Informatique

Olivier Baudon - LaBRI

2022

université de BORDEAUX

Qu’est-ce que l’informatique ?

université de BORDEAUX

Qu’est-ce que c’est ?

Vient de la contraction de INFORmation et autoMATIQUE

INFORMATIQUE:
Science du traitement automatique et rationnel de l'information considérée comme le support des connaissances et des communications. *Larousse*

Ensemble des sciences et techniques en rapport avec le traitement de l'information. *Wikipedia*

Qu’est-ce que ce n’est pas ?

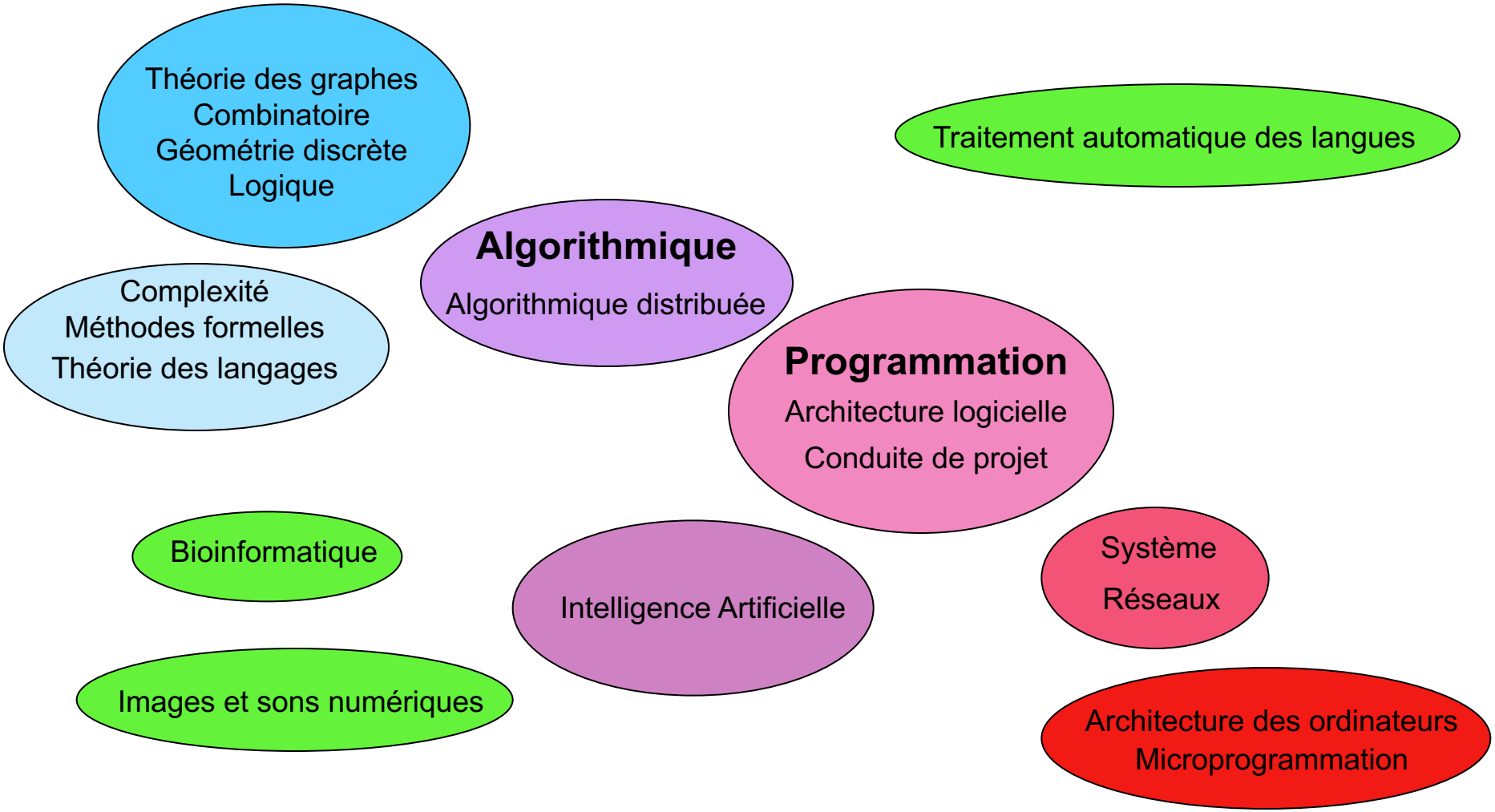
"Computer science is no more about computers than astronomy is about telescopes."

généralement attribuée à Edsger Dijkstra

L´informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes

Où en trouve-t-on ?

Que fait-on ?


Théorie des graphes
Combinatoire
Géométrie discrète
Logique
Traitement automatique des langues
Algorithmique
Algorithmique distribuée
Complexité
Méthodes formelles
Théorie des langages
Programmation
Architecture logicielle
Conduite de projet
Bioinformatique
Intelligence Artificielle
Système
Réseaux
Images et sons numériques
Architecture des ordinateurs
Microprogrammation

Algorithmique

Un algorithme est un énoncé d'une suite finie et non-ambiguë d'opérations permettant de donner la réponse à un problème.

Wikipédia

Exemple :
Chercher le minimum d'un tableau d'entiers *t* à *n* éléments: t[0], …, t[n-1]

1 - min ← t[0]
2 - pour i allant de 1 à n-1 faire
3 - si t[i] < min
4 - min ← t[i]

i	0	1	2	3	4
t[i]	**5**	**3**	**7**	**1**	**9**

i = 2 4

min	**1**

Programmation

Traduire dans un langage de programmation un ou plusieurs algorithmes.

