209
8	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom , **IBM** DOE/NNSA/LLNL **United States**	1,572,864	17,173.2	20,132.7	7,890
9	**Trinity** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , **Cray Inc.** DOE/NNSA/LANL/SNL **United States**	979,968	14,137.3	43,902.6	3,844
10	**Cori** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , **Cray Inc.** DOE/SC/LBNL/NERSC **United States**	622,336	14,014.7	27,880.7	3,939

(à lire, 9€)


WALTER
ISAACSON
LES
INNOVATEURS
COMMENT UN GROUPE
DE GÉNIES, HACKERS
ET GEEKS A FAIT
LA RÉVOLUTION NUMÉRIQUE
Le Livre de Poche

PARTIE 3 : l'environnement Processing

http://processing.org

Exemples de codes affichant « hello world »

Langage machine

```
10111010 00010000
00000001 10110100
00001001 11001101
00100001 00110000
11100100 11001101
00010110 10111000
00000000 01001100
11001101 00100001
01001000 01100101  (he)
01101100 01101100  (ll)
01101111 00100000  (o )
01010111 01101111  (wo)
01110010 01101100  (rl)
01100100 00100001  (d!)
00100100           ($)
```

Assembleur x86 sous DOS

```
cseg    segment
        assume  cs:cseg, ds:cseg
        org     100h
main    proc
        mov     dx, offset Message
        mov     ah, 9   ; Fonction DOS : afficher une chaîne.
        int     21h     ; Appel à MS-DOS
        ret             ; Fin du programme COM
main    endp

Message db      "Hello world!$"

cseg    ends
        end     main
```

C

```c
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
    return 0;
}
```

C++

```cpp
#include <iostream>

int main()
{
    std::cout << "Hello, new world!" << std::endl;
    return 0;
}
```

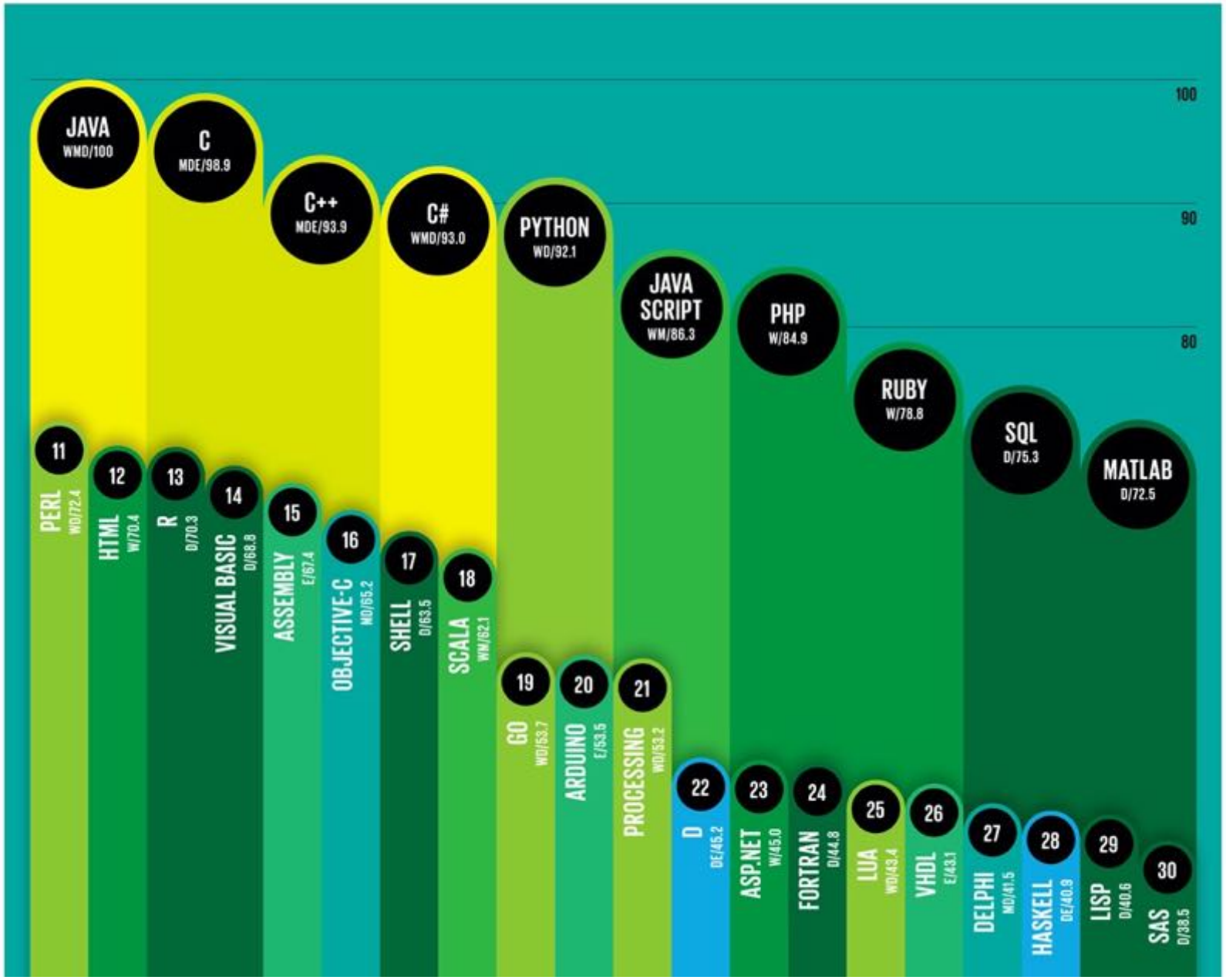
java

```java
/* Affichage console */
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

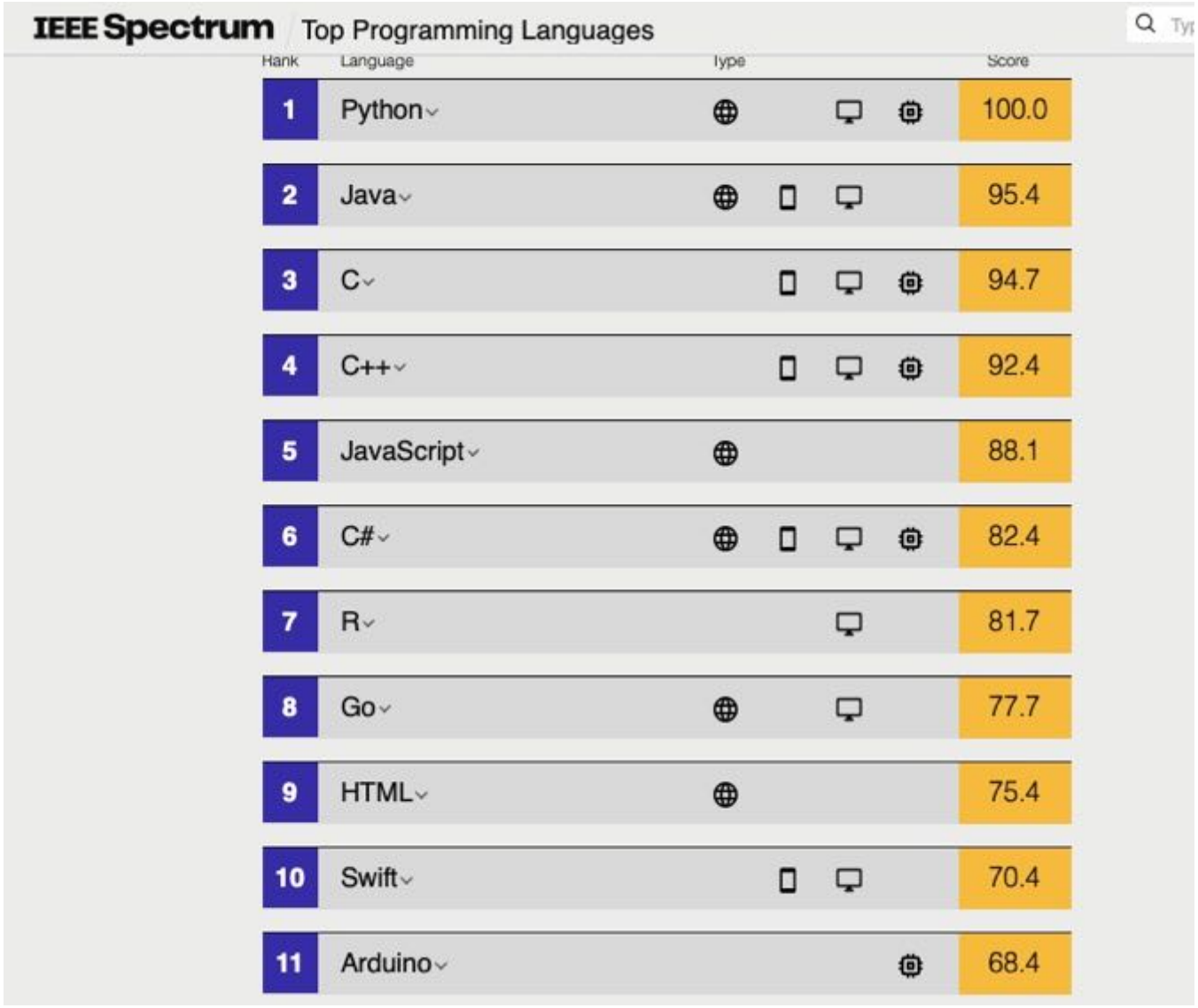
https://fr.wikipedia.org/wiki/Liste_de_programmes_Hello_world

A VOIR : https://rosettacode.org/

Popularité des langages (d'après IEEE Spectrum, sept 2014)



Même étude en sept. 2021



https://spectrum.ieee.org/top-programming-languages/#toggle-gdpr

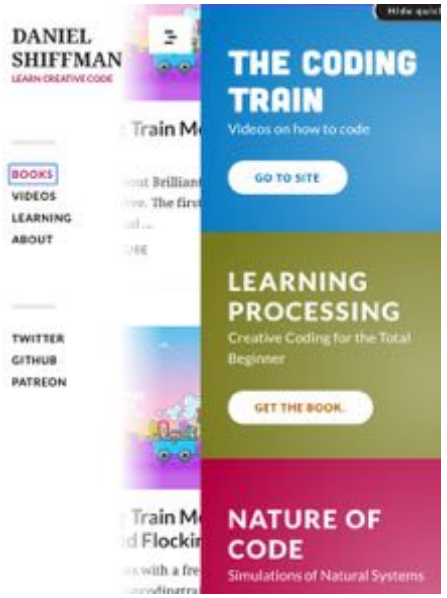
Processing : les fondateurs

http://www.multimedialab.be/cours/logiciels/processing.htm

http://reas.com/
(ucla)

http://benfry.com/
(mit medialab)

https://shiffman.net
(nyu)



REV 0068 - 2 février 2004

REV 0133 - 26 octobre 2007

La 1.0 en nov. 2008

la 2.0 en été 2013

la 3.0 en automne 2014

la 4.0 en beta août 2021 !!!!

professionals. We stand by our mission statement.

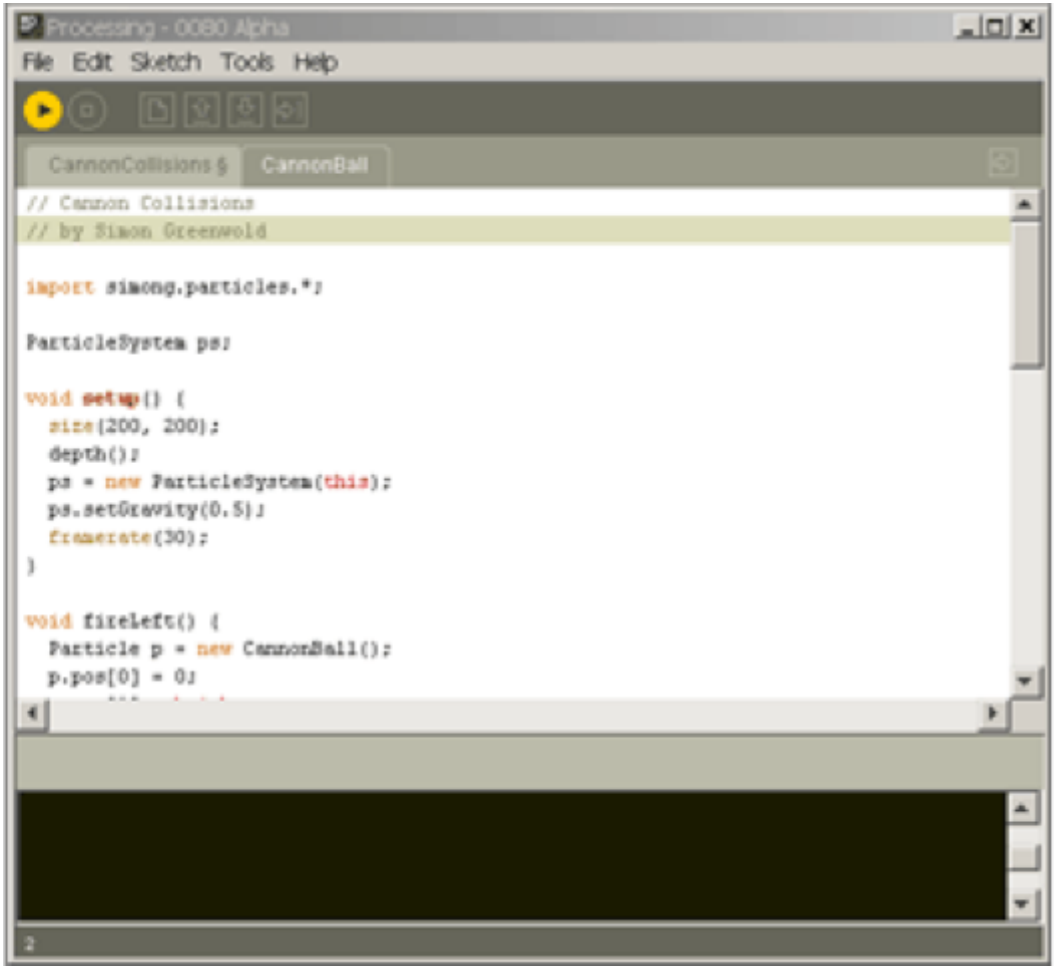
Processing seeks to ruin the careers of talented designers by tempting them away from their usual tools and into the world of programming and computation. Similarly, the project is designed to turn engineers and computer scientists to less gainful employment as artists and designers.

The Processing 2.0 release focuses on faster graphics, new infrastructure for

http://processing.org/overview/

Processing Development Environment (PDE)

Display Window



Menu

Toolbar

Tabs

Text Editor

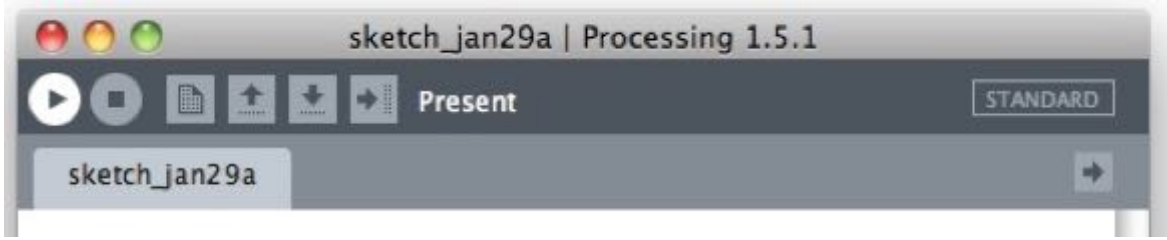
Message Area

Text Area

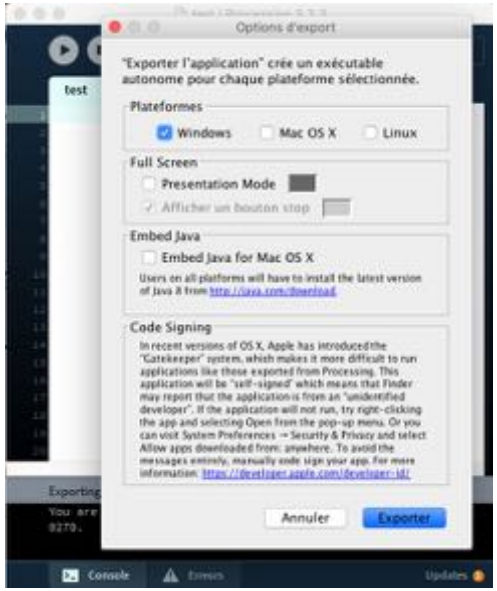
http://processing.org/reference/environment/index.html

Execution du programme : plusieurs possibilités

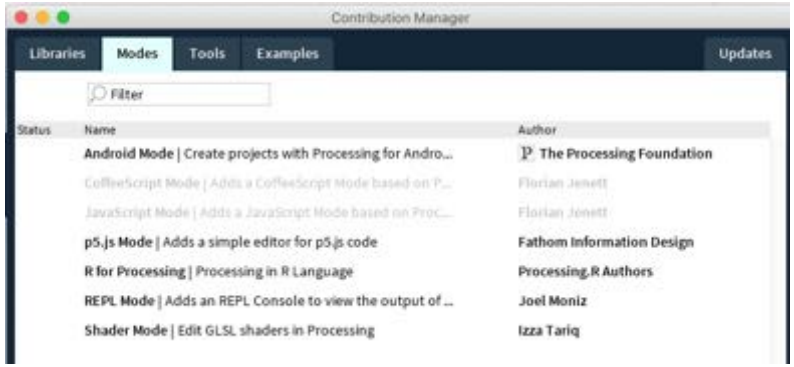
interne
(root ou rootless)



export
(multi-os)

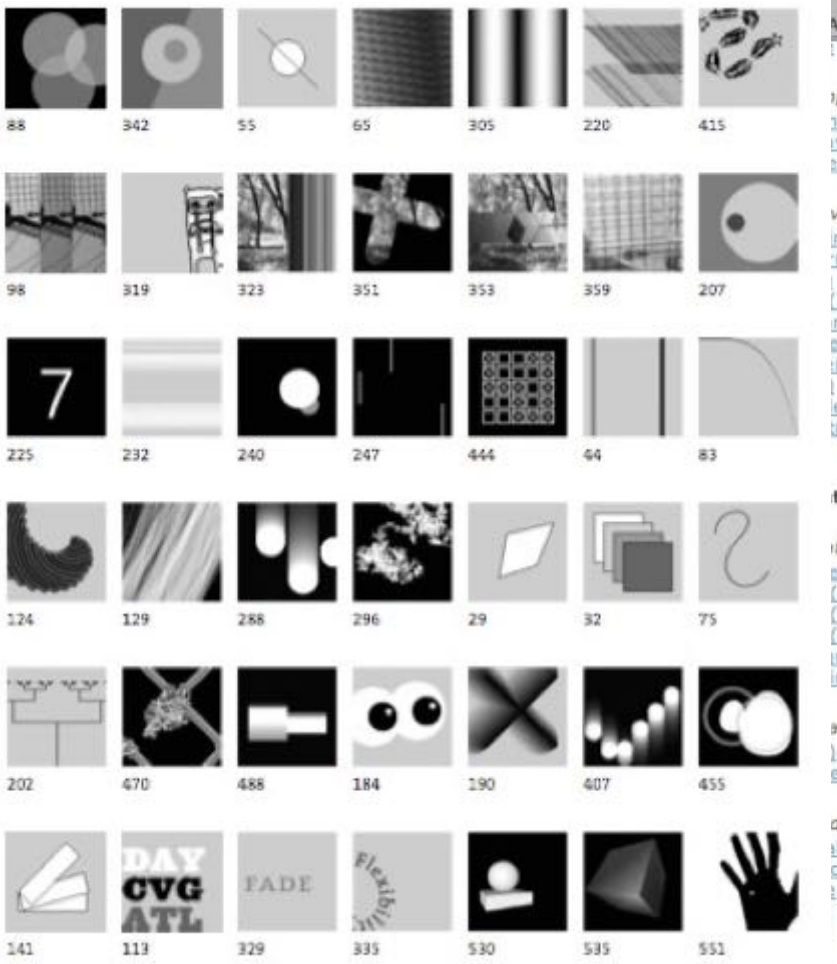


choix du langage
("mode")

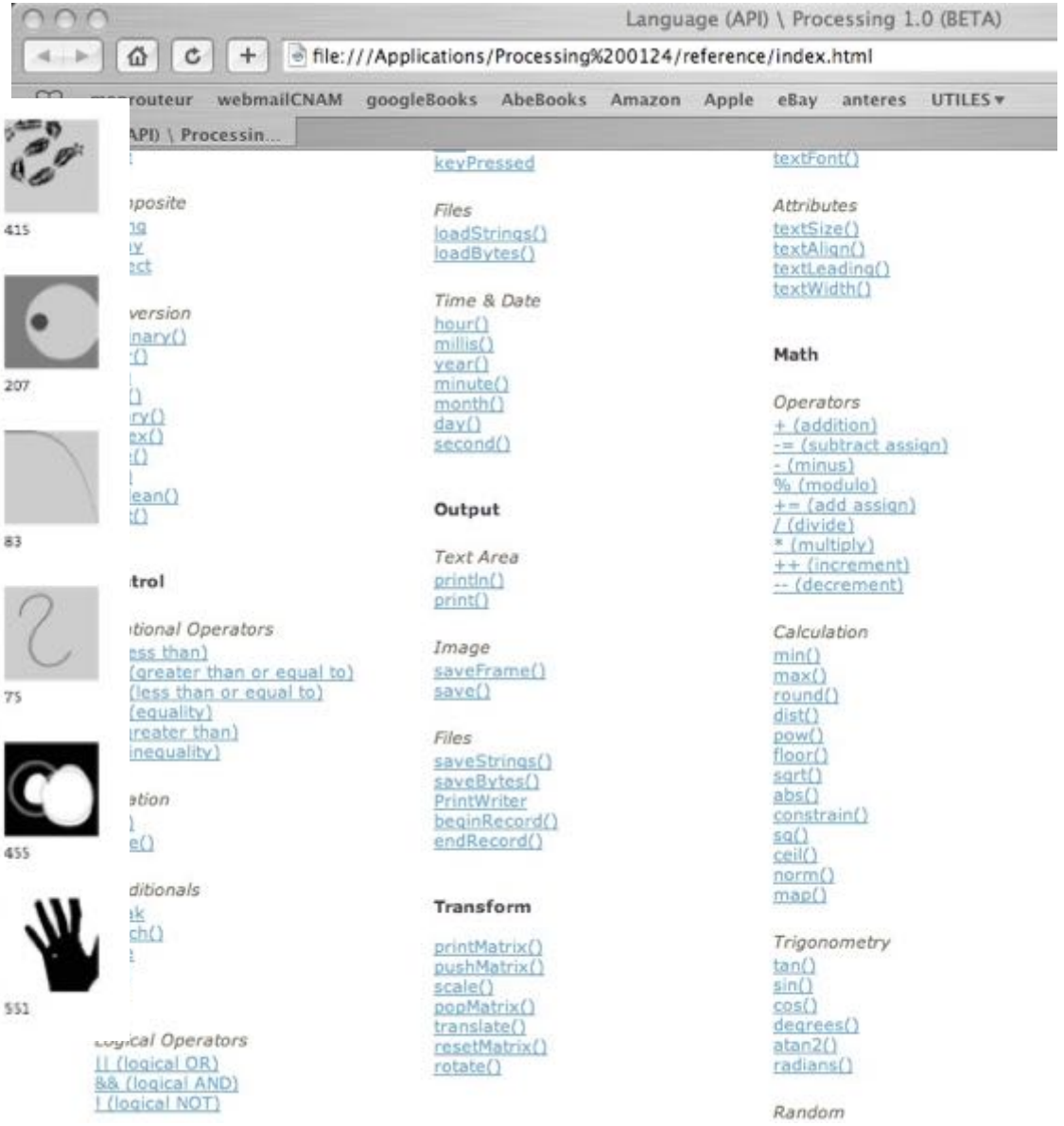


Le langage (par l'exemple…)

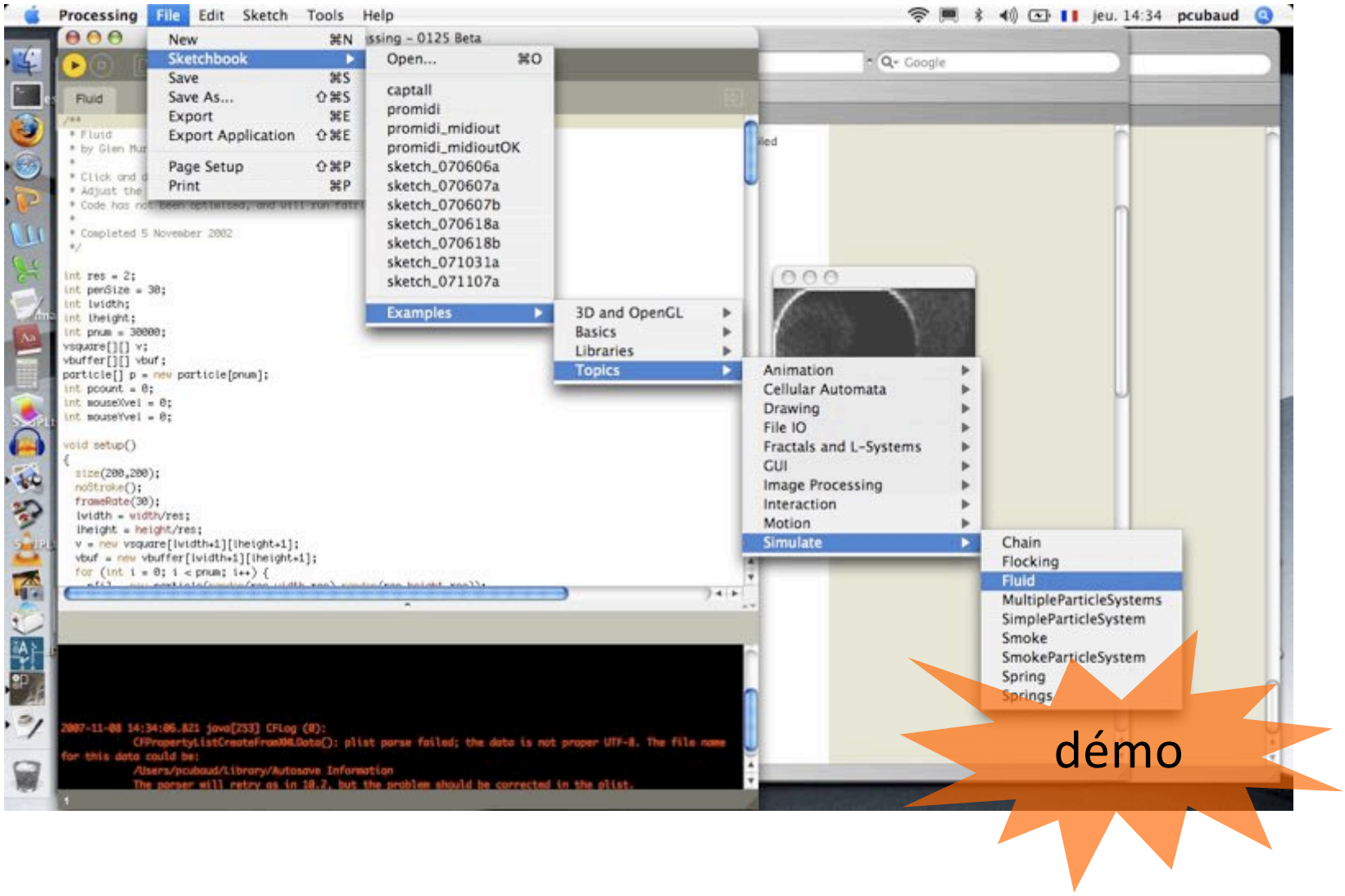
- dans le livre



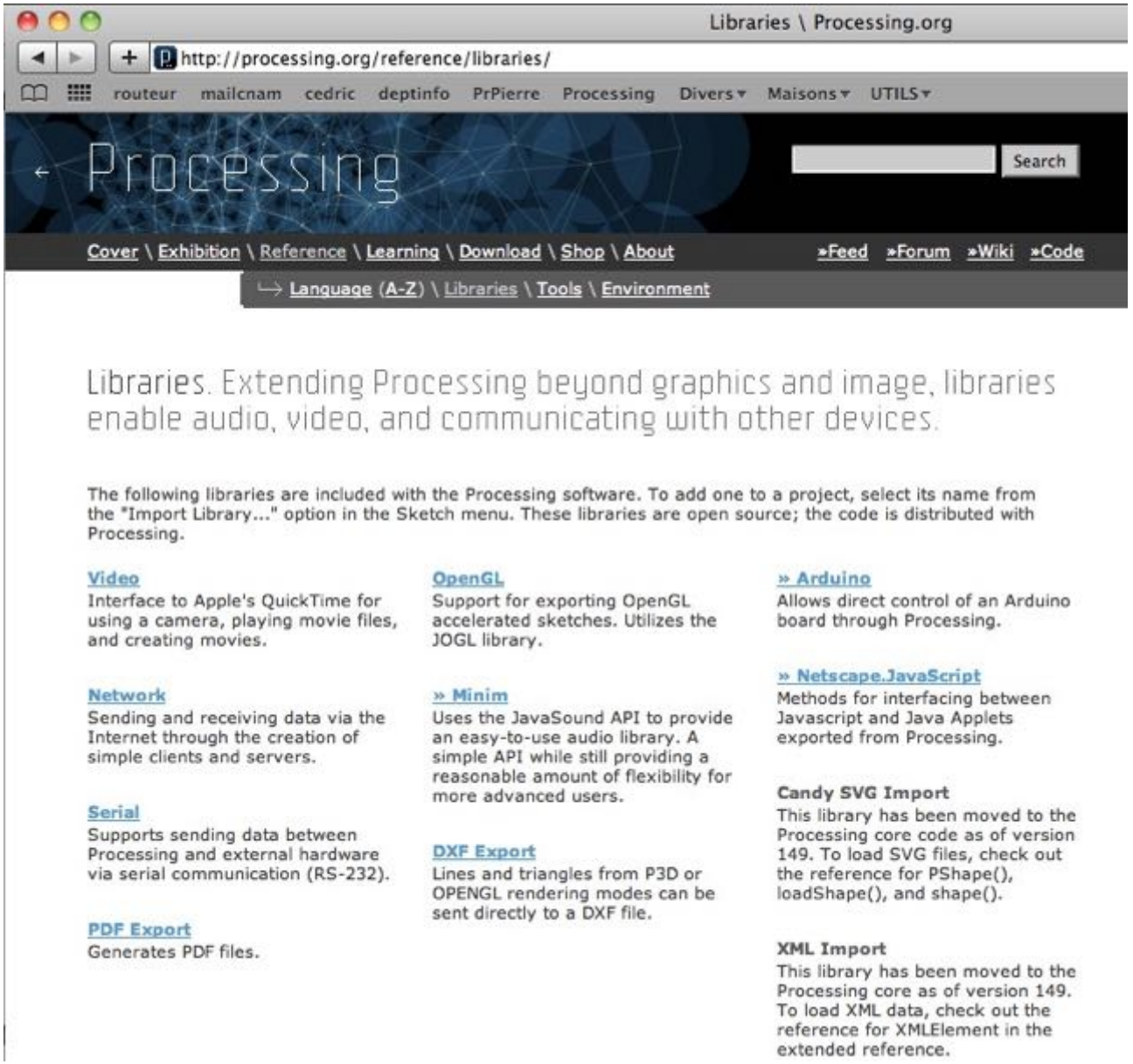
- manuel de ref.



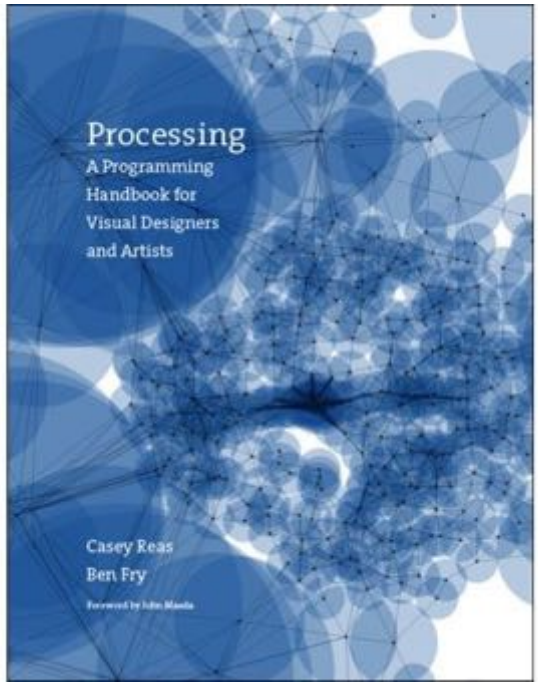
Un exemple de sketch dans le dossier examples, parmi plein



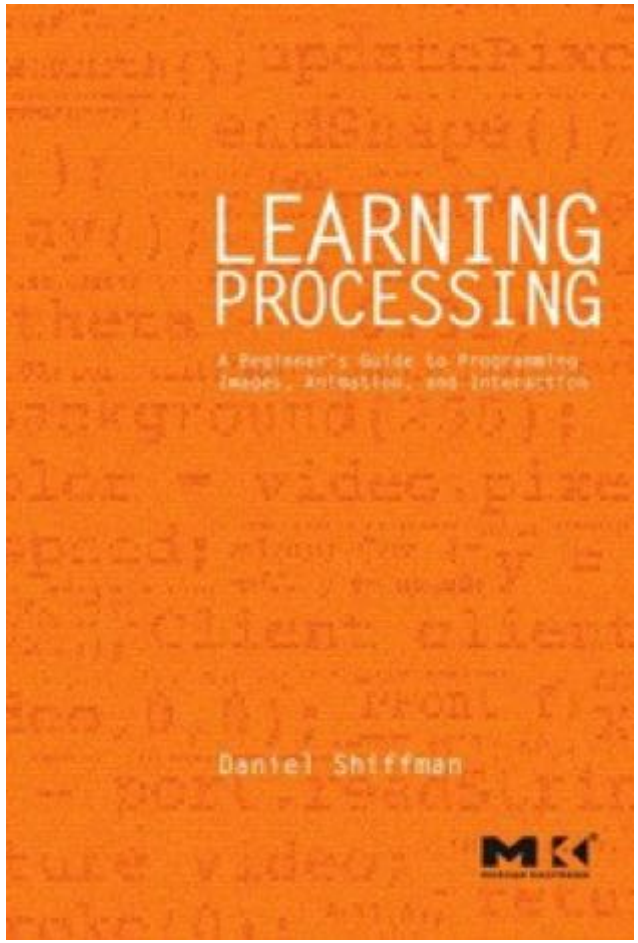
Les librairies : de base (core) ou tierces (contributors)

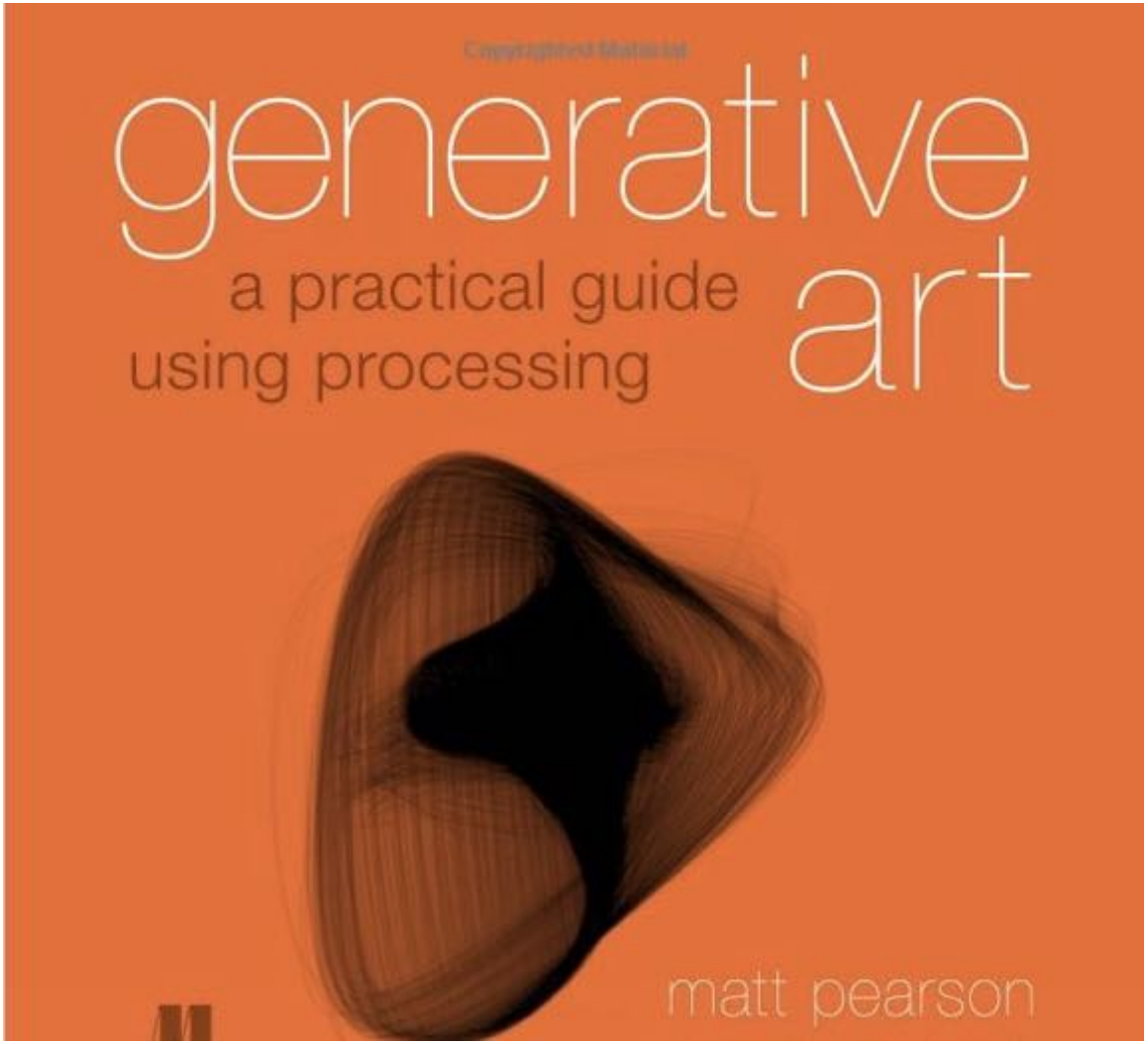


Bibliographie :

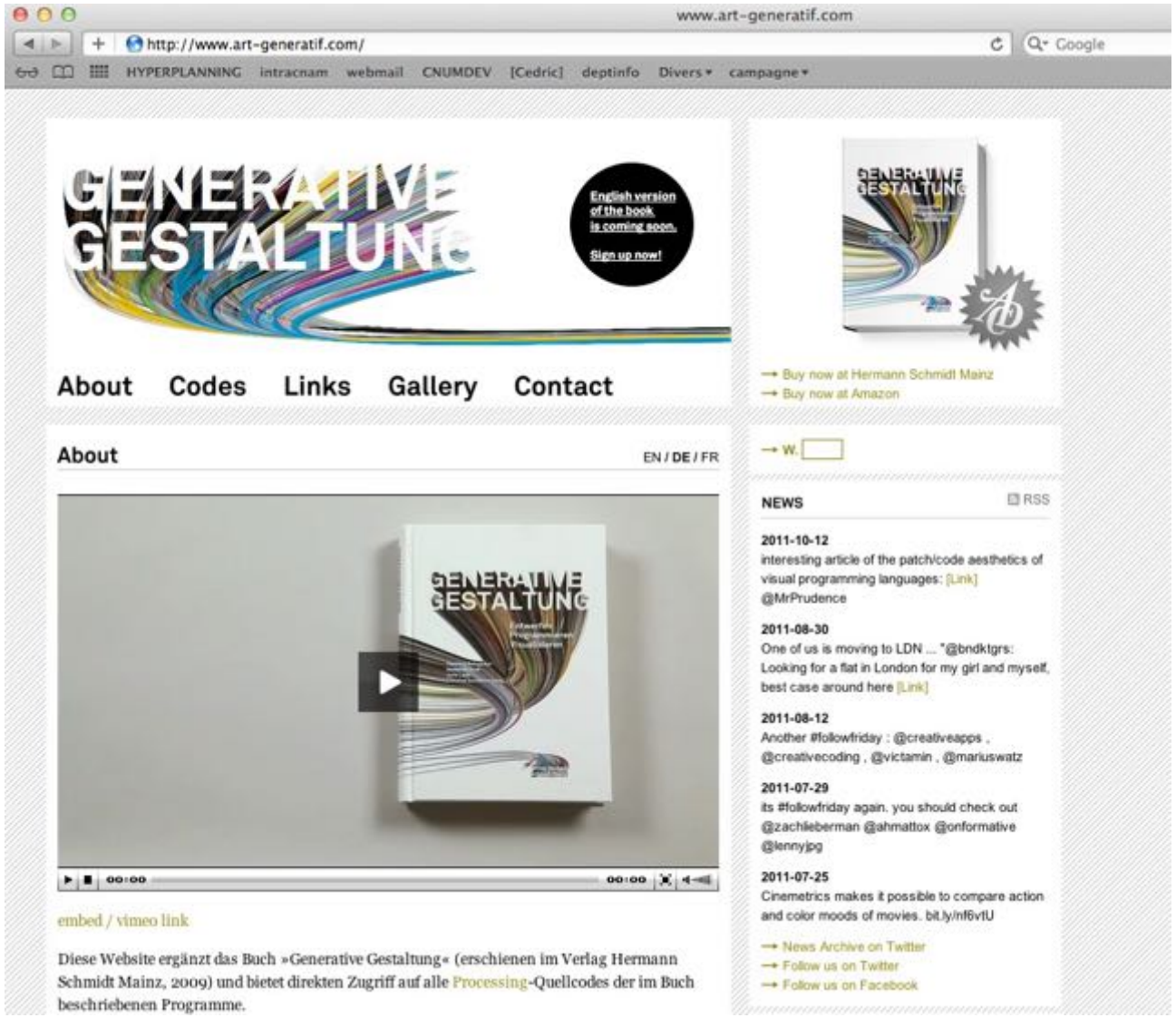


736 p.
MIT Press (35 euros)

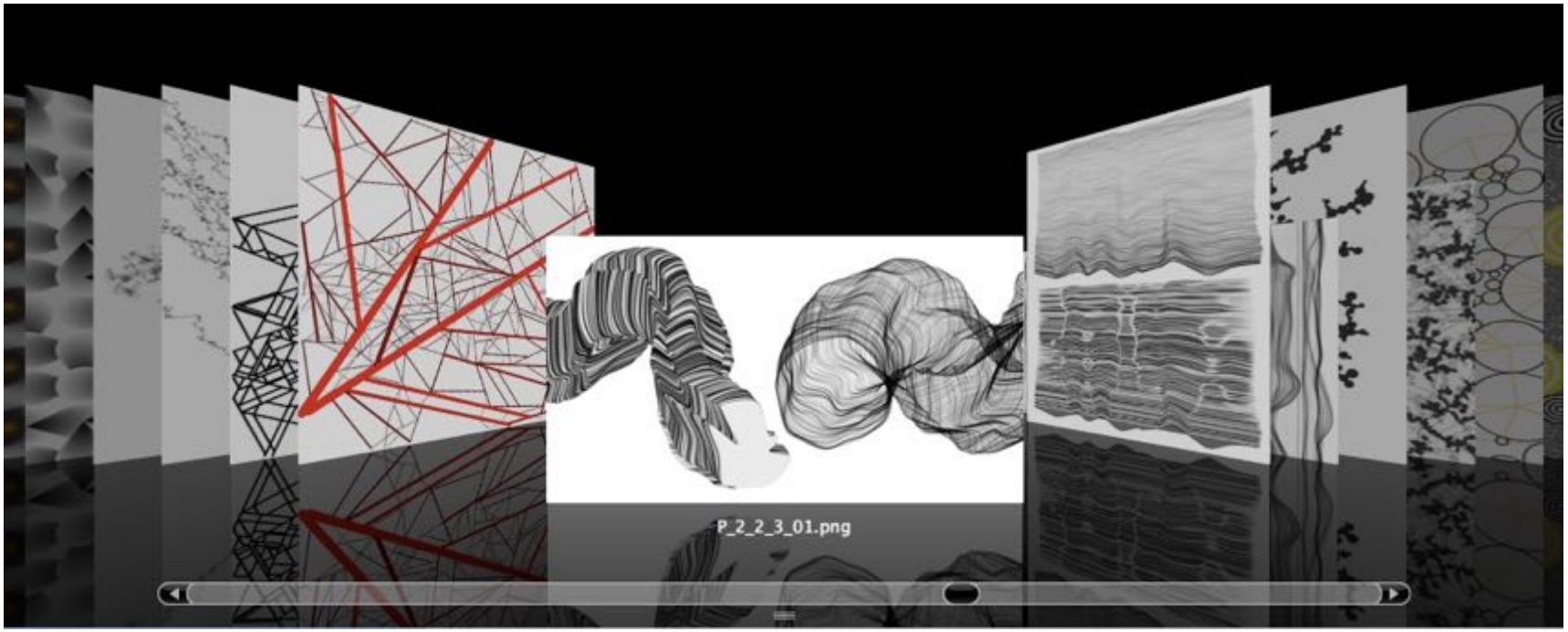



generative art
a practical guide
using processing
matt pearson

300 p. 28€

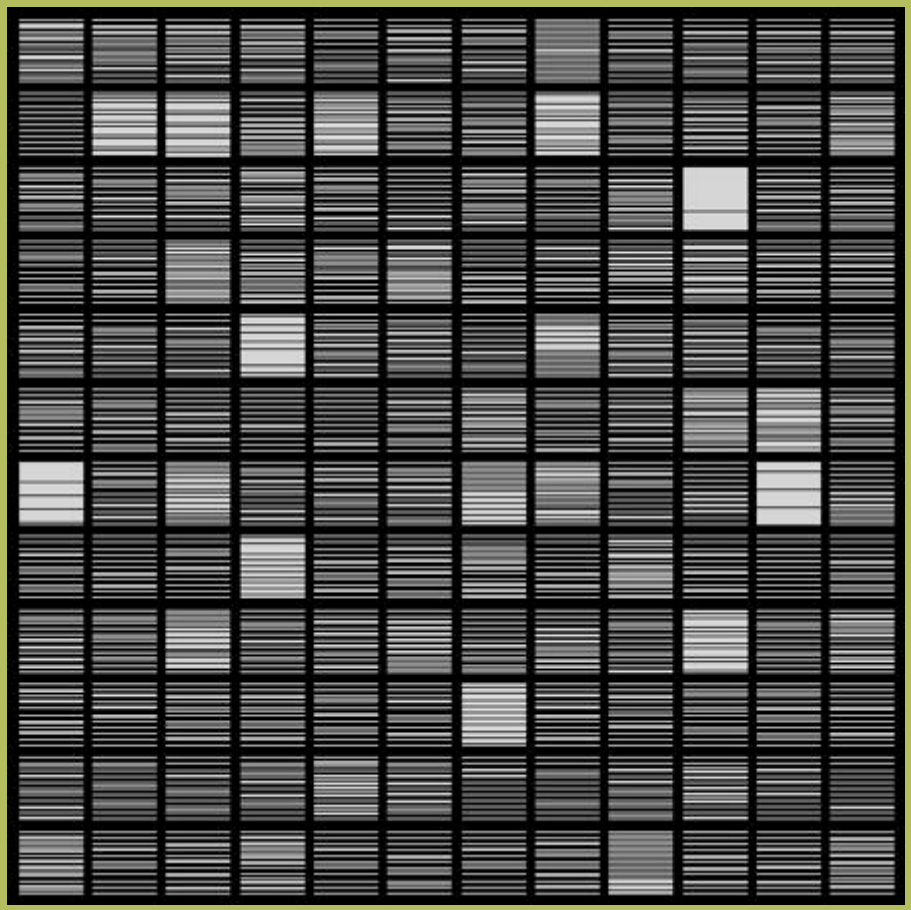


en français : Design génératif, Pyramid, 2011, 75€



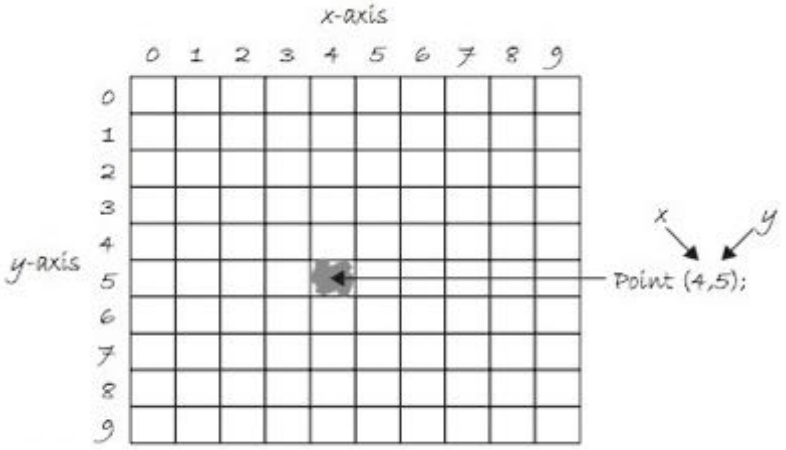
128 programmes commentés (code sur le site)

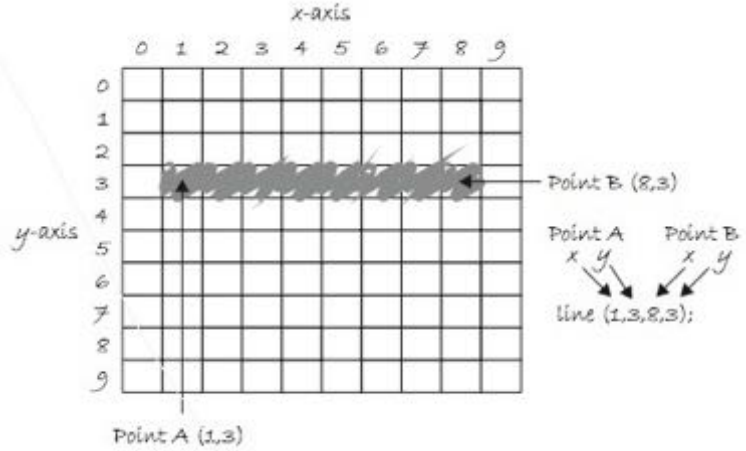
PARTIE 4. Itérations

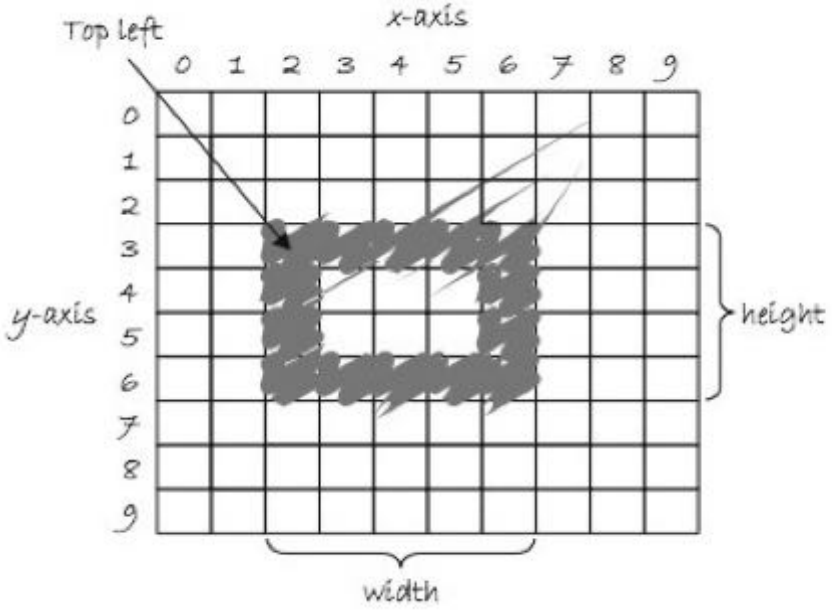


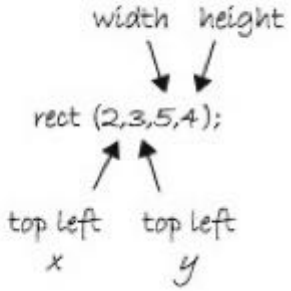
Reas&Fry, p. 152 - d'après Mark Wilson

2.1. Formes élémentaires









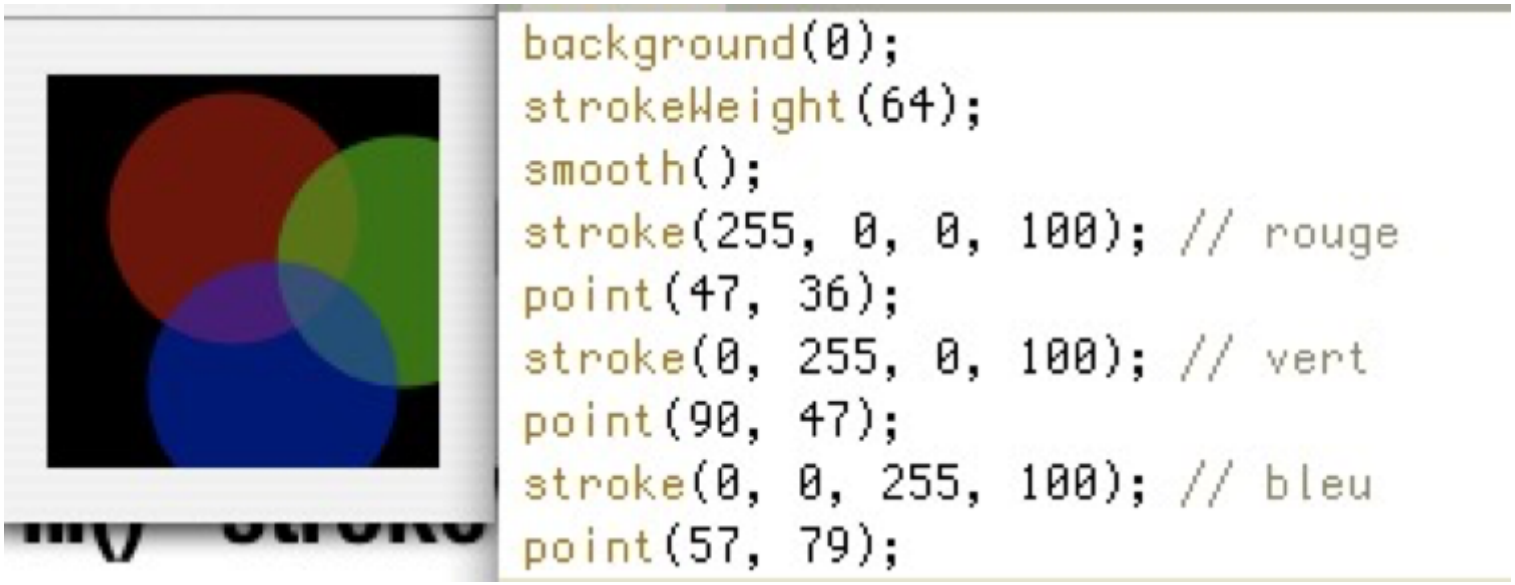
point(…)
line(…)
rect(…)
ellipse(…)
arc(…)
triangle(…)
quad(…)

http://processing.org/learning/tutorials/basics/

Les attributs de forme et de couleur

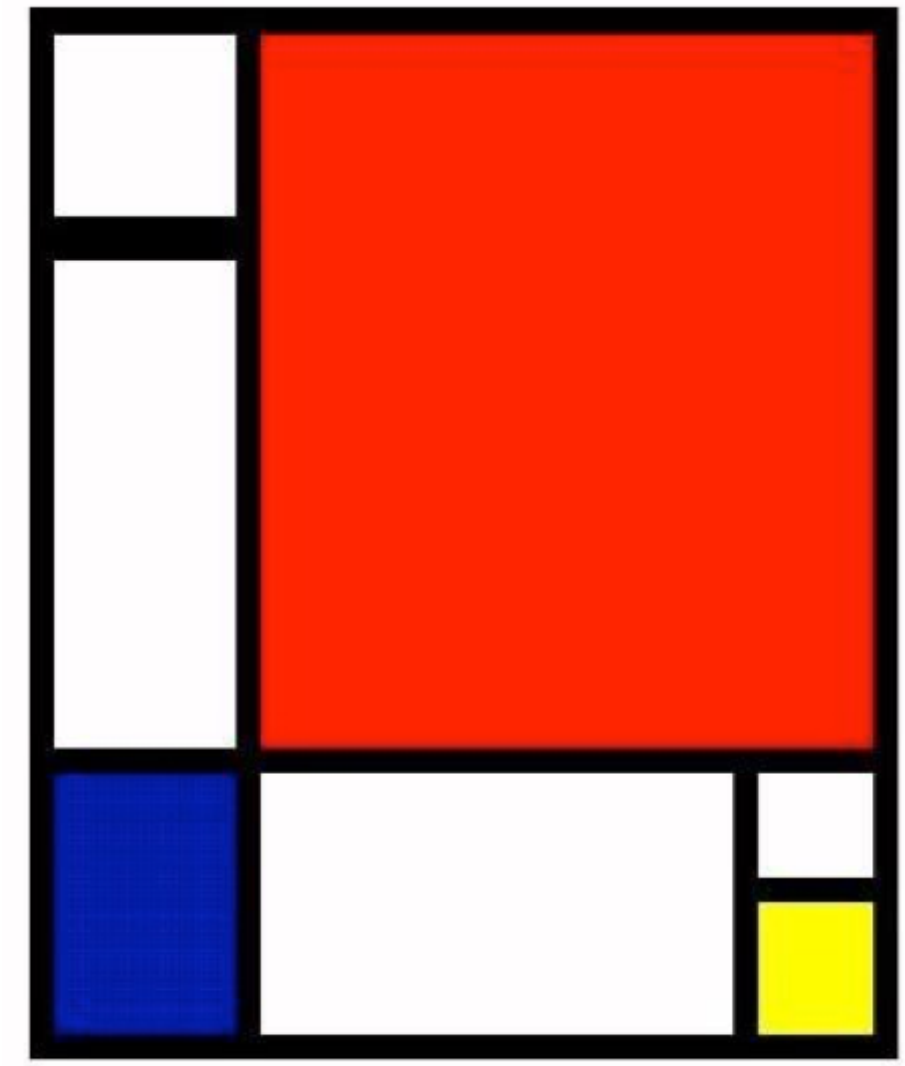
stroke(…) fill(…) noStroke() noFill()
strokeWeight(…) strokeCap(…) strokeJoin(…)

color(…) colorMode(…)
background(…)



```
background(0);
strokeWeight(64);
smooth();
stroke(255, 0, 0, 100); // rouge
point(47, 36);
stroke(0, 255, 0, 100); // vert
point(90, 47);
stroke(0, 0, 255, 100); // bleu
point(57, 79);
```

hommage à Mondrian



http://art.and.facts.site.free.fr/Site/artregions/images/stras/mondrian.jpg

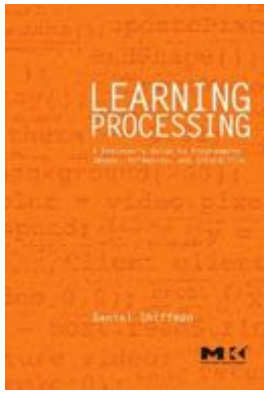
Code "solution" des exercices :

http://cedric.cnam.fr/~cubaud/mjv102codes.zip

(bis)

```
size(400,500);

stroke(0);
strokeWeight(10);
strokeCap(SQUARE);
//le fond
background(255);
// le grand carre rouge
fill(255,0,0);
rect(100,0,300,300);
// le rectangle bleu
fill(0,0,255);
rect(0,300,100,200);
// le carre jaune
fill(255,255,0);
rect(300,400,100,100);
// le trait vertical pres du carré jaune
line(300,300,300,400);
// le trait horizontal en haut
line(0,100,100,100);
// le bord
noFill();
rect(0,0,400,500);

save("mondrian1.png");
```

(mondrian1.pde)



Zoog de Schiffman

```
size(640, 360);
background(255);
ellipseMode(CENTER);
rectMode(CENTER);

// Body
stroke(0);
fill(150);
rect(320, 190, 20, 100);

// Head
fill(255);
ellipse(320, 160, 60, 60);

// Eyes
fill(0);
ellipse(301, 160, 16, 32);
ellipse(339, 160, 16, 32);

// Legs
stroke(0);
line(310, 240, 300, 250);
line(330, 240, 340, 250);
```

(zoog.pde)

Les commentaires en Java

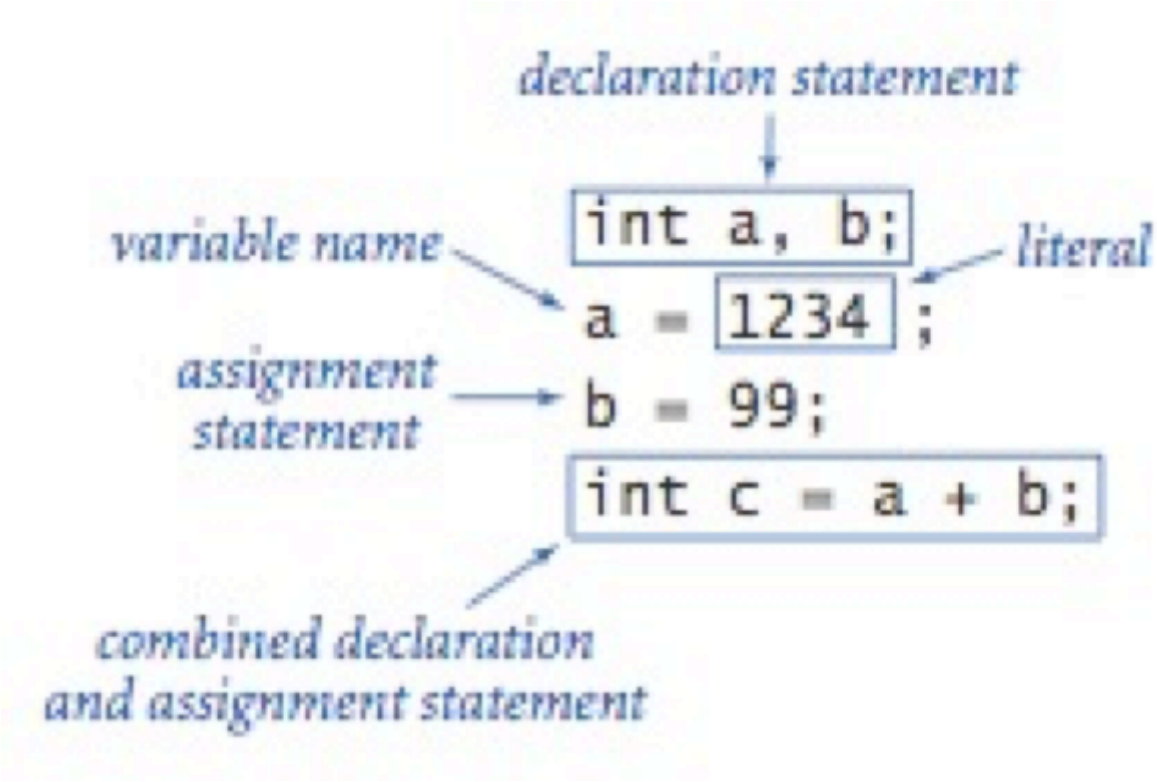
Il est très important de commenter ses programmes :
pour les autres ou pour soi-même plus tard

```
// commentaire sur une ligne

/*
 commentaires
 sur plusieurs lignes
*/

/**
 commentaire qui va etre reconnu par javadoc
*/
```

Il existe aussi des outils pour l'annotation (@author etc)

2.2 Les variables en Java

- doivent être déclarées avant d'être utilisées dans des calculs
- ont un type associé (nombre, texte, booléen) non modifiable
- n'ont pas de valeur initiale par défaut
- peuvent être des constantes (comme PI) : mot clé final



Les valeurs littérales en Java

- Un entier est représenté classiquement :

```
3       56       987
```

- Un réel est représenté avec la notation pointée :

```
3.1415
```

- Un caractère est représenté entre quotes pour le distinguer de tout autre identificateur :

```
'A'    '7'      '$'
```

- Une chaîne de caractères est représentée entre guillemets pour la distinguer de tout autre identificateur ou mot clé :

```
"Hello World "
```

- Un booléen est représenté par les symboles :

```
true, false
```

Exemple : Belle Marquise (Molière, Bourgeois gentilhomme)

bellemarquise

```
String a="Belle Marquise ";
String b="Vos beaux yeux ";
String c="Mourir d'amour ";
String d="Me font ";

println(a+b+c+d);
println(a+b+d+c);
println(a+c+b+d);
println(a+c+d+b);
println(a+d+b+c);
println(a+d+c+b);
```

```
Display 0 does not exist, using the default display instead.
Belle Marquise Vos beaux yeux Mourir d'amour Me font
Belle Marquise Vos beaux yeux Me font Mourir d'amour
Belle Marquise Mourir d'amour Vos beaux yeux Me font
Belle Marquise Mourir d'amour Me font Vos beaux yeux
Belle Marquise Me font Vos beaux yeux Mourir d'amour
Belle Marquise Me font Mourir d'amour Vos beaux yeux
```

- "additionner" des chaines = mettre bout à bout = concaténer
- combien d'autres permutations ?

Identifiants autorisés en Java
- pas que pour les variables (traitements, classes etc)
- sensible à la casse ≠ CASSE ≠ CaSsE
- on peut utiliser le $ et le _ (underscore)
- on peut utiliser les accents éàç (en Unicode) : bof

Illégal : 123xp légal : a, a, xp_12, _$

Coutumes avec la casse :
data : une variable mineure, un package
Hello : une classe, un type
lireSuivant : une "grosse" variable, un traitement (méthode)
MAXWIDTH : une constante

Mauvais style :
ceciestunnomtroplongdevariable xp12

Mots reservés de Java

abstract	continue	for	new	switch
assert[Note 1]	default	goto[Note 2]	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum[Note 3]	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp[Note 4]	volatile
const[Note 2]	float	native	super	while

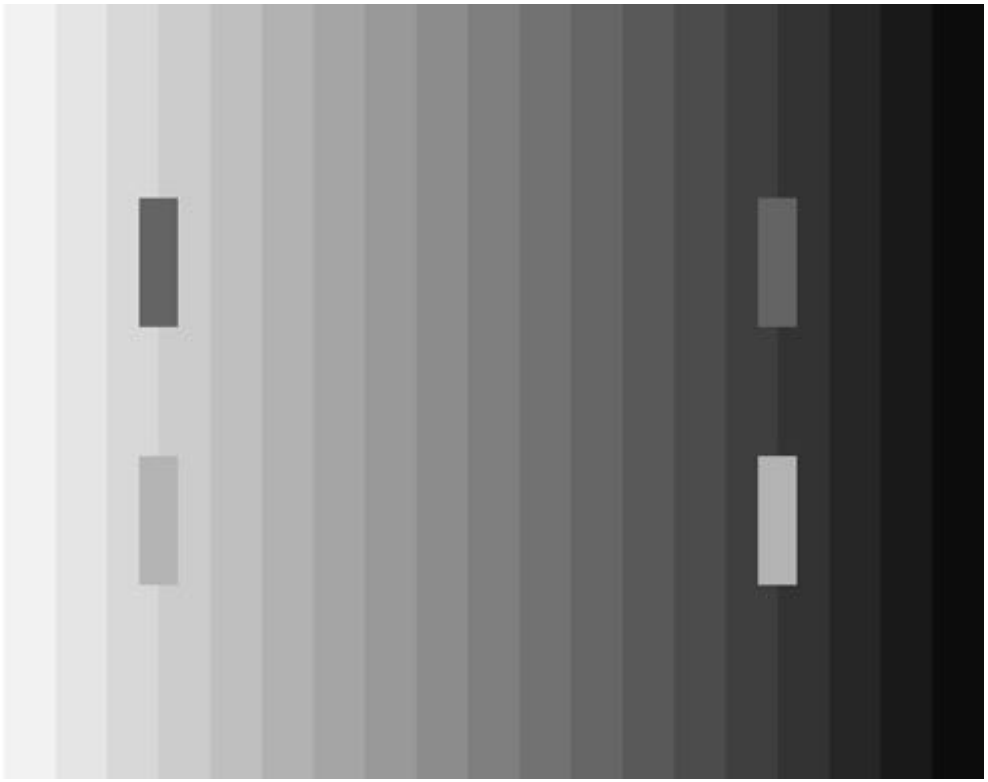
1. ^ Keyword was introduced in J2SE 1.4
2. ^ [a] [b] Keyword is not used
3. ^ Keyword was introduced in J2SE 5.0
4. ^ Keyword was introduced in J2SE 1.2

(wikipedia)

Rque : attention aussi à ceux de Processing

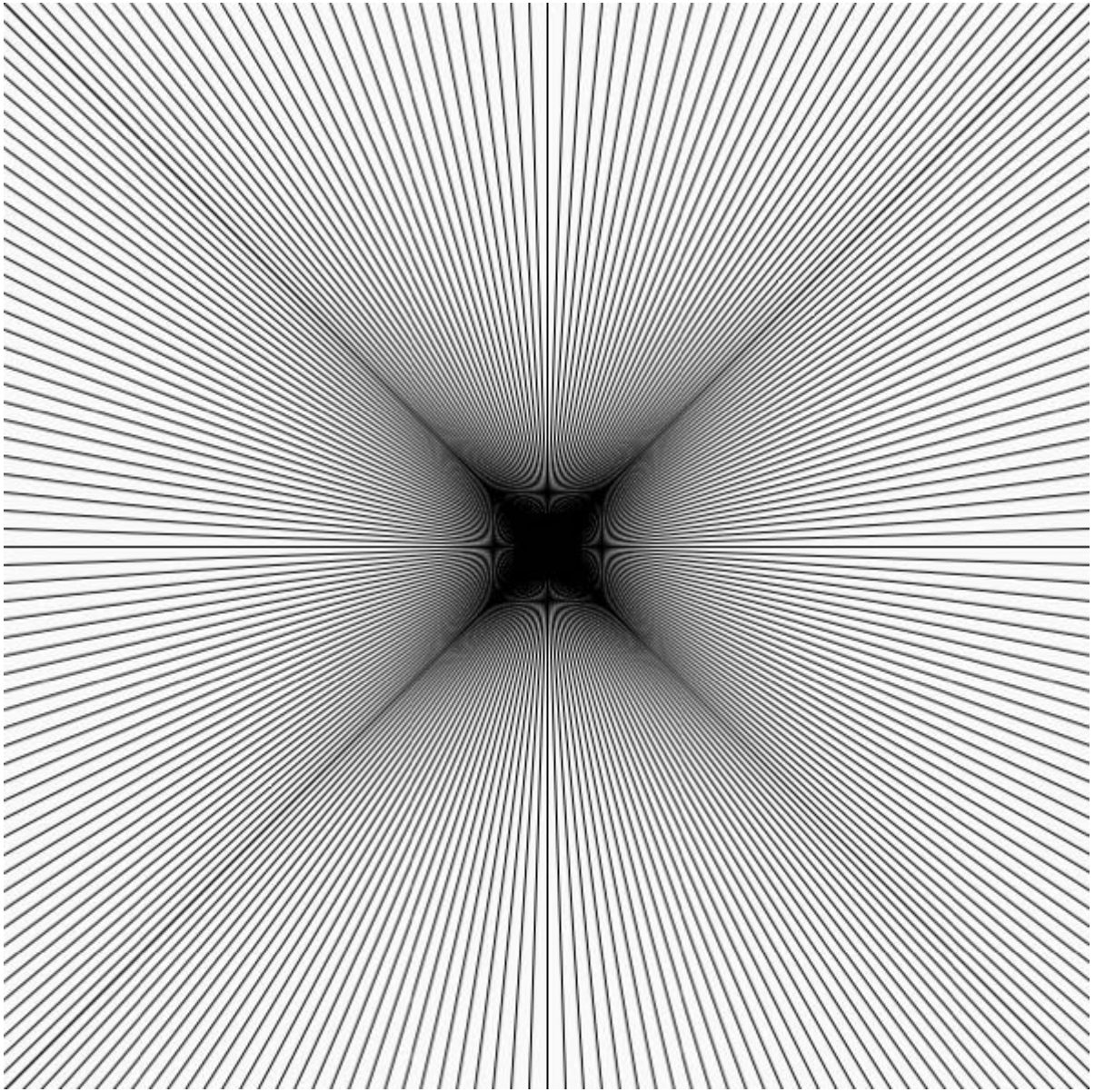
2.3. Les boucles

```
for (int i=0; i<1234567; i++) {
 println(i);
}
```



```
int N=20;
size(800,600);
noStroke();
for (int i=0; i<N;i++){
  fill(map(i,0,N,255,0));
  rect(i*width/float(N),0,width/N,height);
}
rectMode(CENTER);
fill(100);
rect(160,200,30,100);
rect(640,200,30,100);
fill(180);
rect(160,400,30,100);
rect(640,400,30,100);
save("contraste.png");
```

(contraste.pde)



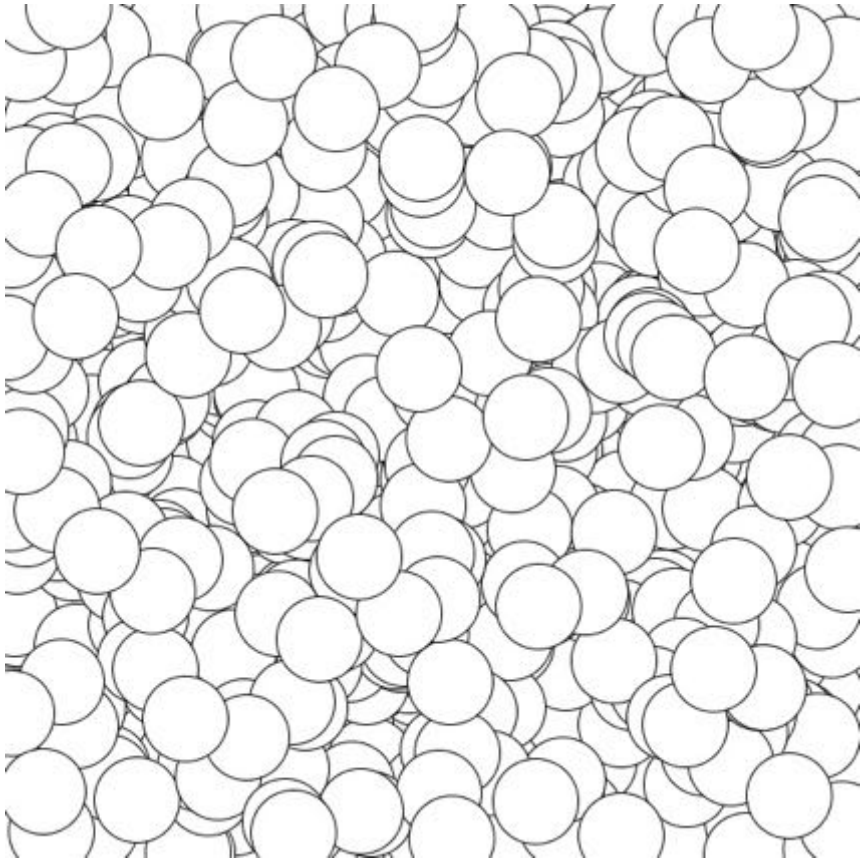
```
size(600,600);
background(250);
for (int i=0;i<=600;i+=6){
 line(i,0,600-i,600);
 line(0,i,600,600-i);
}
save("alias.png");
```

(aliasdroite.pde)

2.4. La perturbation

Bridget Riley - Circles

random(min, max)

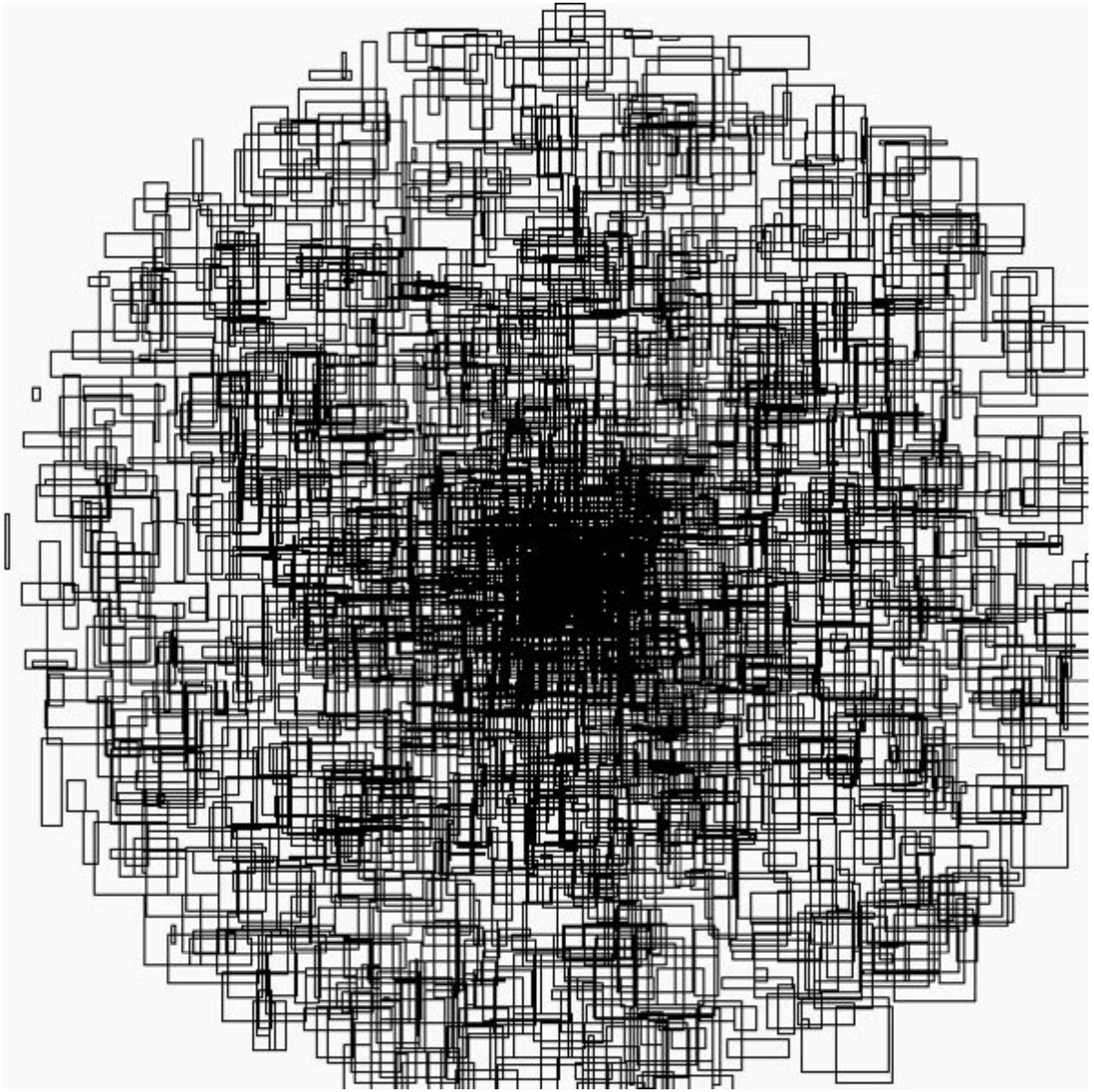


```
size(600,600);

background(250);
//noStroke();fill(0,0,128,30);

for (int i=0; i<1000; i++) {

 float x=random(0,600);
 float y=random(0,600);

 ellipse(x,y,60,60);
}

save("generatif1.png");
```

(generatif1.pde)

modifier :

- les attributs (stroke, fill)
- le nombre d'itérations
- la taille des cercles



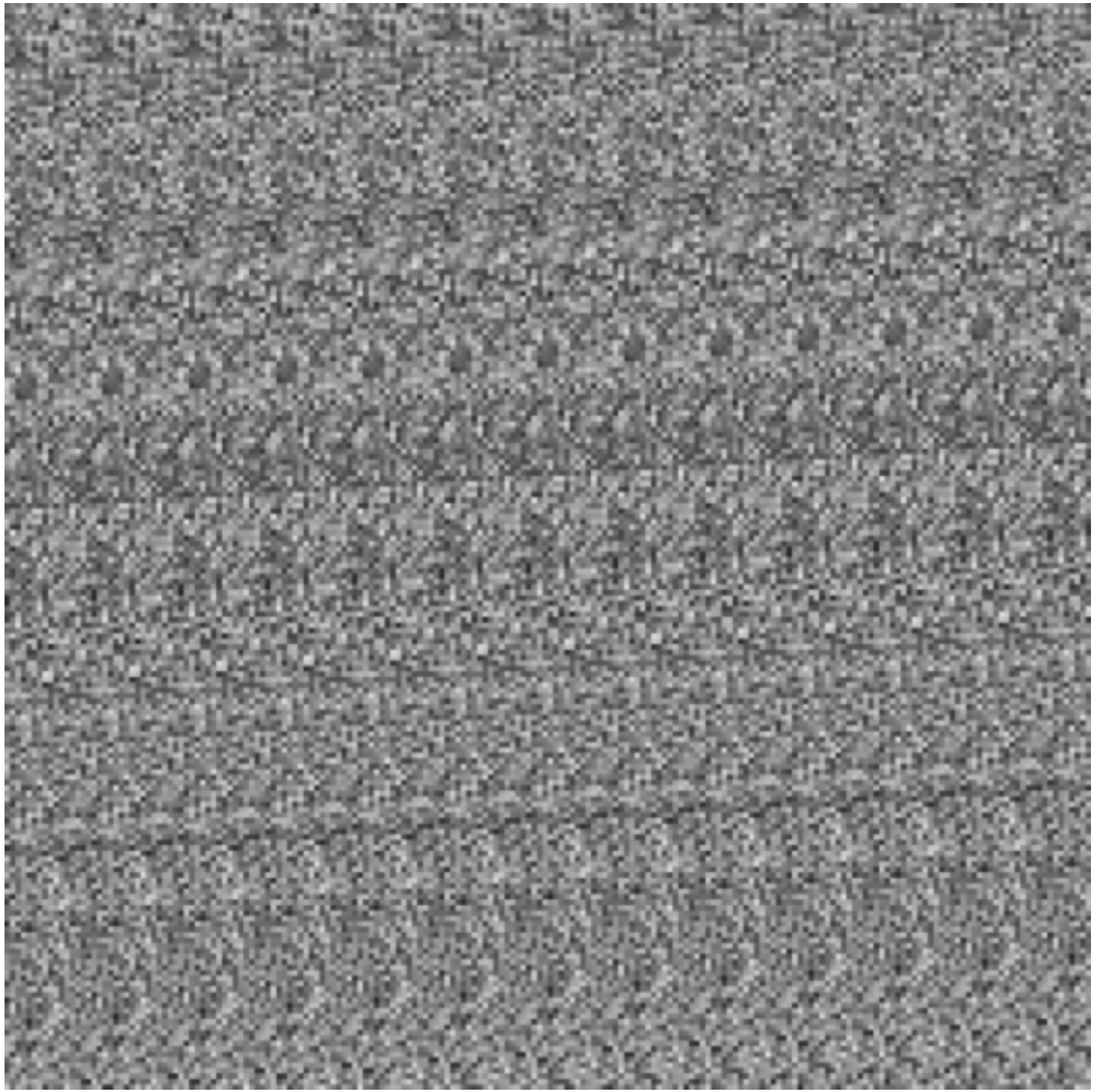

5,5t

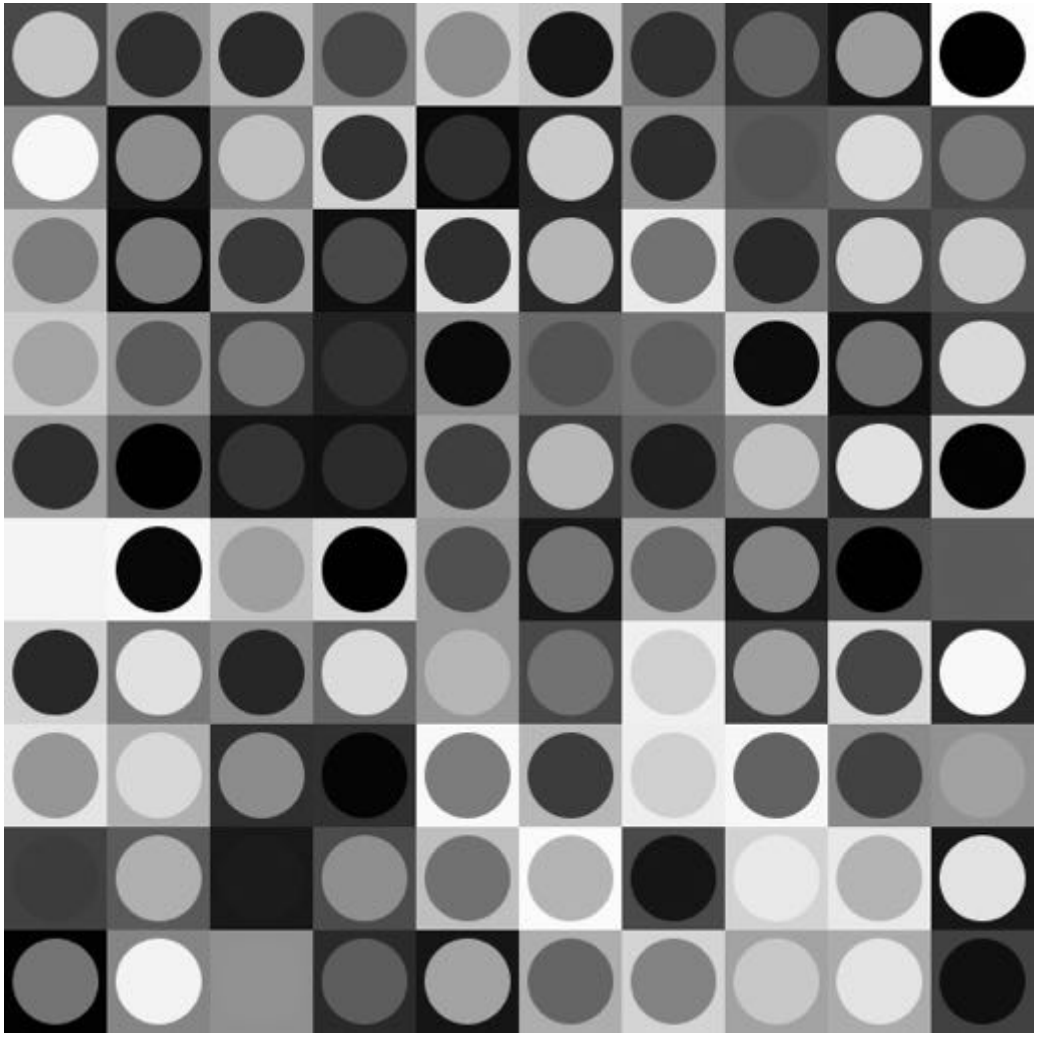
```
float r,th,x,y,l,h;

size(600,600);
background(250); noFill(); stroke(0);

for (int i=0; i<2000; i++) {
  r=random(0,300);
  th=random(0,2*PI);
  x=300+r*cos(th);
  y=300+r*sin(th);
  l=random(1,50);
  h=random(1,50);
  rect(x,y,l,h);
}

save("mondrian2.png");
```

(mondrian2.pde)

« To paint well is simply this: to put the right color in the right place. »
Paul Klee

```
for (int x = 0; x < width; x++) {
    for (int y = 0; y < height; y++) {
        stroke( 255*noise(x,y));
        point(x,y);
    }
}
```





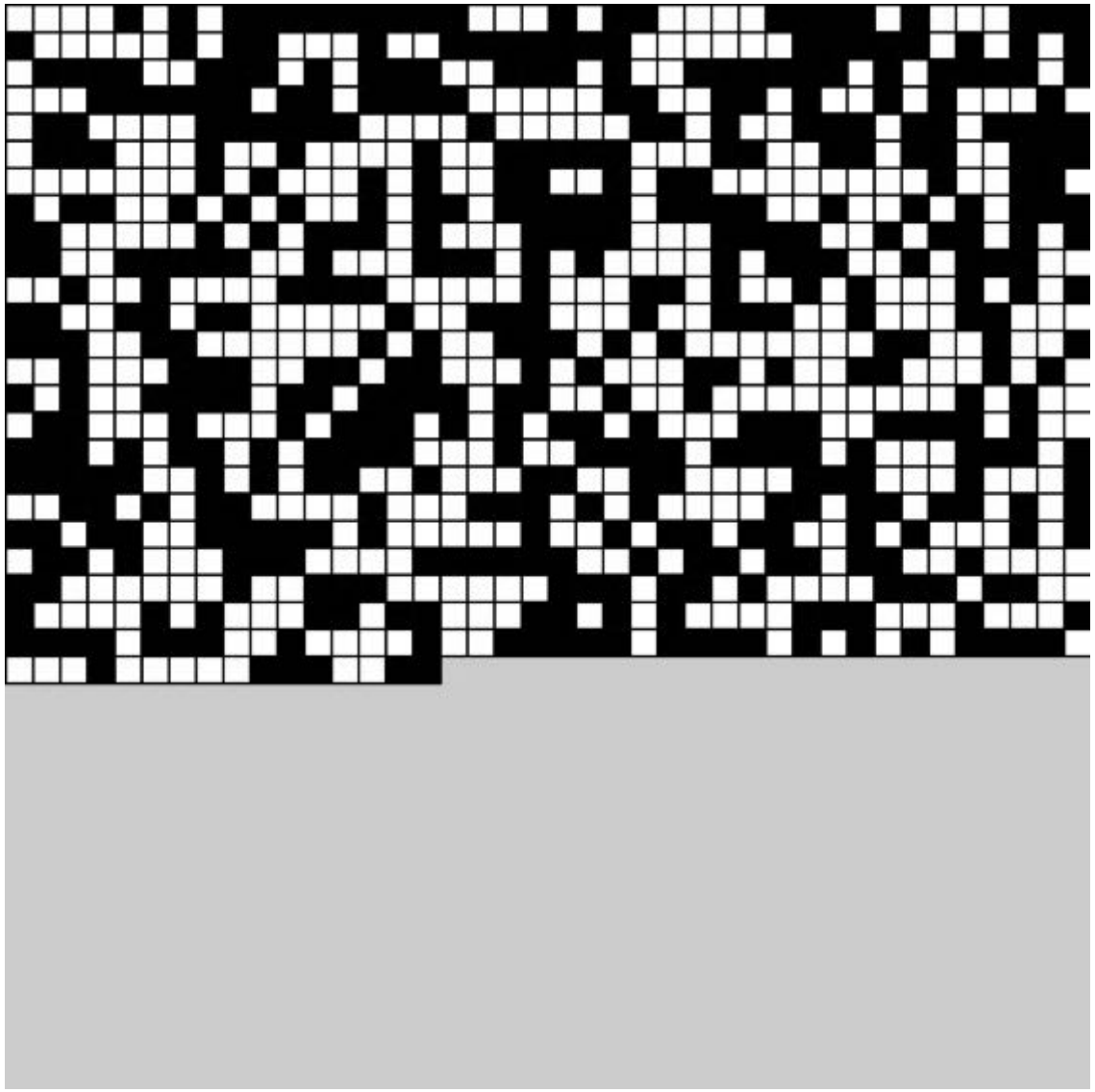
Hommage à Le Parc et Vasarely

```
size(600,600);
rectMode(CENTER);
ellipseMode(CENTER);
smooth();
noStroke();

for (int x=30; x<600; x+=60) {
  for (int y=30; y<600; y+=60) {
    fill(random(0,255));
    rect(x,y,60,60);
    fill(random(0,255));
    ellipse(x,y,50,50);
  }
}

save("leparc.png");
```

(leparc.pde)

2.6. La décision

```
if ( score < 314) {
 println(« game over »);
}
else {
 println(« shoot again »);
}
```



```
size(400,400);
int x=0;
int y=0;
for (int i=0;i<1000;i++){
  float t=random(0,1);
  if (t<0.5)
   fill(0);
  else
   fill(255);
  rect(x,y,10,10);
  if (x < width) x+=10;
  else {
    x=0;
    if (y < height) y+=10;
    else y=0;
  }
}
save("pavagernd.png");
```

(pavagernd.pde)

Le grand mystère de la suite de Syracuse

$$\forall n > 1,\ A_{n+1} = \begin{cases} 3A_n + 1 & \text{si } A_n \text{ impair} \\ A_n / 2 & \text{sinon} \end{cases}$$

Quelque soit la valeur initiale de A, la suite se termine en bouclant infiniment sur les valeurs 4, 2, 1, 4.

A ce jour, personne n'a pu démontrer pourquoi, ni prédire au bout de combien de termes la boucle est atteinte.

```
int c=1;
int A=27;
print(A);
while (A != 1) {
 if (A%2 == 0)
   A = A/2;
 else
  A = 3*A + 1;
 c++;
}
println(":"+c);
```

(troisaplus1.pde)

beginShape() … endShape()

To make a sharp turn, use the same position to specify the vertex and the following control point. To close the shape, use the same position to specify the first and last vertex.

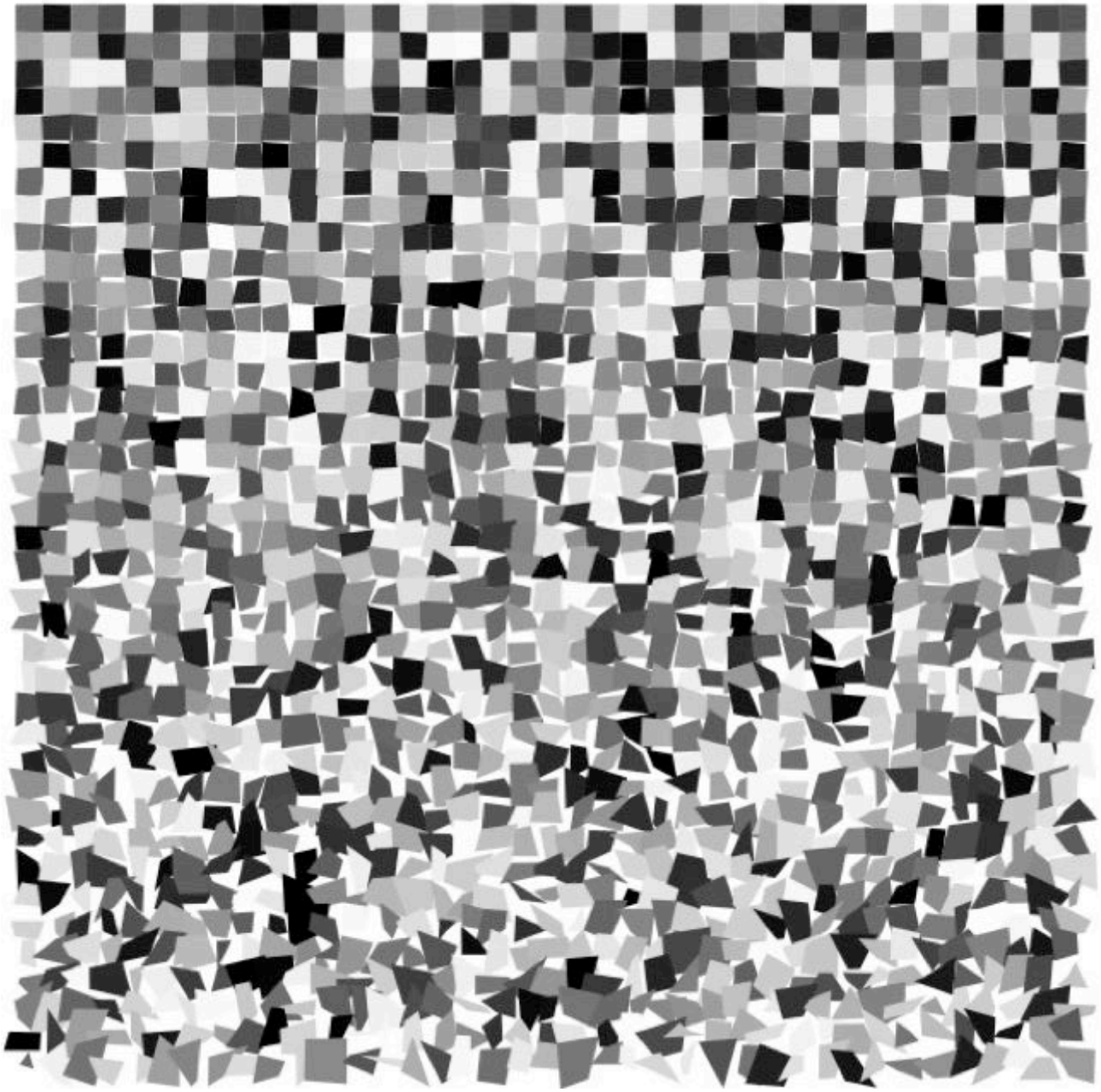
```
smooth();
noStroke();
beginShape();
vertex(90, 39);                          // V1 (see p.76)
bezierVertex(90, 39, 54, 17, 26, 83);    // C1, C2, V2
bezierVertex(26, 83, 90, 107, 90, 39);   // C3, C4, V3
endShape();
```

Place the `vertex()` function within `bezierVertex()` functions to break the sequence of curves and draw a straight line.

```
smooth();
noFill();
beginShape();
vertex(15, 40);                          // V1 (see p.76)
bezierVertex(5, 0, 80, 0, 50, 55);       // C1, C2, V2
vertex(30, 45);                          // V3
vertex(25, 75);                          // V4
bezierVertex(50, 70, 75, 90, 80, 70);    // C3, C4, V5
endShape();
```

A good technique for creating complex shapes with `beginShape()` and `endShape()` is to draw them first in a vector drawing program such as Inkscape or Illustrator. The coordinates can be read as numbers in this environment and then used in Processing. Another strategy for drawing intricate shapes is to create them in a vector-drawing program and then import the coordinates as a file. Processing includes a simple library for reading SVG files. Other libraries that support more formats and greater complexity can be found on the Processing website at *www.processing.org/reference/libraries*.

Reas & Fry p. 77



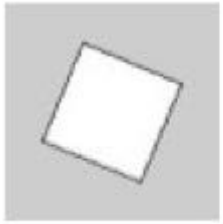
Quads - Greenberg, p. ???

(monquads.pde)

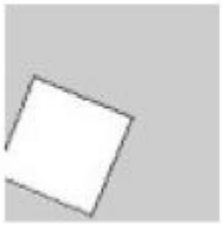
```
int DIM=15;
float VAR=0.2;

size(600,600);
background(255);
smooth();
noStroke();

int k = 0;
for (int i=0;i<height; i+=DIM){
  for (int j=0; j<width; j+=DIM){
    fill(random(0,255));
    float x0 = 0 + random(-k*VAR, k*VAR);
    float y0 = 0 + random(-k*VAR, k*VAR);
    float x1 = DIM + random(-k*VAR, k*VAR);
    float y1 = 0 + random(-k*VAR, k*VAR);
    float x2 = DIM + random(-k*VAR, k*VAR);
    float y2 = DIM + random(-k*VAR, k*VAR);
    float x3 = 0 + random(-k*VAR, k*VAR);
    float y3 = DIM + random(-k*VAR, k*VAR);
    quad(x0,y0,x1,y1,x2,y2,x3,y3);
    translate(DIM, 0);
  }
  translate(-width,DIM);
  k ++;
}
save("quads.tif");
```

Le repère de tracé (la caméra)

pushMatrix() popMatrix() translate(…) scale(…) rotate(…)

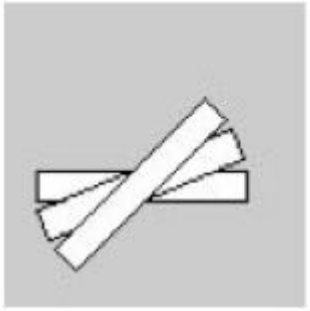


```
translate(width/2, height/2);
rotate(PI/8);
rect(-25, -25, 50, 50);
```



```
rotate(PI/8);
translate(width/2, height/2);
rect(-25, -25, 50, 50);
```

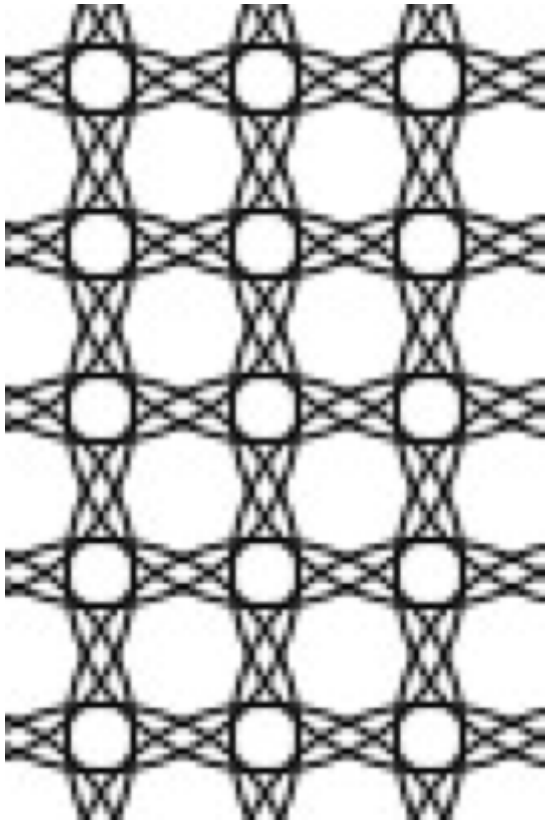
Casey & Reas
p.139 et 141



```
translate(45, 60);
rect(-35, -5, 70, 10);
rotate(-PI/8);
rect(-35, -5, 70, 10);
rotate(-PI/8);
rect(-35, -5, 70, 10);
```

ex: création de trame par déplacement du repère

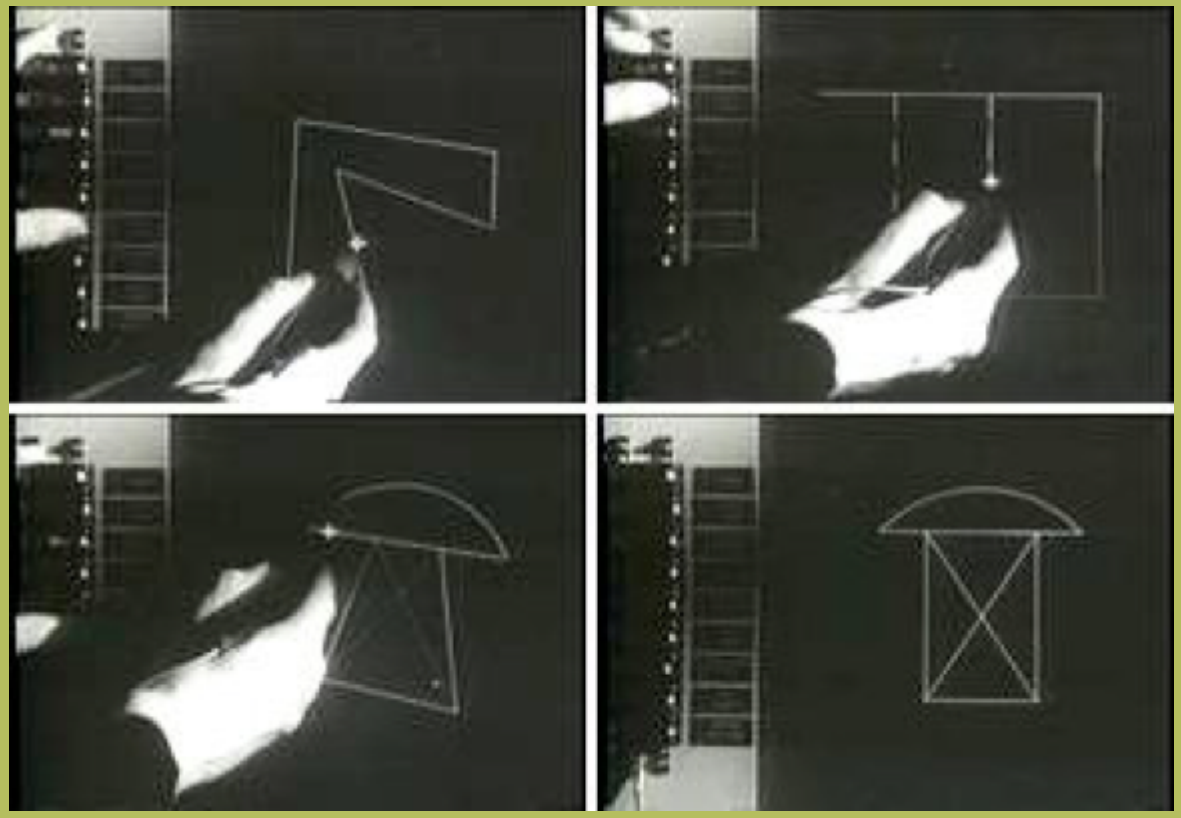

```
for (int i=0; i<24; i++){
   pushMatrix();
   for (int j=0; j<12; j++){
      ellipse(0,0,60,60);
      translate(25,0);
   }
   popMatrix();
   translate(0,25);
 }
```



les fonctions

les splines

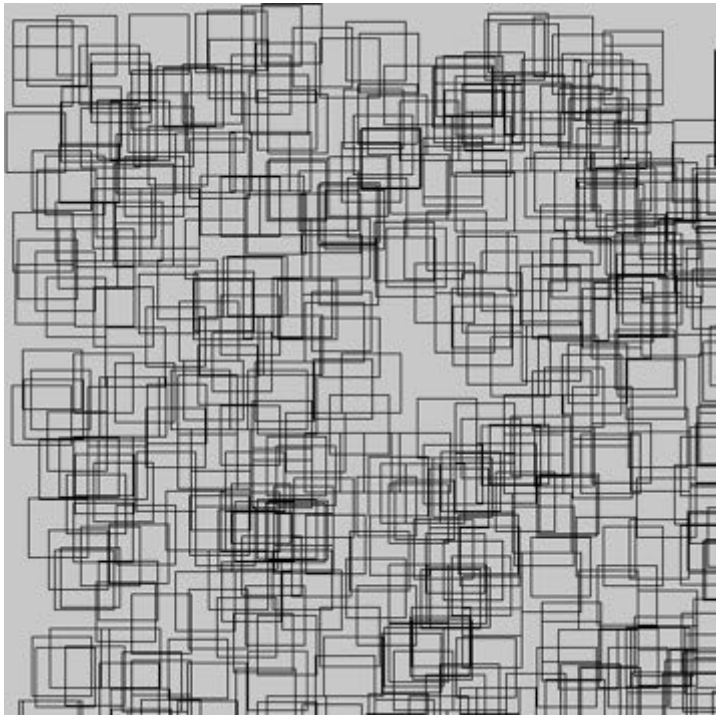
PARTIE 5. L'interaction



Y. Sutherland
sketchpad

Processing s'anime

setup(){…} draw(){…}
frameRate(…) frameCount



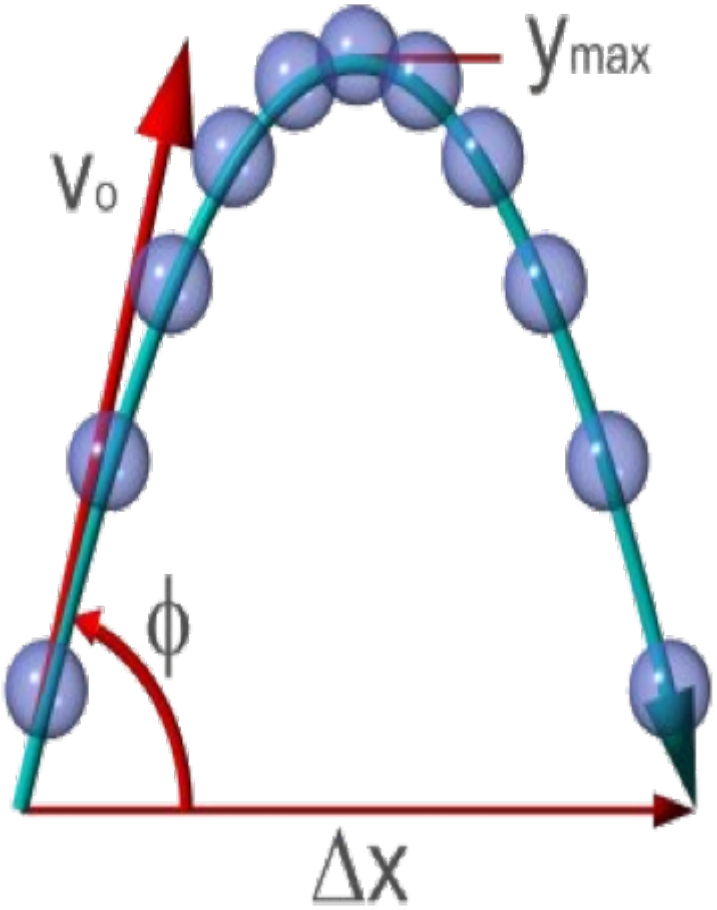
```
void setup() {
  size(600,600);
  noFill();stroke(0);
  background(200);
}

void draw() {
  float x=random(0,width);
  float y=random(0,height);
  rect(x,y,50,50);
}
```

(modereactif1.pde)

vu à l’expo Artistes & robots – crédits ?

Un peu de physique

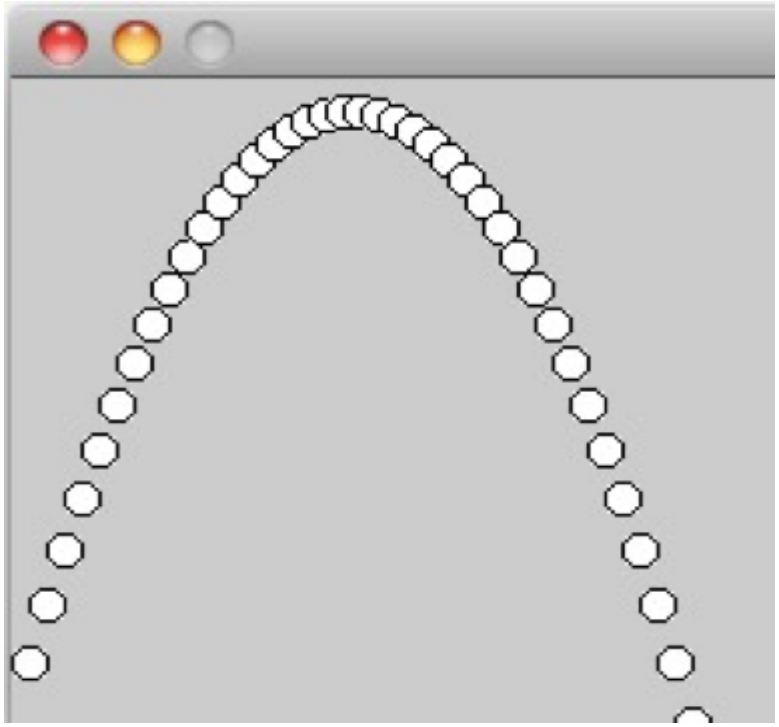
$$\sum_i \vec{F}_i = m\vec{a}$$



$$\sum \vec{F} = \begin{pmatrix} 0 \\ -mg \\ 0 \end{pmatrix} = m\vec{a} = \begin{pmatrix} m\ddot{x} \\ m\ddot{y} \\ m\ddot{z} \end{pmatrix} \Rightarrow \begin{cases} \dot{x} = v_0 \cos\Phi \\ \dot{y} = v_0 \sin\Phi - gt \\ \dot{z} = 0 \end{cases} \Rightarrow \begin{cases} x = v_0 t \cos\Phi \\ y = v_0 t \sin\Phi - \frac{1}{2} g t^2 \\ z = 0 \end{cases}$$

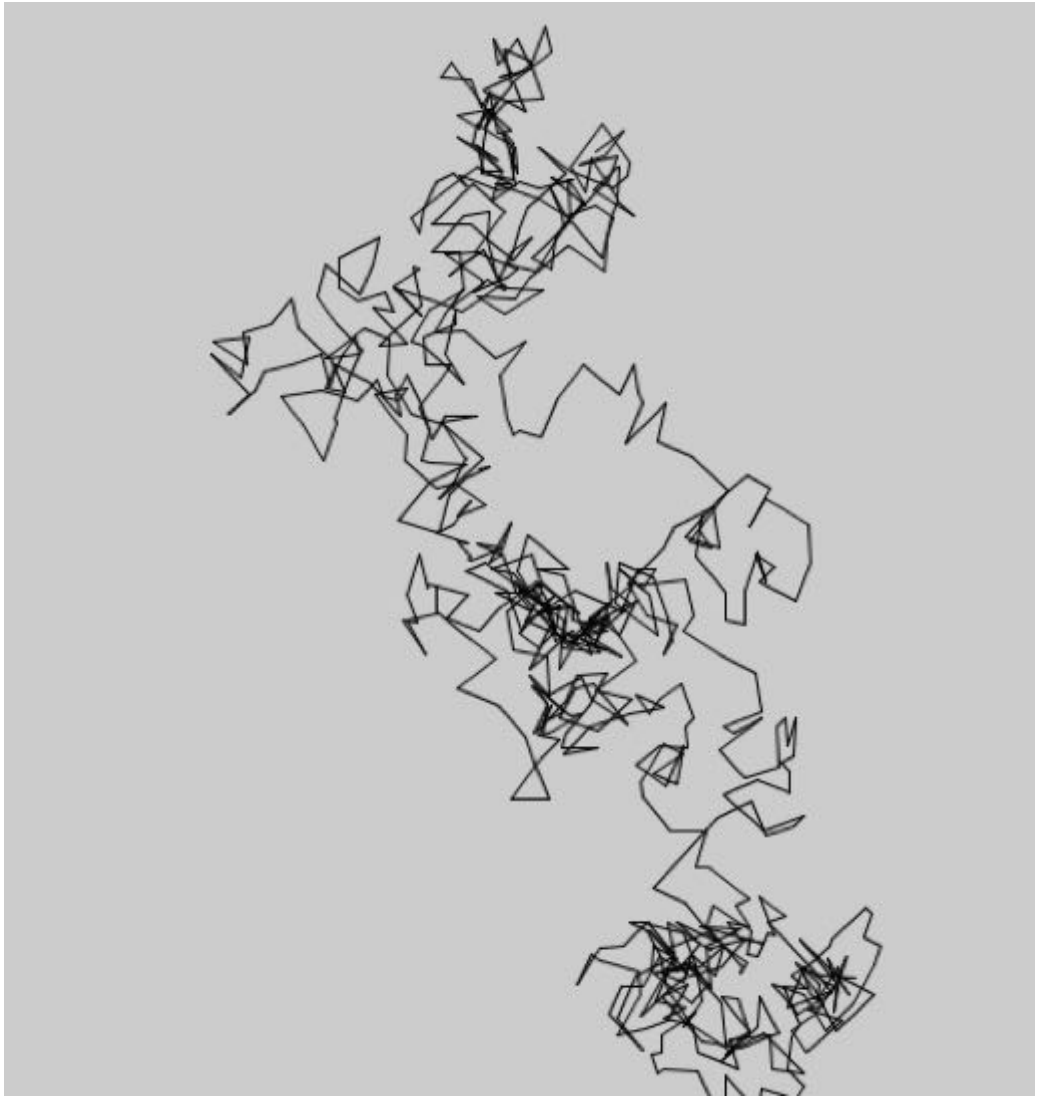
boulet

```
float x,y,vx,vy,ax,ay;

void setup(){
  size(600,200);
  x = 0;
  y = 0;
  vx = 5;
  vy = 20;
}

void draw()  {
  ax = 0;
  ay = -1                    = 1 m/s2
  vx = vx + ax;
  vy = vy + ay;
  x = x + vx;
  y = y + vy;
  ellipse(x,height-y,10,10);
}
```



ajouter les rebonds

Mouvement brownien



```
float ox=300;
float oy=300;
float x,y;

void setup() {
  size(600,600);
  stroke(0);
  smooth();
}

void draw() {
  x = ox + random(-20,+20);
  y = oy + random(-20,+20);
  line(ox,oy,x,y);
  ox = x;
  oy = y;
}
```

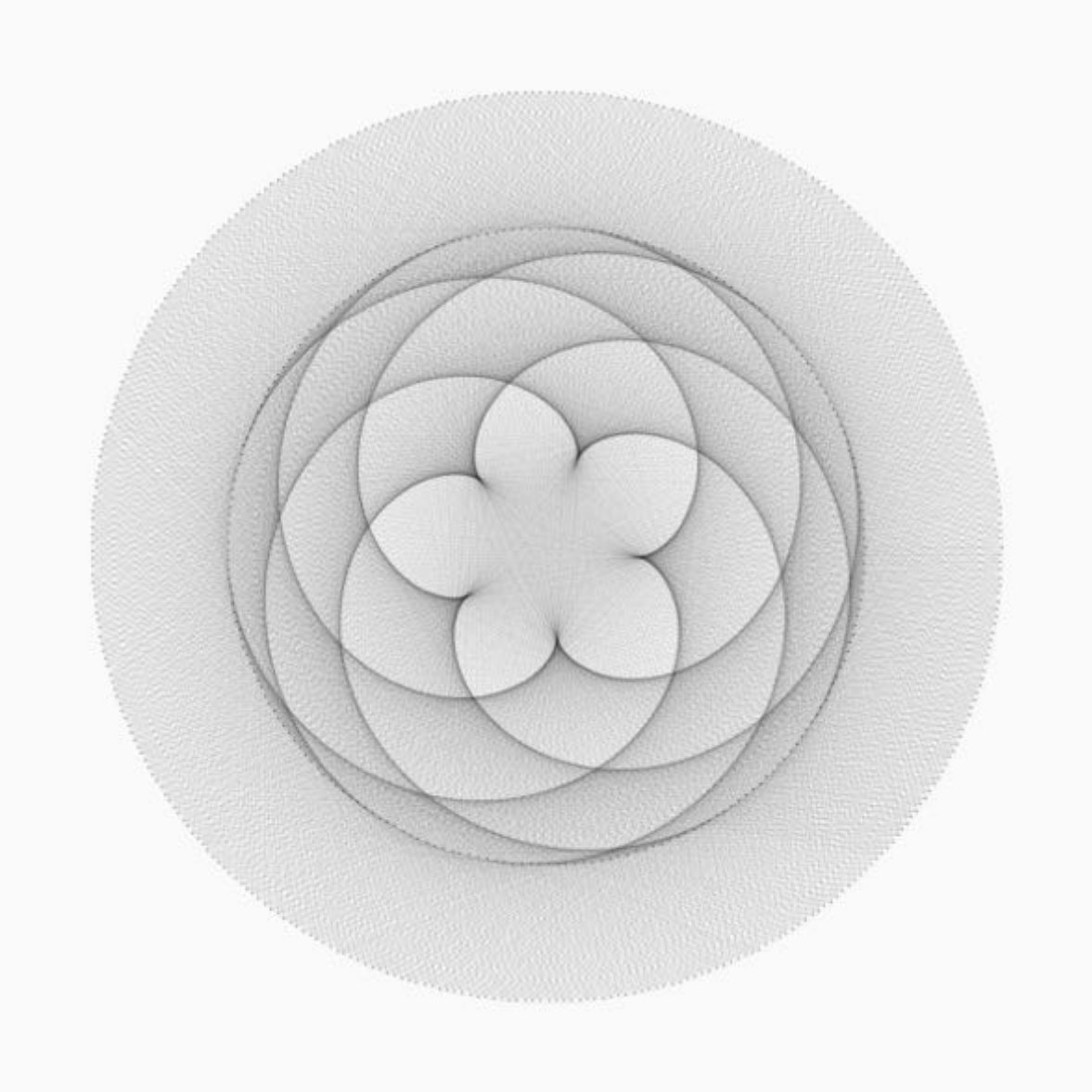
(brownien1.pde)

Variante : particule brownienne

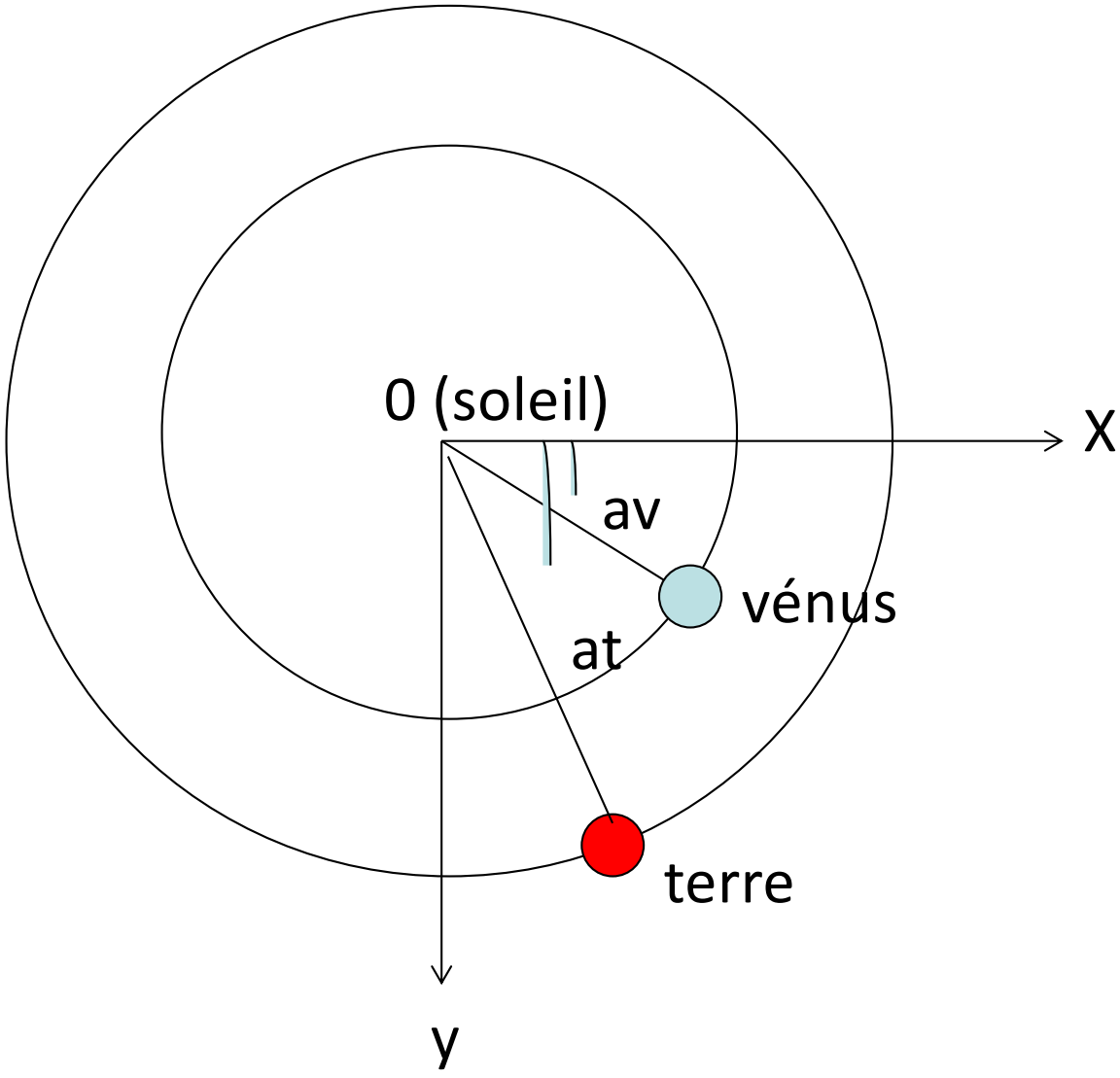
animation2

```
float x=300;
float y=300;

void setup() {
  size(600,600);
  fill(100);noStroke();
  smooth();
}

void draw() {
  background(200);
  x = x + random(-2,+2);
  y = y + random(-2,+2);
  ellipse(x,y,50,50);
}
```

Orbites planétaires



Un peu de trigonométrie



d(soleil, venus) = 0.7 UA
=>175 pixels
d(soleil, terre) = 1 UA
=> 250 pixels

revolution terre = 365j
revolution vénus = 225j

xv = 175*cos(av)
yv = 175*sin(av)

av += 2π/225

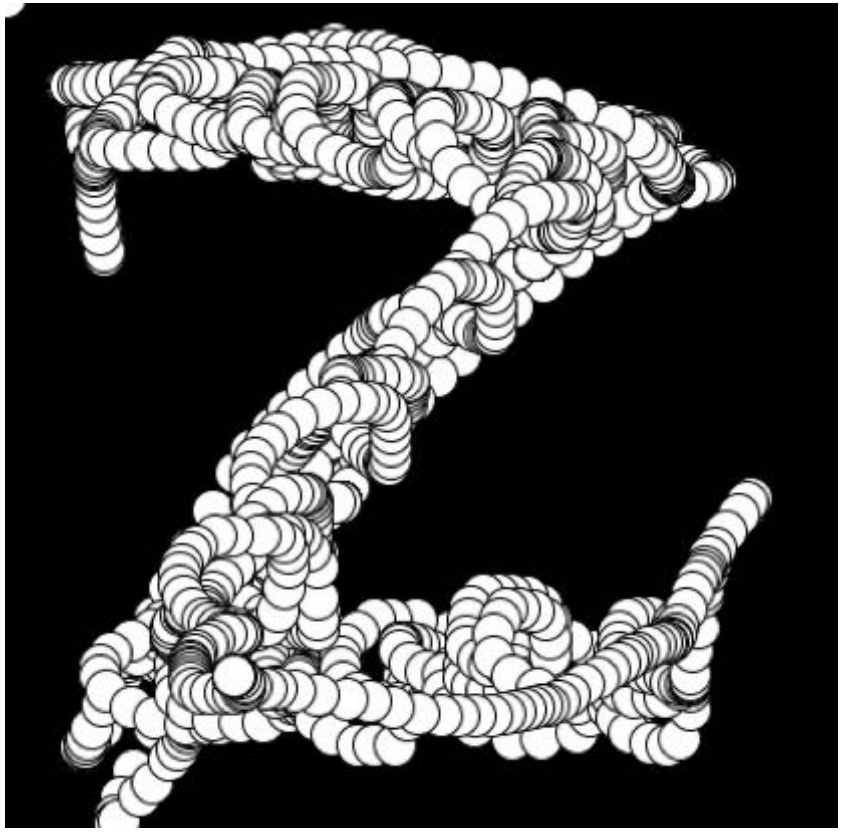
```
float at,av,dt,dv;

void setup(){
  size(600,600);
  smooth();
  background(250);
  //stroke(150,0,250,10);
  stroke(0,10);
  ellipseMode(CENTER);
  noFill();
  ellipse(300,300,500,500);
  ellipse(300,300,350,350);
  dt = 2*PI/365/1;
  dv = 2*PI/225/1;
  at = av = 0;
}
```

```
void draw(){
  // position terre
  float xt = 250*cos(at);
  float yt = 250*sin(at);
  //ellipse(300+xt,300+yt,10,10);
  // position venus
  float xv = 175*cos(av);
  float yv = 175*sin(av);
  //ellipse(300+xv,300+yv,10,10);
  // arc
  line(300+xt,300+yt,300+xv,300+yv);
  // avance des planetes
  at = (at+dt)%(2*PI);
  av = (av+dv)%(2*PI);
}

void keyPressed(){
  saveFrame("rosace###.png");
}
```

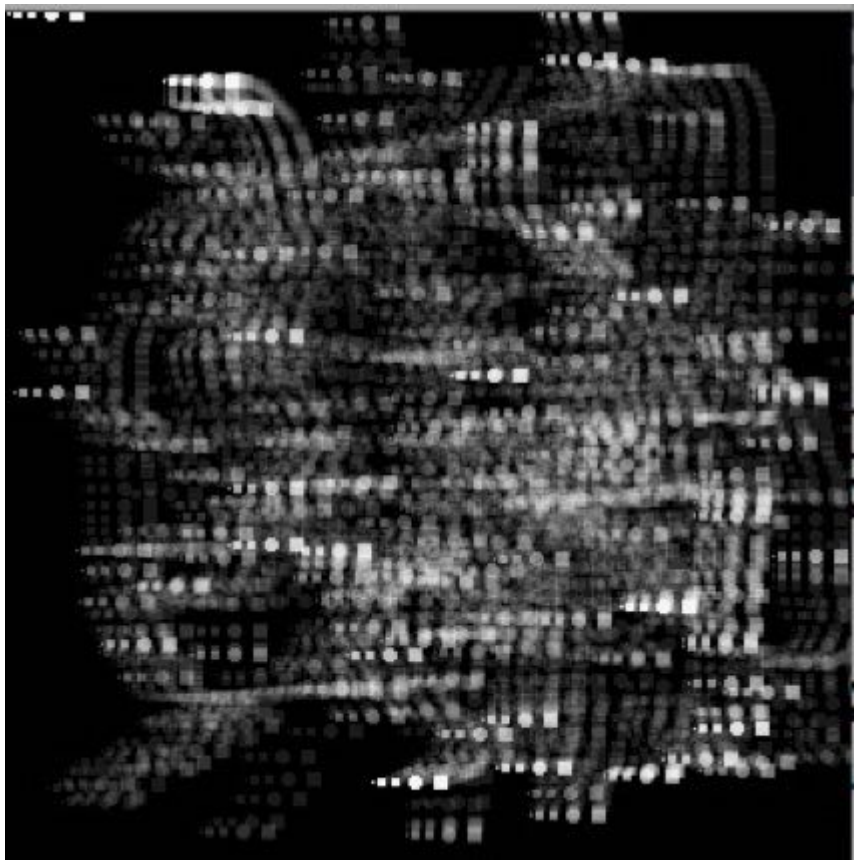
(planetes.pde)

Souris, curseur

```
void setup(){
  size(400,400);
  smooth();
  background(0);
  noCursor();
}

void draw(){
//  background(0);
  int x = mouseX;
  int y = mouseY;
  ellipse(x,y,20,20);
}
```



(demosouris.pde)



Symétries

```
boolean dessin = false;

void setup() {
  size(500, 500);
  background(226);
  smooth();
  strokeWeight(20); stroke(0, 100);
}

void draw() {
  point(mouseX,mouseY);
  point(width-mouseX, mouseY);

  //// variante pour symetrie centrale
  //point(mouseX,height-mouseY);
  //point(width-mouseX,height-mouseY);

}
```

(symetries.pde)

Gestion d'évènements

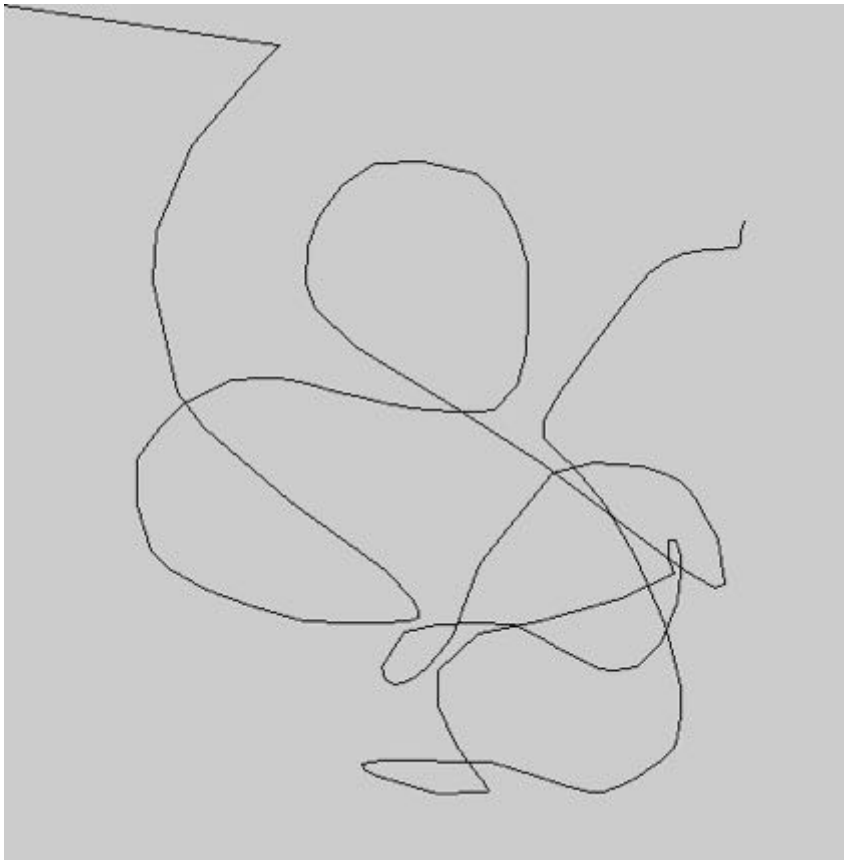
Pour que le code réagisse aux actions de l'utilisateur, il faut renseigner les fonctions appropriées :

void mousePressed()
void mouseReleased()
void mouseMoved()
void mouseDragged()
+ utiliser les variables pré-définies mouseX mouseY

void keyPressed()
void keyReleased()
+ utiliser la variable pré-définie key

linepmouse

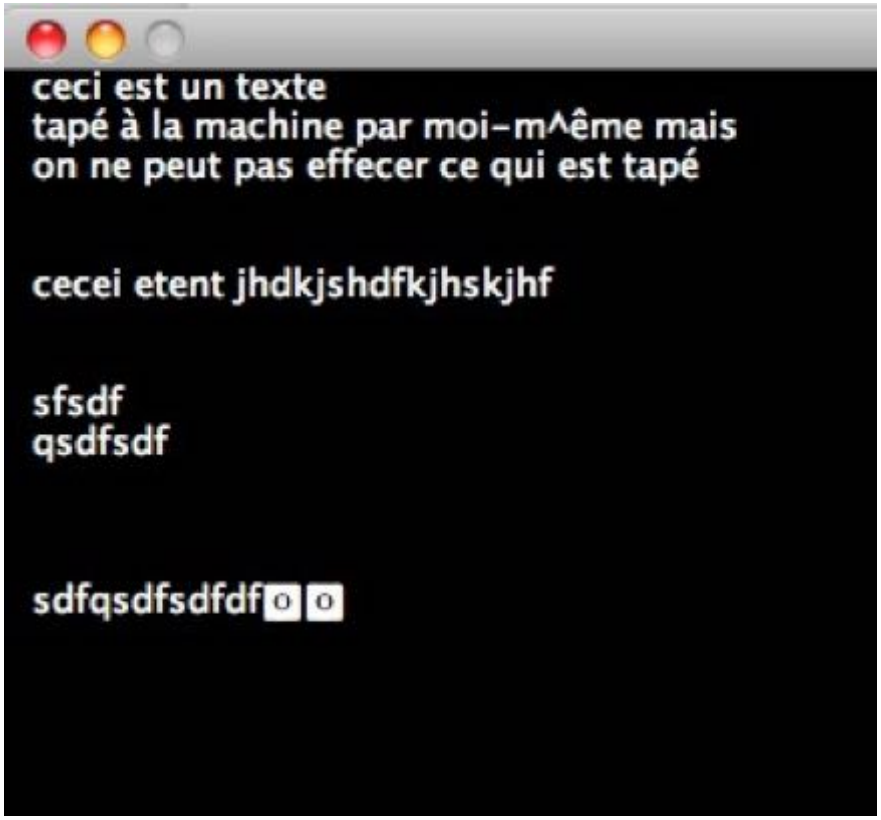
```
void setup(){
  size(600,600);
}
void draw(){
  line(pmouseX,pmouseY,mouseX,mouseY);
}
```



une (mauvaise) machine à écrire

machineecrire

```
String letexte="";

void setup(){
  size(800,600);
  stroke(255);fill(255);
}

void draw(){
  background(0);
  text(letexte,10,10);
}

void keyPressed() {
 letexte += key;
}
```



pas de retour (feedback) utilisateur !

slider sur particule

Les images avec Processing

```
Pimage img = loadImage("gnagna.jpg");
image(img, 30,40, 640, 480);
```

Coloriage de l'image : tint(…) noTint()

Accès aux pixels de l'écran : get(…) set(…) copy(…)
Pour une image : ima.get(…) ima.set(…)

filter(…) blend(…) mask(…)

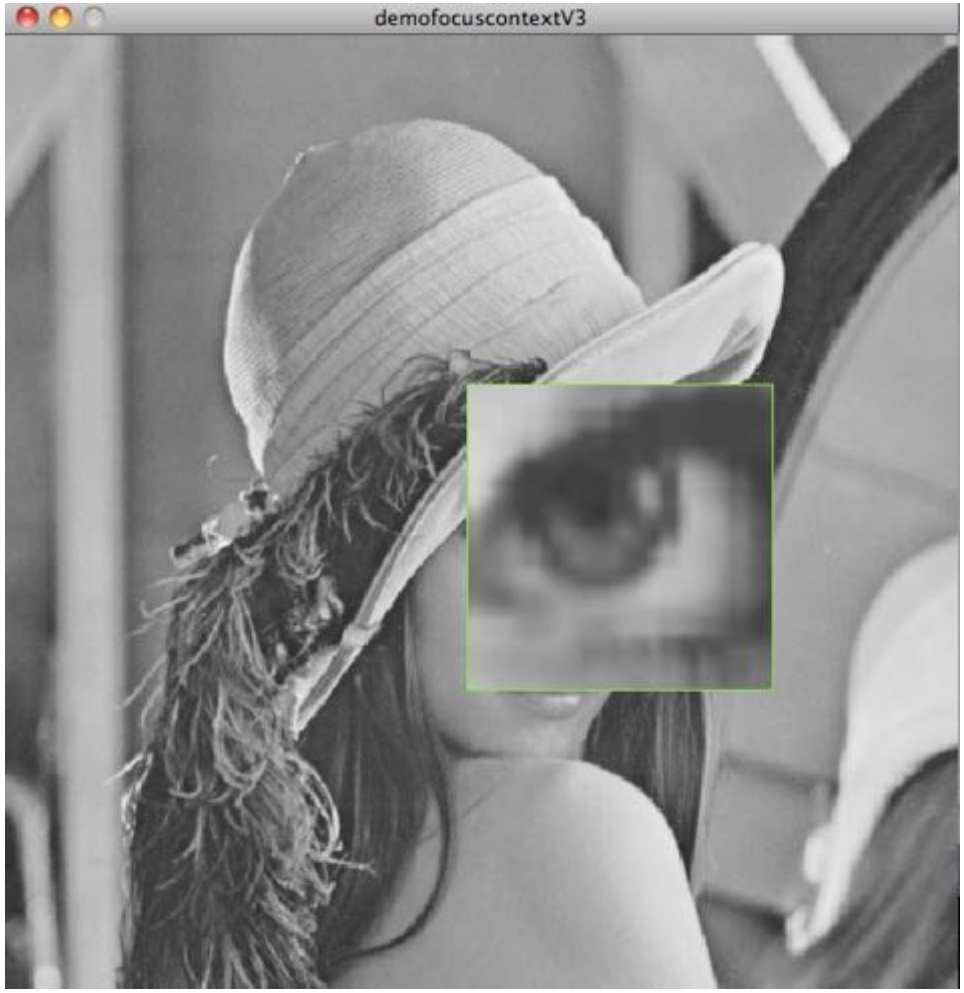
loadPixels(…) Pixel[] updatePixels()

Hommage à Akakliké

```
PImage ima;

void setup(){
  size(600,600);
  ima=loadImage("Photo.jpg");
}

void draw(){
  copy(ima,mouseX,mouseY,30,30,mouseX,mouseY,30,30);
}
```

(akaklike2016.pde)

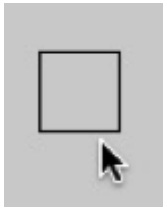
Loupe



```
PImage pgmat;
float ar;

void setup(){
 pgmat = loadImage("lenna.gif");
 ar = pgmat.width/float(pgmat.height);
 size(int(600*ar), 600);
}

void draw(){
  int x = mouseX;
  int y = mouseY;
  image(pgmat, 0,0, width,height);
  copy(x-16,y-16,32,32, x-96, y-96, 192,192);
  stroke(0,255,0);noFill();rect(x-96, y-96, 192,192);
}
```

(demofocuscontextV3.pde)

Dessine-moi un bouton (radio)



état 1 : non sélectionné, non désigné

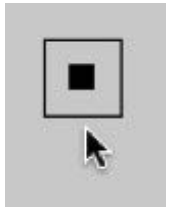
entrée de zone : "roll over"

état 2 : non sélectionné, désigné

clic

état 3 : sélectionné, désigné

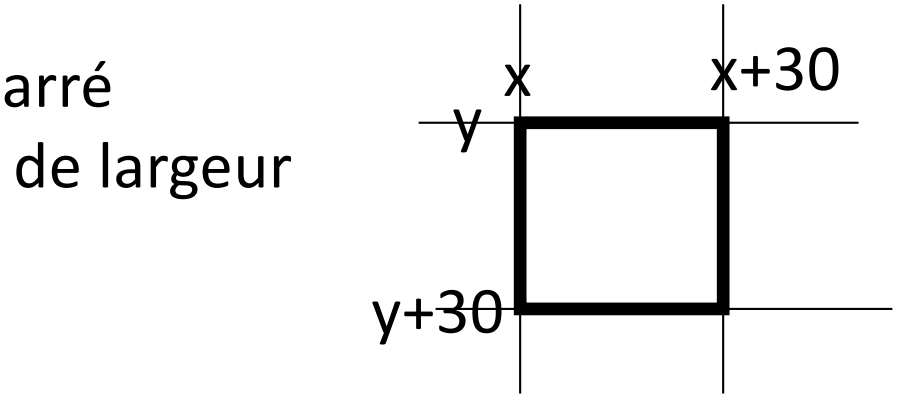
sortie de zone



état 4 : non sélectionné, désigné

Détection du rollover :

bouton = un carré
en coord (x,y) de largeur
30 pixels



si (mouseX > x) et (mouseX < x+30)
et (mouseY >y) et (mouseY < y+30)
alors le curseur est dans la boite du bouton
sinon il est dehors

⇒une variable booléenne ("boolean")
pour le rollover + une autre pour la selection

début du code

monboutonV0

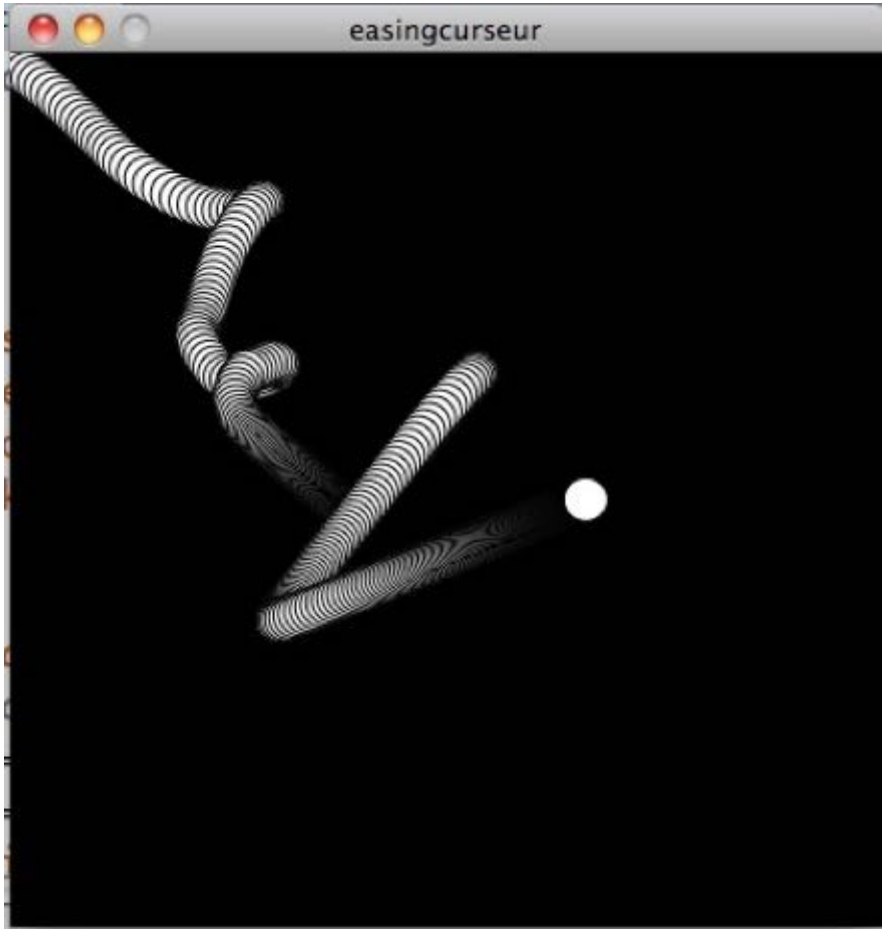
```
int x,y;
boolean rollover, selected;

void setup() {
  size(200,200);
  x = 50; y = 50;
  rollover = false; selected = false;
}

void draw() {
  background(200);
  stroke(0);noFill();
  if (rollover) strokeWeight(4); else strokeWeight(1);
  rect(x,y,30,30);
  if (selected) {
    noStroke();fill(0);
    rect(x+10,y+10,10,10);
  }
}
```

Suite du code :

```
void mouseMoved() {
 int mx = mouseX;
 int my = mouseY;
 if (mx > x && mx < x + 30 && my > y && my < y + 30)
   rollover = true;
 else
   rollover = false;
}

void mousePressed() {
 if (rollover)
   selected = ! selected;
}
```

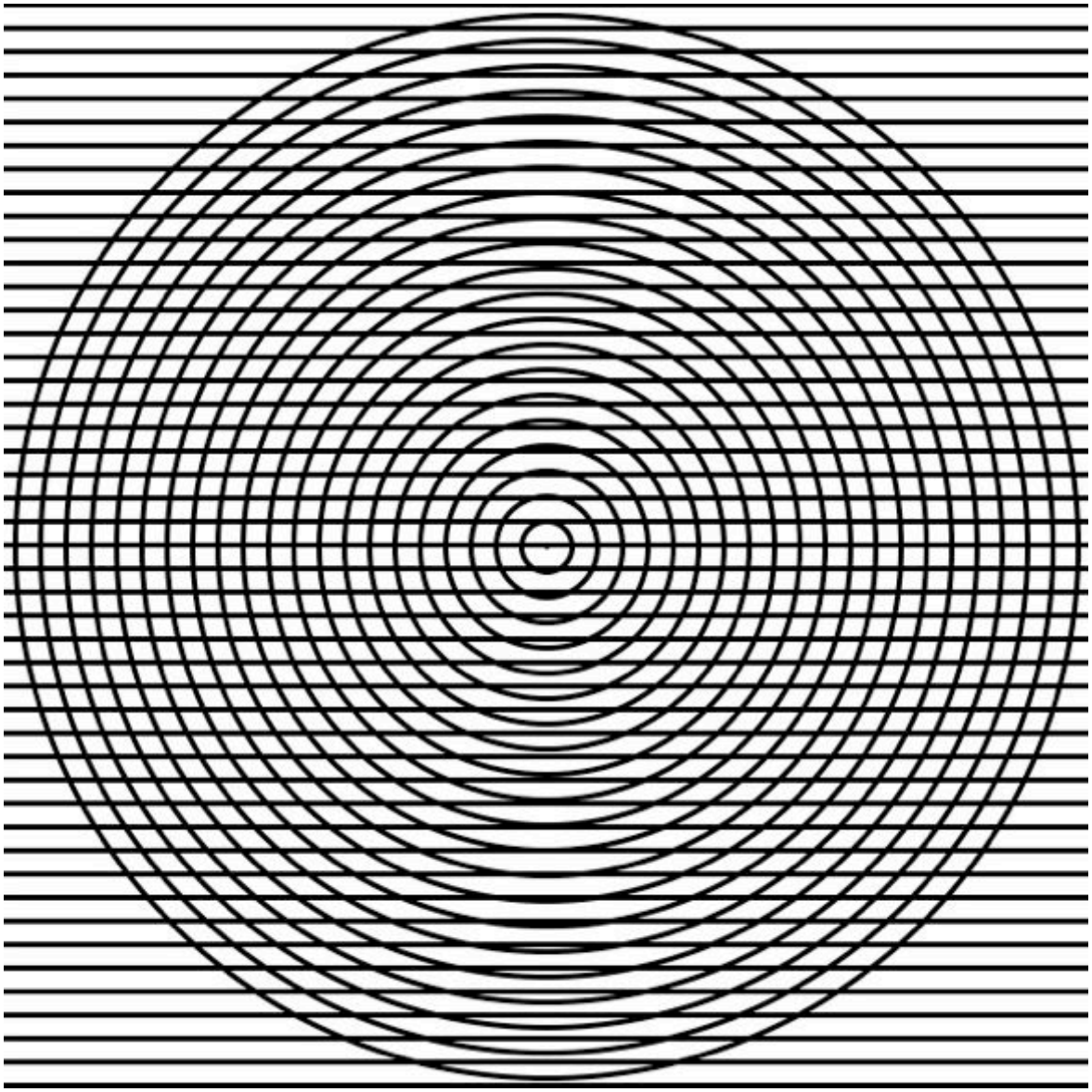
essayer d'autres
formes de boutons
et d'autres feedbacks

Easing



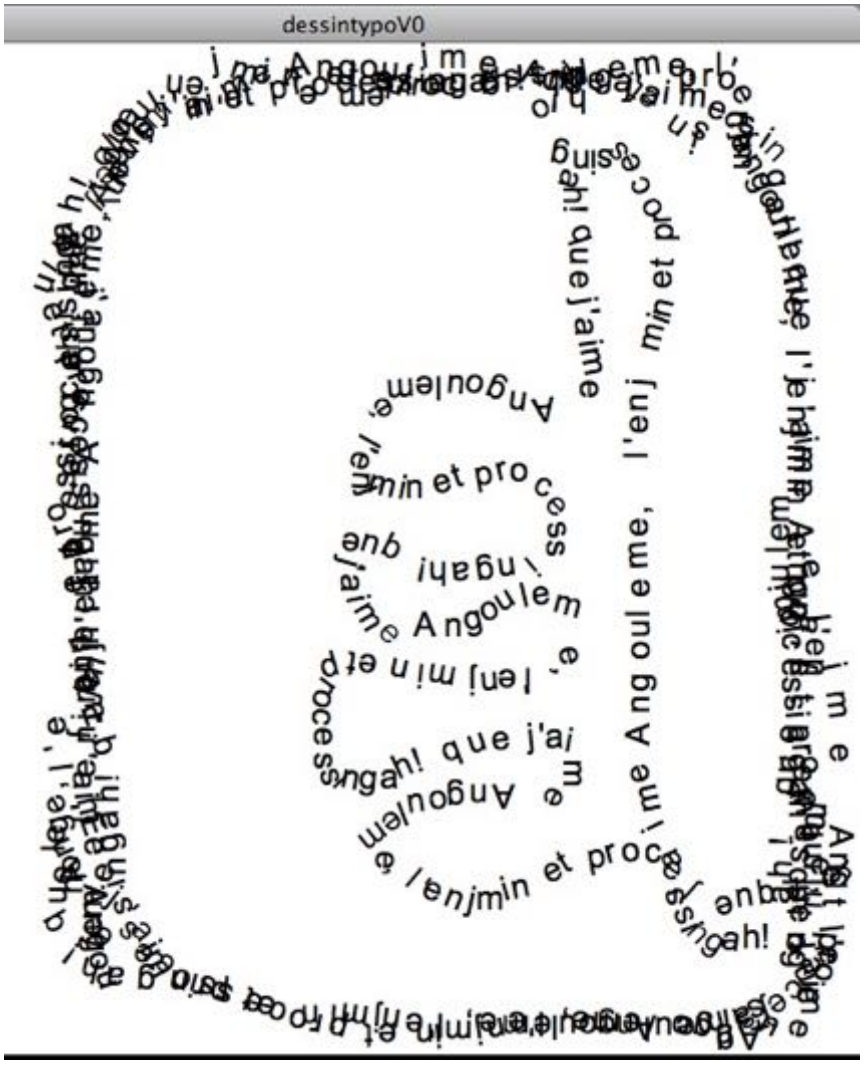
```
// d'apres Geridan & Lafargue "Processing »
float x=0;
float y=0;
float ease=0.1;

void setup(){
  size(400,400);
  smooth();
  background(0);
}

void draw(){
//  background(0);
  x = x*(1-ease) + mouseX*ease;
  y = y*(1-ease) + mouseY*ease;
  ellipse(x,y,20,20);
}
```

Le moiré interactif, avec la souris



```
void draw(){
  background(255);
  //dessin grille
  int y = 0;
  while (y<height){
    line(0,y,width,y);
    y += delta;
  }
  //dessin cibles
  float delta2=map(mouseY,0,height,delta,2*delta);
  pushMatrix();
  translate(mouseX, mouseY);
  int r = 0;
  while (r<height/2){
   ellipse(0,0,2*r,2*r);
   r += delta2+1; //ou delta ou etc.
  }
  popMatrix();
}
```

(moiresouris.pde)



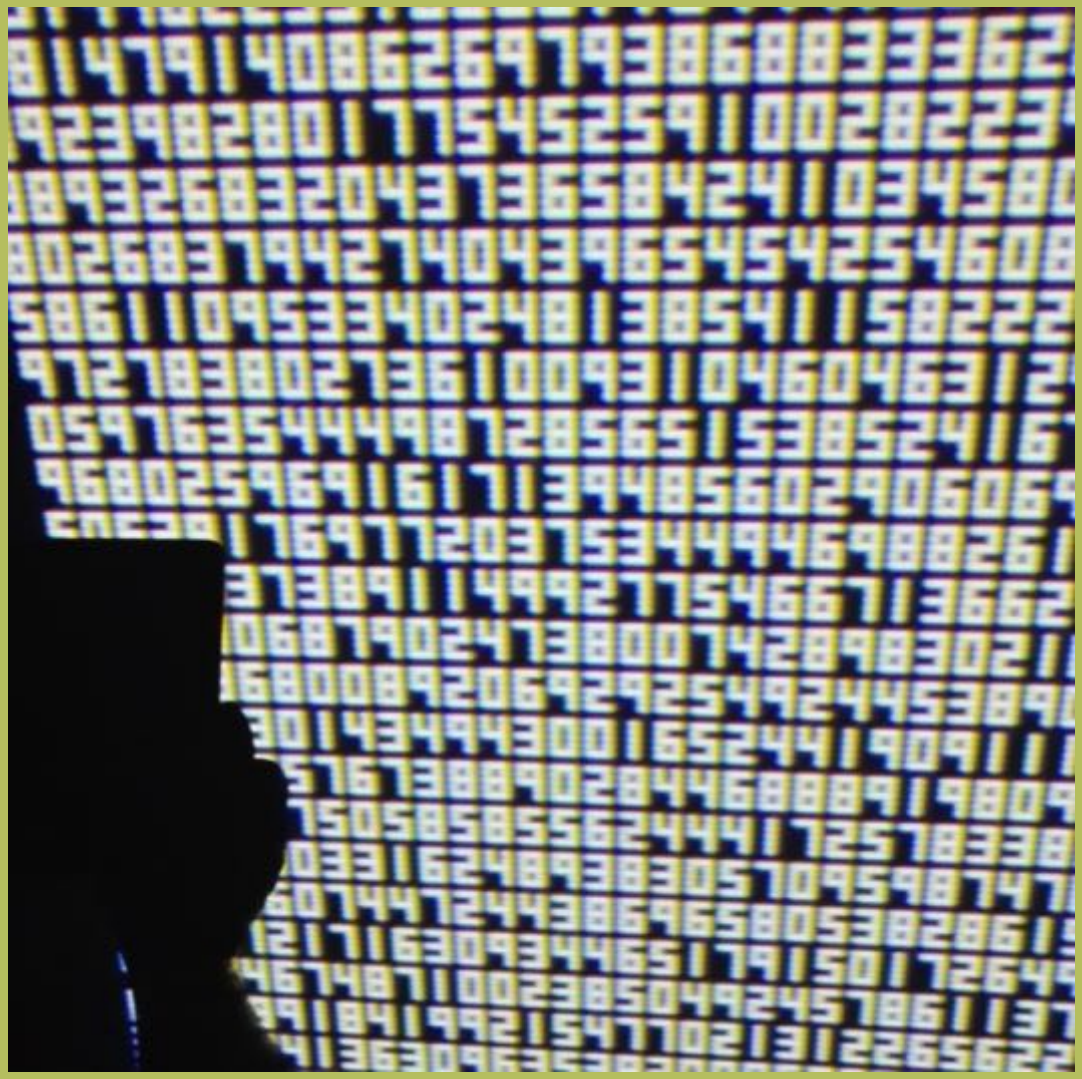
d'après programme P.2.3.3 de "Design génératif"

```
float x, y, ox, oy, d;
float stepSize;
PFont font;
String letters = "ah! que j'aime Angouleme, l'enjmin et processing";
int counter = 0;

void setup() {
  size(600,600);
  background(255);
  fill(0);
  smooth();
  cursor(CROSS);
  font = createFont("Arial",10);
  textFont(font,20);
  textAlign(LEFT);
  stepSize = 0;
  x = mouseX;
  y = mouseY;
  ox = x;
  oy = y;
}
```

```
void draw() {
  x = mouseX;
  y = mouseY;
  d = dist(x,y, ox,oy);

  if (d >= stepSize) {
   float angle = atan2(y-oy, x-ox);
   pushMatrix();
   translate(x, y);
   rotate(angle);
   //textFont(font,fontSizeMin+d);
   char newLetter = letters.charAt(counter);
   text(newLetter, 0, 0);
   counter = (counter+1)%letters.length();
   popMatrix();

   stepSize = textWidth(newLetter);
   ox = x;
   oy = y;
  }
}
```

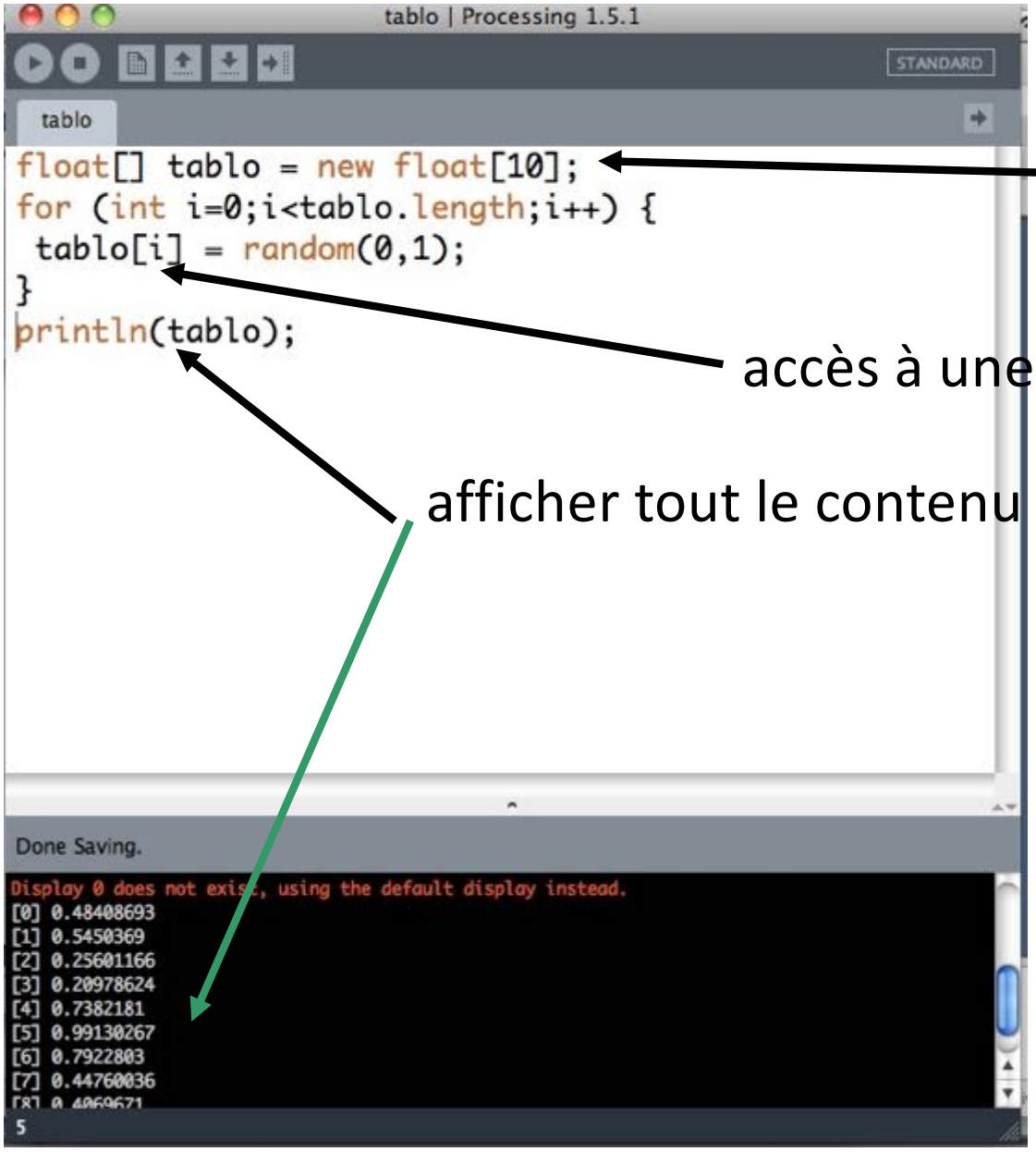
(dessintypoV0.pde)

PARTIE 6. Data/objets



Rioji Ikeda – Data.tron

4.1 Les tableaux



déclaration

accès à une case

afficher tout le contenu

declaration ≠ allocation

```
float[] tablo;
tablo = new float[10];
```

la taille peut dépendre d'une expression entière

```
tablo = new float[x+n%25];
```

on commence à la case 0 jusqu'à length-1

```
x = tablo[0] - tablo[i] + tablo[tablo.length-1];
```

Exemple graphique

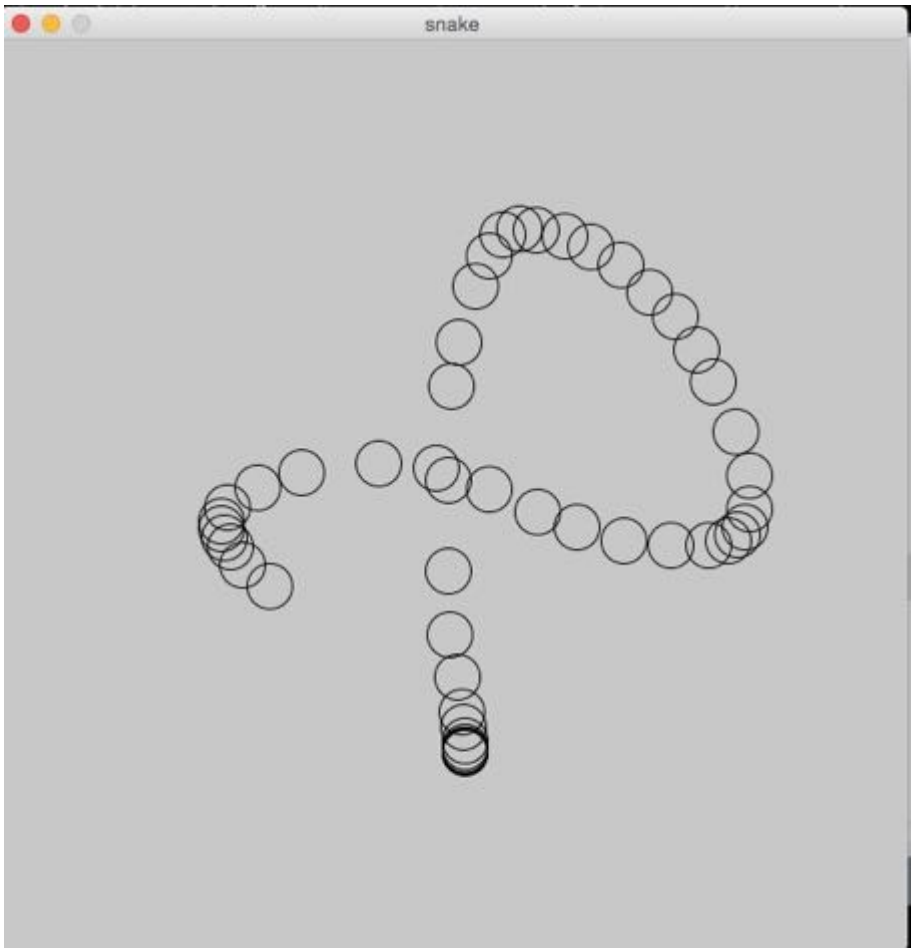
```
size(100,600);

float[] tablo = new float[100];
for (int i=0;i<tablo.length;i++) {
 tablo[i] = random(0,1);
}

fill(0);
int y=0;
for (int i=0;i<tablo.length;i++) {
 rect(0,y,100*tablo[i],3);
 y += 6;
}

save("tablo.png");
```

(tablo.pde)

Des tableaux aux listes : arraylist

```
ArrayList<maclasse> maliste;   //declaration
maliste= new ArrayList<maclasse>();  //creation

int N = maliste.size();

maclasse ob = maliste.get(i); // acces

maclasse ob = new maclasse(); // insertion
maliste.add(ob);

maliste.remove(i);   // destruction
```

snake : historique des positions



```
ArrayList<PVector> hist = new ArrayList<PVector>();

void setup() {
  size(600,600);
  noFill();
  smooth();
}
```

(snake.pde)

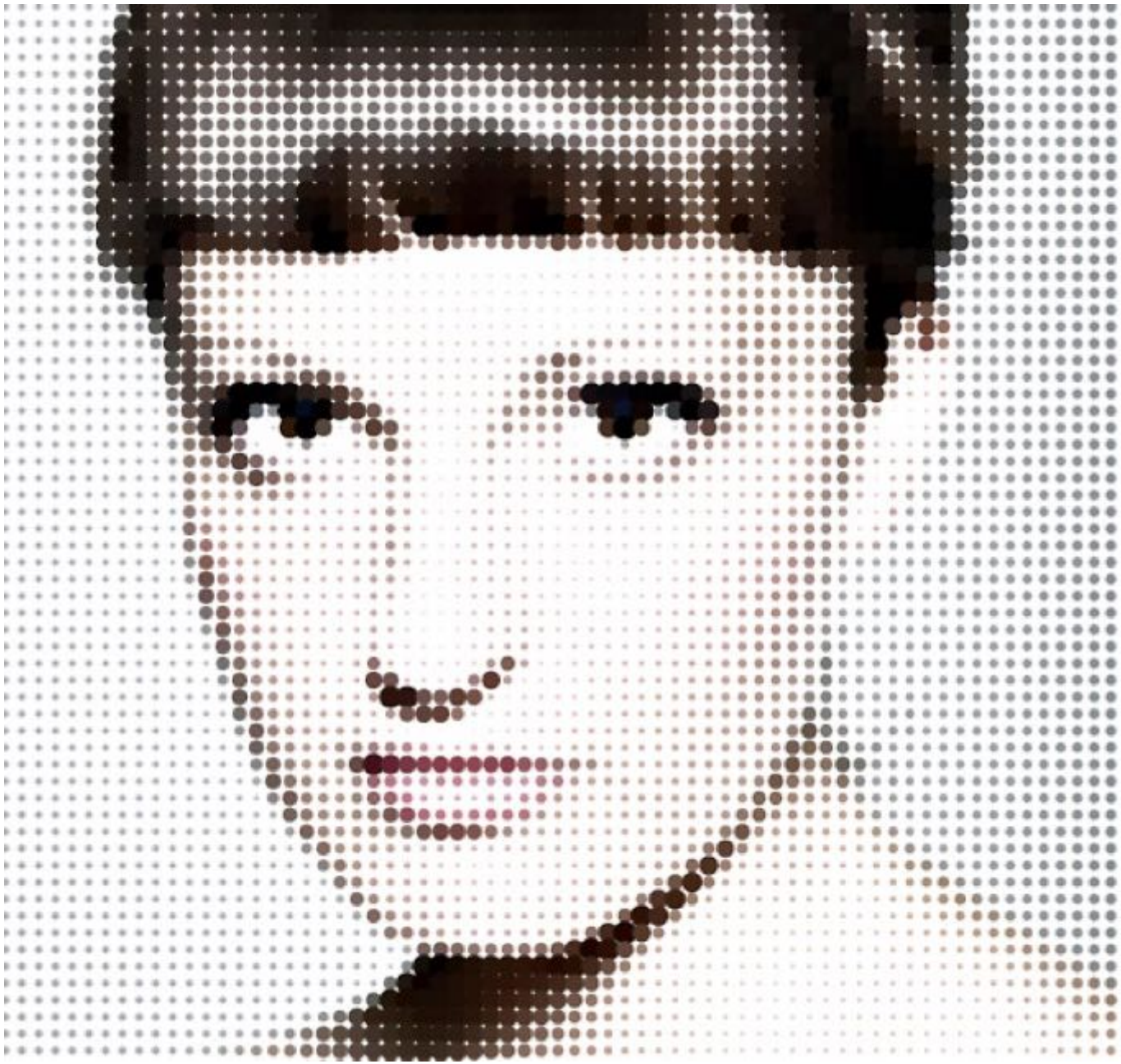
```
void draw() {
  background(200);
  for (int i=0;i<hist.size();i++) {
   ellipse(hist.get(i).x, hist.get(i).y, 30, 30);
  }
}

void mouseMoved() {
    PVector newpos= new PVector(mouseX,mouseY);
    hist.add(newpos);
    if (hist.size()>50) hist.remove(0);
}
```

Manipuler les pixels d'une image : un tableau

image

```
// chargement du fichier
PImage ima = loadImage("lenna.gif");
// traitement sur l'image
ima.loadPixels();
for (int i=0;i<ima.width*ima.height;i++){
  ima.pixels[i] = 255 - ima.pixels[i];
}
ima.updatePixels();
// affichage du resultat
size(ima.width,ima.height);
image(ima,0,0);
// sauvegarde
ima.save("resultat.gif");
```

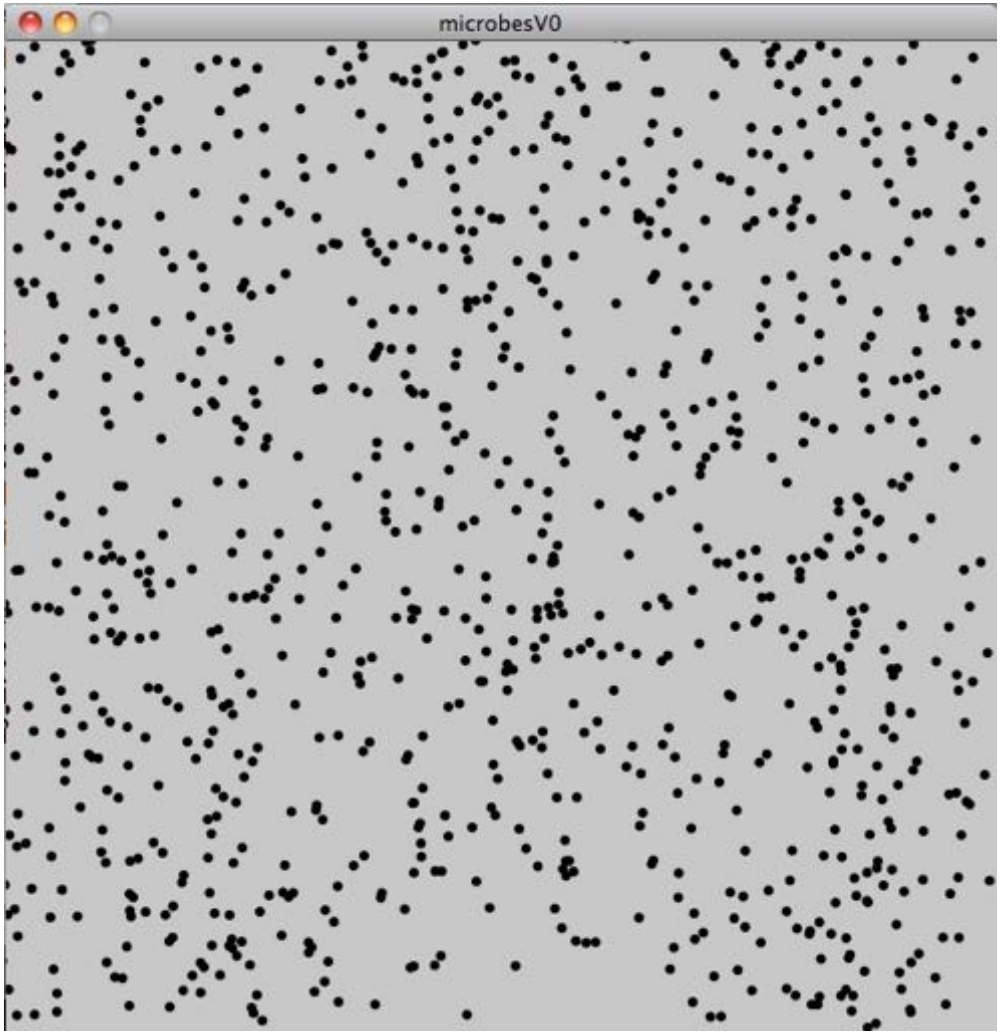
Pointillisme ("Design génératif" P_4_3_1)



```
PImage img;
int Dx, Dy;

void setup() {
  img = loadImage("truc.jpg");
  println(img.width+" x "+img.height);
  size(img.width, img.height);  ////////////// PAS CORRECT
  smooth();
  Dx = int(img.width / 20);
  Dy = int(img.height / 20);
}

void draw() {
  background(255);
  for (int x = 0; x < img.width; x+=Dx) {
   for (int y = 0; y < img.height; y+=Dy) {
    // get current color
    color c = img.pixels[y*img.width+x];
    // greyscale conversion
    int greyscale =round(red(c)*0.222+green(c)*0.707+blue(c)*0.071);
    float w6 = map(greyscale, 0,255, 25,0);
    noStroke();
    fill(c);
    ellipse(x, y, w6, w6);
    }
  }
}
```

(pointillisme.pde)

le mouvement brownien – version tableaux



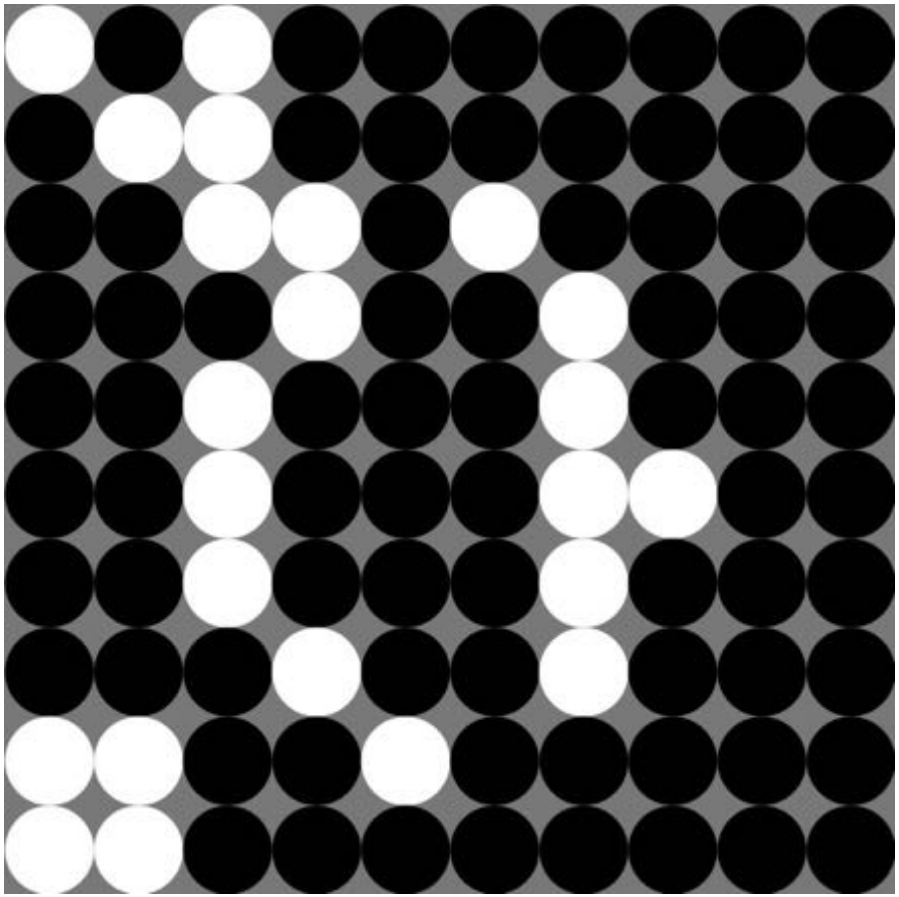
microbesV0

```
final int NBMICROBES = 1000;
float[] mX = new float[NBMICROBES];
float[] mY = new float[NBMICROBES];

void setup() {
 size(600,600);
 smooth();
 for (int i=0;i<NBMICROBES;i++) {
  mX[i] = random(0,width);
  mY[i] = random(0,height);
 }
}
```

```
void draw() {
 background(200);
 // dessins
 for (int i=0;i<NBMICROBES;i++) {
  fill(0);noStroke();
  ellipse(mX[i],mY[i],6,6);
 }
 // deplacements
 for (int i=0;i<NBMICROBES;i++) {
  mX[i] = (mX[i]+random(-1,+1))%width;
  mY[i] = (mY[i]+random(-1,+1))%height;
 }
}
```

on contraint
les objets à rester
dans la zone de dessin

Les matrices : ex. des automates cellulaires



Règle du jeu de
la vie : cf wikipedia

```
int DIM = 10;
float R;

int[][]C = new int[DIM][DIM];
int[][]D = new int[DIM][DIM];
int i,j;
float x,y;

void setup(){
  size(600,600);
  smooth();
  noStroke();
  R = width/DIM;
  frameRate(1);
  for (i=0;i<DIM;i++)
   for (j=0;j<DIM;j++)
    {if (random(1)<0.5) C[i][j]=0; else C[i][j]=1;}

}

void draw(){
 background(120);
 // calcul nvelle etape
 for (i=0;i<DIM;i++)
  for (j=0;j<DIM;j++){
   int m=SommeMoore(i,j);
   if (C[i][j]==0)
     if (m==3) D[i][j]=1; else D[i][j]=0;
   else
     if ((m==2)||(m==3)) D[i][j]=1; else D[i][j]=0;
  }
```

```
// affichage resultat
 x = R/2; y = R/2;
 for (i=0;i<DIM;i++){
  for (j=0;j<DIM;j++){
    if (D[i][j]==0) fill(0); else fill(255);
    ellipse(x,y,R,R);
    x += R;
   }
   y += R;x = R/2;
 }

 // recopie etat courant
 for (i=0;i<DIM;i++) for (j=0;j<DIM;j++) C[i][j]=D[i][j];

 println(frameCount);
}
```

```
void keyPressed(){
  noLoop();
  save("jeu"+frameCount+".png");
}

int SommeMoore(int i, int j) {
 int im1 = (i == 0)?DIM-1:i-1;
 int ip1 = (i == DIM-1)?0:i+1;
 int jm1 = (j == 0)?DIM-1:j-1;
 int jp1 = (j == DIM-1)?0:j+1;
 //int s= C[i][j];
 int s = 0;
 s+= C[im1][jm1];
 s+= C[im1][j];
 s+= C[im1][jp1];
 s+= C[i][jm1];
 s+= C[i][jp1];
 s+= C[ip1][jm1];
 s+= C[ip1][j];
 s+= C[ip1][jp1];
 return s;
}
```

(jeudelavide.pde)

Le bouton, avec classe

Comment faire si on veut 2 (ou 100) boutons ???
- 4 paramètres par boutons : x, y, rollover, selected
- lourdeur de la boucle draw()
- test de detection de zone ??

$\Rightarrow$avantage décisif de la programmation "objets"

on va créer une **classe** Bouton qui regroupe les traitements de dessin et la gestion des événements et les paramètres propres à chaque bouton

Squelette du code de la classe :

```
class Button {
  /// ici declaration des attributs x,y, selected, rollover

  Button(float Px, float Py, boolean Pselected) {
    x = Px;
    y = Py;
    selected = Pselected;
  }

  void display() {
   /// ici code d'affichage
  }

  void rollover(int mx, int my) {
   /// ici code detection de rollover
  }

  void clic(int mx, int my) {
   //// ici code gestion du booleen selected
  }
}
```

constructeur pour les objets Button

les autres méthodes

Code complet de la classe

```
class Button {
  boolean selected = false;
  boolean rollover = false;
  float x,y;

  Button(float Px, float Py, boolean Pselected) {
    x = Px;
    y = Py;
    selected = Pselected;
  }

  void display() {
    stroke(0);noFill();
    if (rollover) strokeWeight(4);
    else strokeWeight(1);
    rect(x,y,30,30);
    if (selected) {
      noStroke();fill(0);
      rect(x+10,y+10,10,10);
    }
}
```

```
  void rollover(int mx, int my) {
    if (mx > x && mx < x + 30 && my > y && my < y + 30)
      rollover = true;
    else
      rollover = false;
  }

  void clic(int mx, int my) {
    if (rollover) selected = ! selected;
  }

}
```

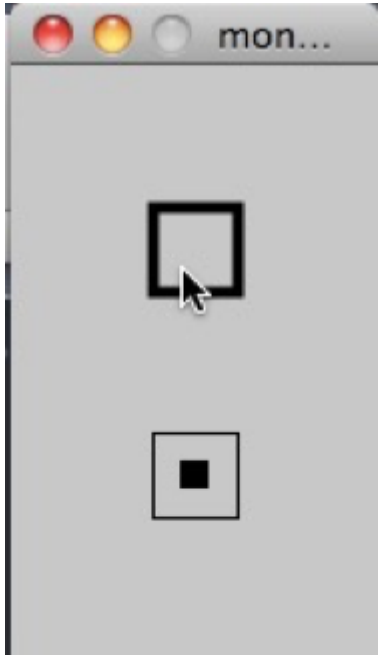
Utilisation dans le programme Processing

monboutonCLASSE | Processing 1.5.1

STANDARD

monboutonCLASSE | Bouton

```
Button b1,b2;

void setup() {
  size(130,210);
  smooth();
  b1 = new Button(50,50,false);
  b2 = new Button(50,130,true);
}

void draw() {
  background(250);
  b1.display();
  b2.display();
}

void mouseMoved() {
  b1.rollover(mouseX,mouseY);
  b2.rollover(mouseX,mouseY);
}

void mousePressed() {
  b1.clic(mouseX,mouseY);
  b2.clic(mouseX,mouseY);
}
```



Rque : mettre le code
de la classe dans un fichier
séparé est plus commode

Le mouvement brownien, avec classe !

microbesV0

```
final int NBMICROBES = 1000;
float[] mX = new float[NBMICROBES];
float[] mY = new float[NBMICROBES];

void setup() {
 size(600,600);
 smooth();
 for (int i=0;i<NBMICROBES;i++
  mX[i] = random(0,width);
  mY[i] = random(0,height);
 }
}
```

```
void draw() {
 background(200);
 // dessins
 for (int i=0;i<NBMICROBES;i++) {
  fill(0);noStroke();
  ellipse(mX[i],mY[i],6,6);
 }
 // deplacements
 for (int i=0;i<NBMICROBES;i++) {
  mX[i] = (mX[i]+random(-1,+1))%width;
  mY[i] = (mY[i]+random(-1,+1))%height;
 }
}
```

classe ?
attributs ?
méthodes ?

La classe particule

brownienclasse | particule

```
class particule {
  float x,y;

  particule(){
    x=random(0,width);
    y=random(0,height);
  }

  void dessin(){
    noStroke();fill(0);
    ellipse(x,y,6,6);
  }

  void evolution(){
    x = (x+random(-1,+1))%width;
    y = (y+random(-1,+1))%height;
  }
}
```

Le programme principal

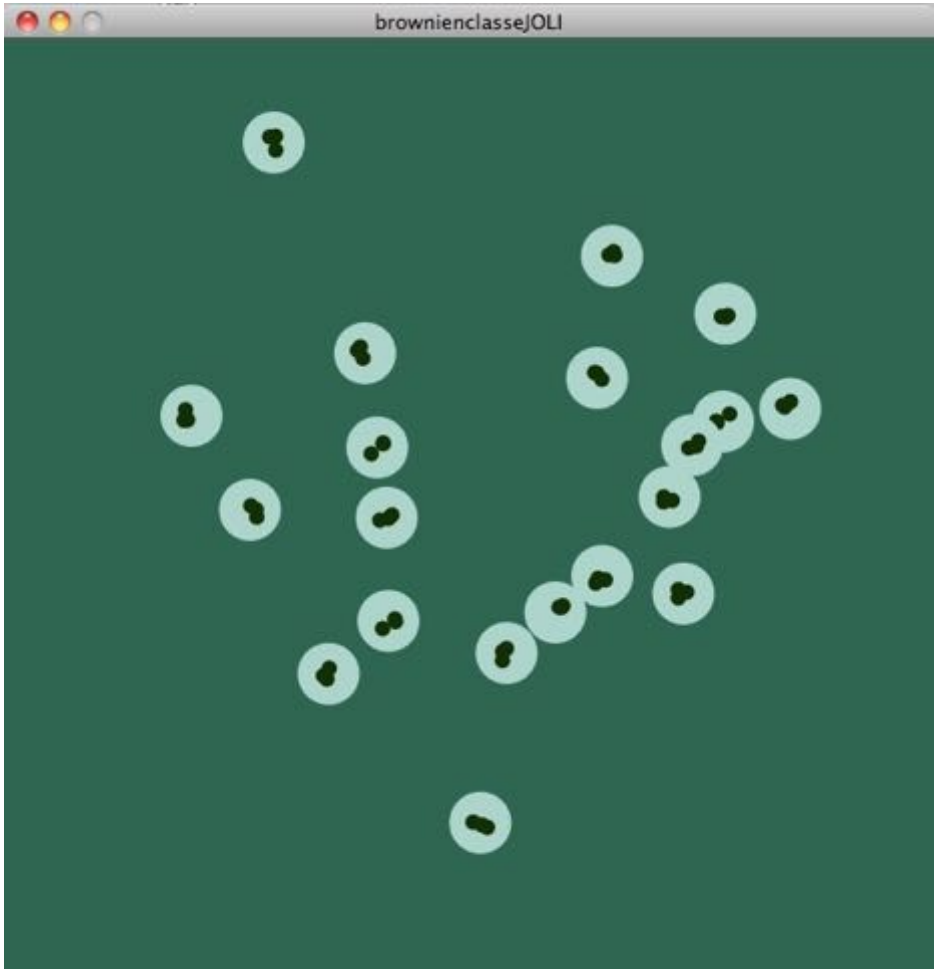
brownienclasse | particule

```
particule[] p = new particule[1000];

void setup() {
 size(600,600);
 smooth();
 for (int i=0;i<p.length;i++) p[i] = new particule();
}

void draw() {
 background(200);
 for (int i=0;i<p.length;i++){
  p[i].dessin();
  p[i].evolution();
 }
}
```

déclaration d'un tableau d'objets

allocation pour chaque objet

Une version plus jolie



Seul le code de la classe Particule est impacté :-)

Nouvelle méthode dessin() :

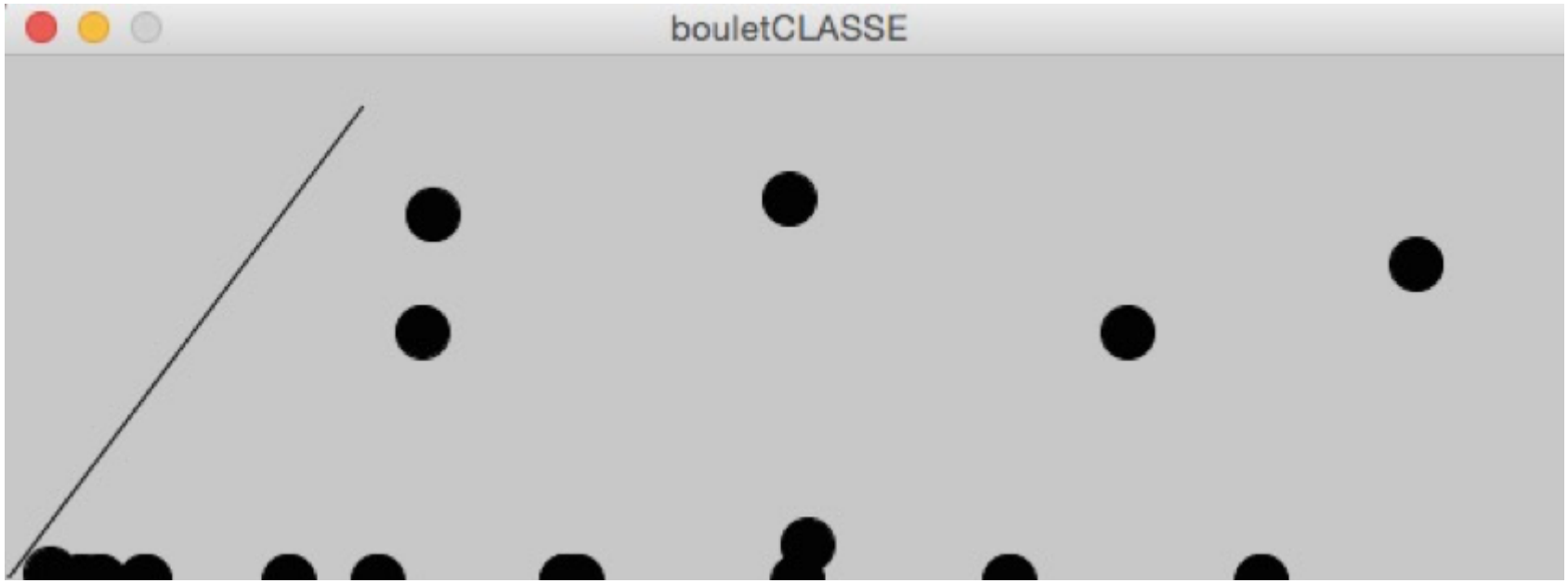
```
void dessin(){
  noStroke();fill(#A2D8CB);
  ellipse(x,y,40,40);
  fill(0,50,0);
  ellipse(random(x-5,x+5),random(y-5,y+5),10,10);
  ellipse(random(x-5,x+5),random(y-5,y+5),10,10);
  ellipse(random(x-5,x+5),random(y-5,y+5),10,10);
}
```

avec quand même un nouveau background(#0A6A52) dans la fonction draw()

Rque : pour choisir des couleurs, utiliser le sélecteur de couleurs dans le menu Tools de Processing

C. Sommerer & L. Mignonneau. Portrait on the fly

Le boulet : passage à une version en classe

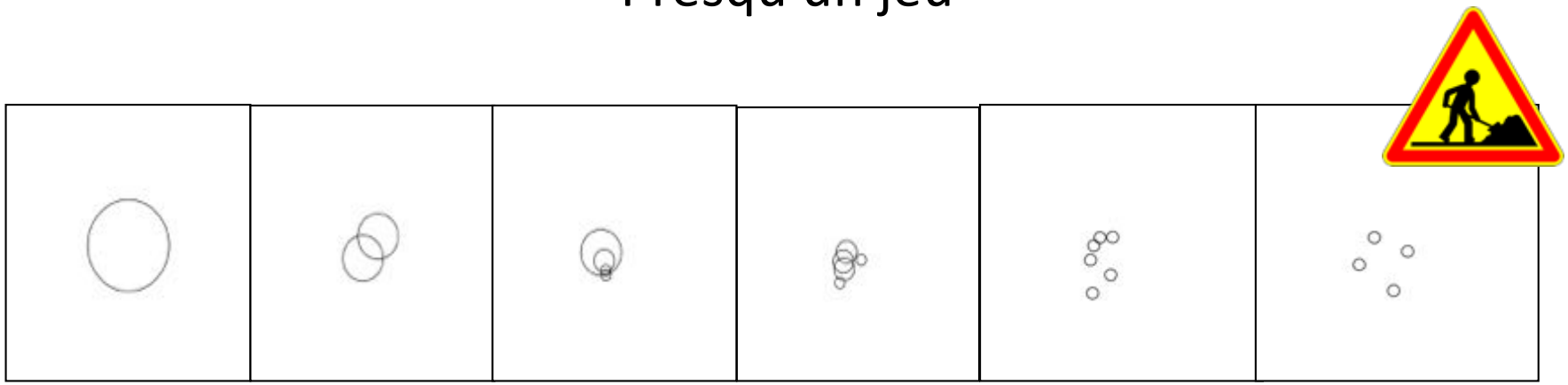


```
class balle {
  float x,y;
  float vx,vy;
  float ax,ay;
  balle(){
    vx=0.1*mouseX;
    vy=0.1*(height-mouseY);
    x=0;
    y=0;
  }
  void evolution(){
   ax = 0;
   ay = -1.0; // ici G = 1 m/s2
   vx = vx + ax;
   vy = vy + ay;
   x = x + vx;
   y = y + vy;
   if (y<0) { vy *= -0.9;y = 0;} // collision sol
   if (y>height){ vy *= -0.9; y = height;} // collision plafond
   if (x<0) { vx *= -0.9; x = 0; } // collision mur gauche
   if (x>width){ vx *= -0.9; x = width;} // collision mur droit
   }

  void dessin(){
   ellipse(x,height-y,20,20);
  }
}
```

```
balle[] b = new balle[100];
int N=0;

void setup(){
  size(600,200);
  smooth();fill(0);stroke(0);
}

void draw()  {
  background(200);
  line(0,height,mouseX,mouseY);
  for (int i=0;i<N;i++)b[i].dessin();
  for (int i=0;i<N;i++)b[i].evolution();
 }

void keyPressed(){
  println(N);
  b[N] = new balle();
  N++;
  N = constrain(N,1,100);
}
```

(bouletCLASSE.pde)

Presqu’un jeu



Au début, il y a sur l'écran une seule particule en mouvement, de grande taille (diamètre 100 pixels).

Si l'utilisateur clique dans l'intérieur de la particule, celle-ci se divise en 2 particules, de diamètre moitié (donc 50 pixels à ce stade).

A chaque fois que l'utilisateur clique dans une particule, le phénomène se reproduit.

Quand le diamètre d'une particule descend au dessous d'un seuil fixé à l'avance (20 pixels, par ex.), la particule est détruite.

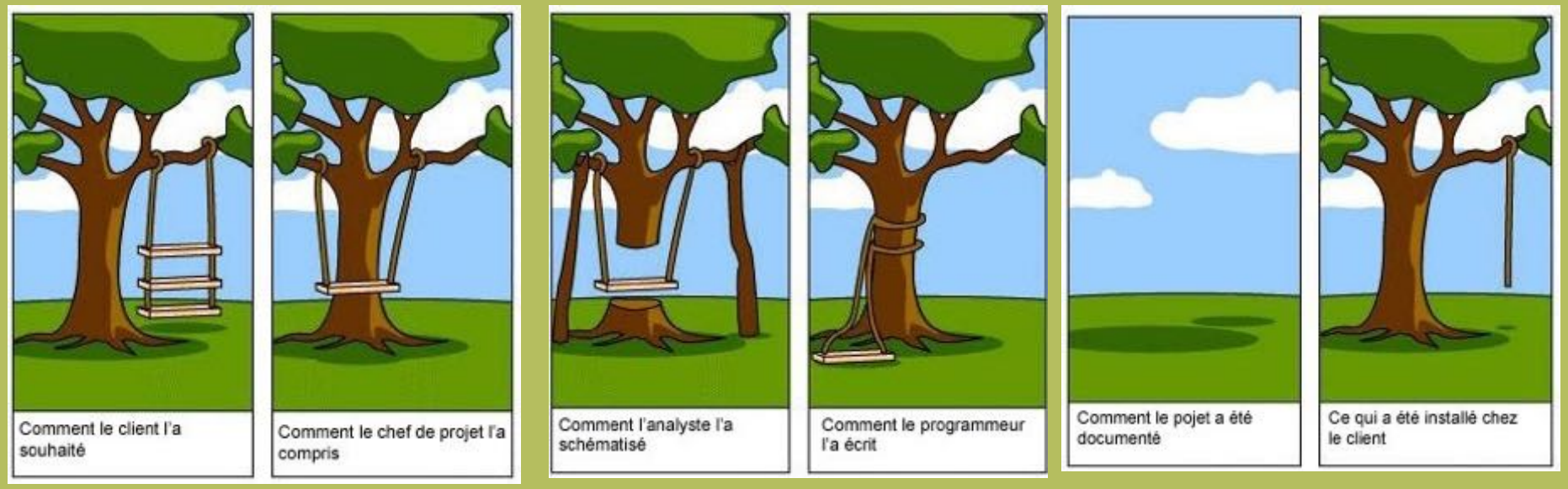
```
class Brown {
  float x; float y; float dia;
  Brown(float px, float py, float pdia){
    this.x = px; this.y = py; this.dia = pdia;
  }
  void display(){
    ellipse(x,y,dia,dia);
  }
  void move(){
    x += random(-3,+3);
    y += random(-3,+3);
  }
  boolean finished(){
    return (dia <= 20);
  }
  boolean rollOver(){
    boolean test = (mouseX > x-dia/2)&&(mouseX < x+dia/2 );
    test = test && (mouseY > y-dia/2)&&(mouseY < y+dia/2 );
    return test;
  }
}
```

```
ArrayList<Brown> lesbrowns;

void setup() {
  size(600, 600);
  ellipseMode(CENTER);
  noFill();stroke(0);strokeWeight(2);
  lesbrowns = new ArrayList<Brown>();
  lesbrowns.add(new Brown(width/2, height/2, 200));
}

void draw() {
  background(255);
  for (int i = 0; i<lesbrowns.size(); i++) {
    Brown b = lesbrowns.get(i);
    b.move();
    b.display();
    if (b.finished()) {
      lesbrowns.remove(i);
    }
  }
  if (lesbrowns.size() == 0) {
    fill(0);text("game over",width/2,height/2);}
}
```

```
void mouseReleased() {
 for (int i = 0; i<lesbrowns.size(); i++) {
    Brown b = lesbrowns.get(i);
    if (b.rollOver()) {
     // on divise la particule en deux plus petites
     b.dia /= 2;
     lesbrowns.add(new Brown(b.x, b.y, b.dia));
     break; // on traite une seule particule dans le for
    }
  }
}

void keyPressed(){
  saveFrame("minijeu#####.png");
}
```

(minijeu.pde)

Pour conclure



« Le simple n’est pas le facile » (Le Corbusier)

(à lire plus tard)

Cycle de développement (logiciel)

Il existe différents types de **cycles de développement** entrant dans la réalisation d'un logiciel. Ces cycles prendront en compte toutes les étapes de la conception d'un logiciel.

Sommaire [masquer]

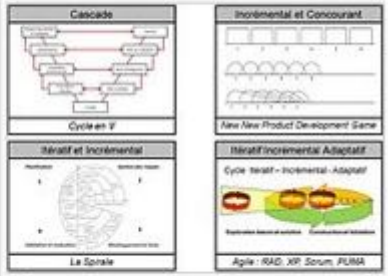
Les grandes familles [modifier le code]

Modèle en cascade [modifier le code]

Le modèle en cascade[1] est hérité de l'industrie du BTP. Ce modèle repose sur les hypothèses suivantes :

- on ne peut pas construire la toiture avant les fondations ;
- les conséquences d'une modification en amont du cycle ont un impact majeur sur les coûts en aval (on peut imaginer la fabrication d'un moule dans l'industrie du plastique).

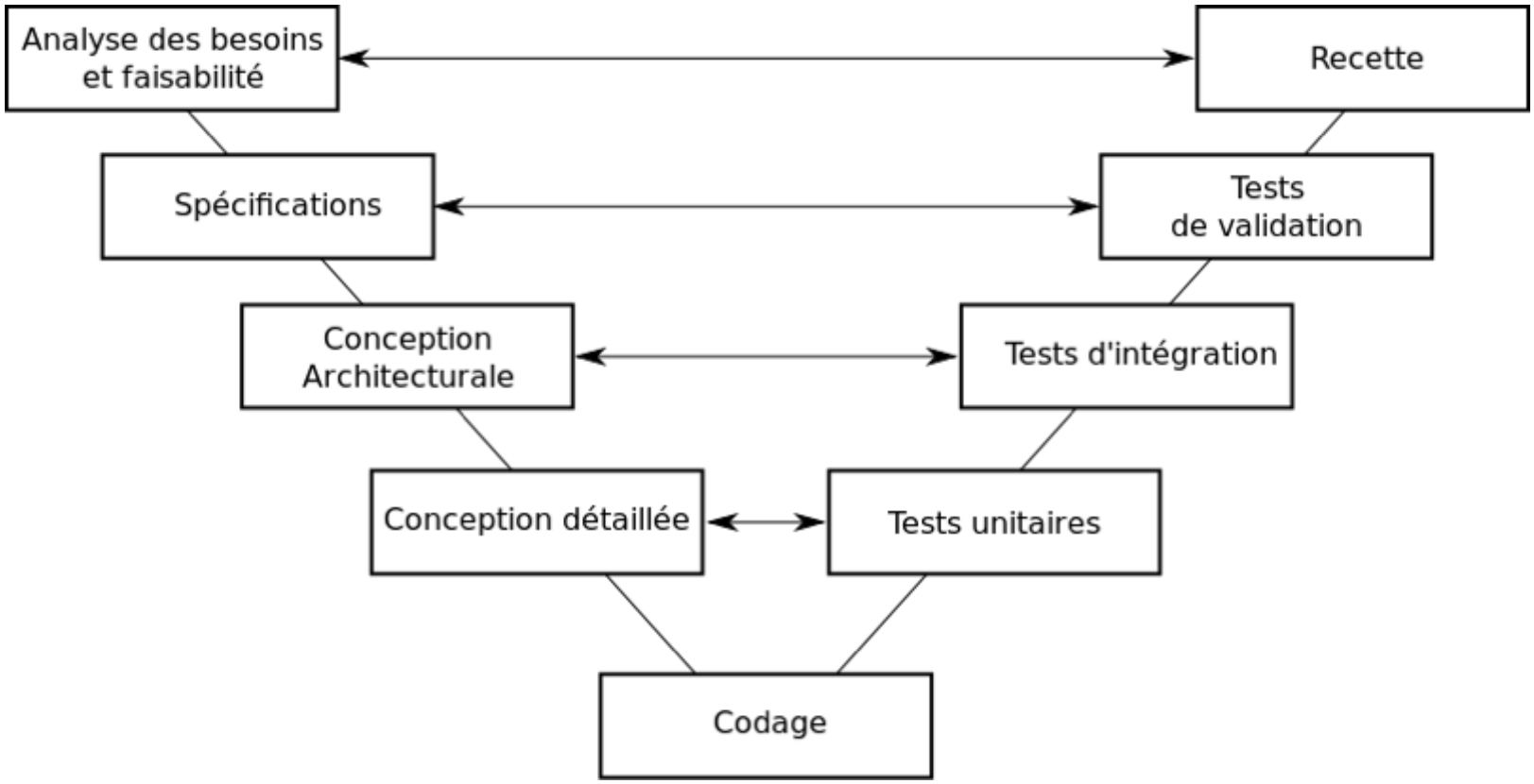
Les phases traditionnelles de développement sont effectuées simplement les unes après les autres, avec un retour sur les précédentes, voire au tout début du cycle.



Évolution des cycles basiques

Gros code : des étapes ≠ et des acteurs ≠

exemple du cycle en V



autres approches : méthodes "agiles" RAD

La phase de conception

- concevoir n'est pas coder !!!! et réciproquement

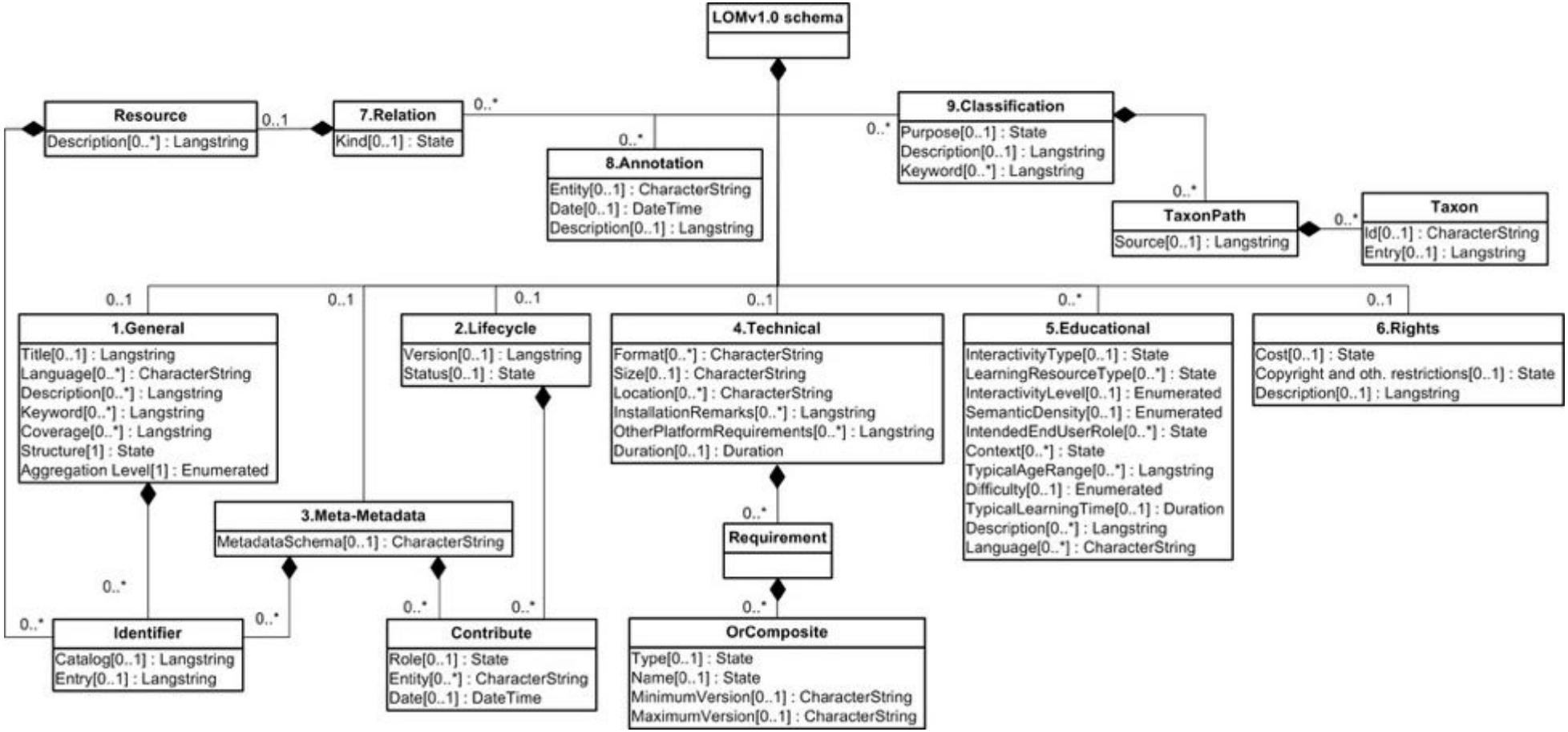
- en gros : définir les entités (classes), leurs relations
étudier aussi la dynamique temporelle (processus, interaction)
et l'implantation (si application répartie)

- il existe de très nombreuses méthodes pour assister
le concepteur

- Exemple de UML : Unified Modeling Language (fin 90s)

- les "Design patterns"

Exemple : diagrammes de classe en UML



≠ types de relation : aggregation, composition, héritage

on peut déduire le squelette du code Java correspondant automatiquement