Pour écrire un programme :
- définir le où les algorithmes nécessaires,
- les traduire dans le langage de programmation choisi,
- traduire le programme ainsi obtenu en langage « machine » grâce à un compilateur ou un interprèteur.

Programme Python

Exemple :
Chercher le minimum d'un tableau d'entiers *t* à *n* éléments: t[0], …, t[n-1]

1 - min ← t[0]
2 - pour i allant de 1 à n-1 faire
3 - si t[i] < min
4 - min ← t[i]

i	0	1	2	3	4
t[i]	**5**	**3**	**7**	**1**	**9**

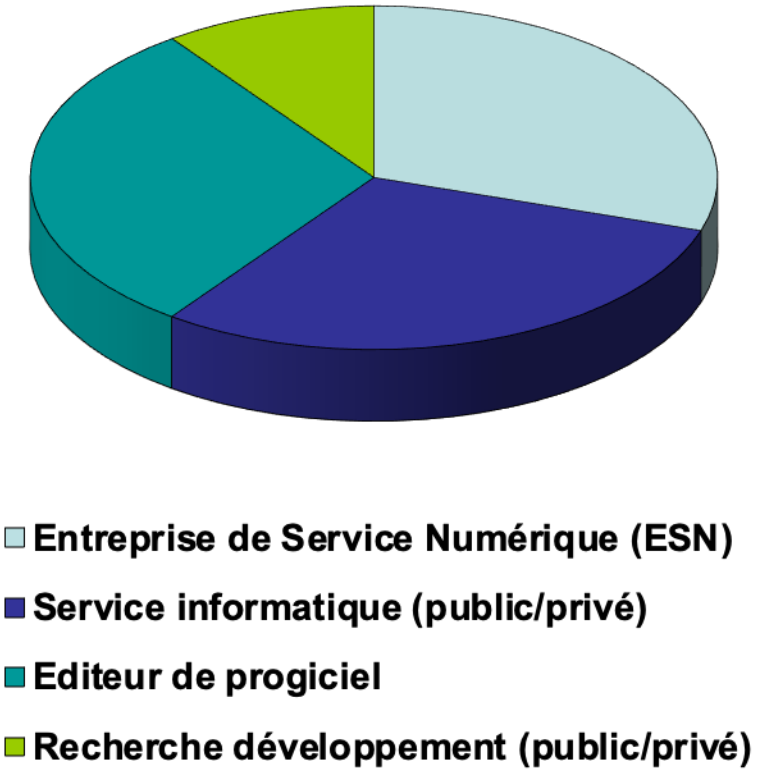
Programme Python
➔

```
t = [5, 3, 7, 1, 9]
n = len(t)
min = t[0]
for i in range(1,n) :
    if t[i] < min :
        min = t[i]
print("min de", t, "=", min)
```

METIERS : Niveaux

- Technicien
  - Bac STMG ou Professionnel
- Technicien Supérieur
  - BTS (2 ans)
- Assistant Ingénieur
  - Après un BUT ou une Licence Professionnelle (3 ans)
- Ingénieur
  - Après une école d'ingénieurs ou un Master (5 ans)
- Enseignant
  - CAPES NSI ou Agrégation d'informatique (5 ans)
- Chercheur, Enseignant-chercheur, Ingénieur de recherche
  - Après un doctorat (8 ans)

METIERS : Sociétés



METIERS : Activité

- Maintenance (Bac à Bac + 2)
- Développeur (Bac+2 à Bac+5)
- Système – Réseau – Sécurité (Bac+2 à Bac +5)
- Systèmes d'informations (Bac+2 à Bac+5)
- Images, Sons, Multimédia (Bac+3 à Bac+5)
- Double compétence (Bac+5)
  - Automatique,
  - Electronique,
  - Bio-informatique,
  - Mathématiques appliquées
  - …
- Chercheur (Bac+5 à Bac+8)

LaBRI *Laboratoire Bordelais de Recherche en Informatique*

- Unité de Recherche mixte CNRS, Université de Bordeaux, Institut Polytechnique de Bordeaux
- Équipes associées à INRIA, Bordeaux–Sud-Ouest
- > 300 personnes (Chercheurs, Enseignants-chercheurs, Ingénieurs, Administratifs, Doctorants)

LaBRI

Départements

- Combinatoire et Algorithmique
- Image et Son
- Méthodes et Modèles Formels
- Systèmes et Données
- Supports et AlgoriThmes pour les Applications Numériques hAutes performanceS (SATANAS)

Axes

- Intelligence artificielle
- *Santé numérique*

Nomenclature CNU 27

1 Systèmes d'information
11 BD, SGBD, gestion des données, entrepôts, fouille de données
12 Recherche d'information, ingénierie des documents, information multimédia
13 Ingénierie des SI (méthodes et modèles pour la conception, processus, SI spécifiques)
14 Web sémantique

2 Algorithmique, recherche opérationnelle
21 Recherche opérationnelle, optimisation combinatoire
22 Graphes, combinatoire, algorithmique géométrique
23 Algorithmique distribuée, parallèle
24 Calculabilité et complexité

Nomenclature CNU 27

3 Informatique fondamentale
31 Informatique théorique, théorie des langages, modèles de calcul
32 Calcul formel, interface mathématique et informatique, cryptographie, codage
33 Logique et fondements de la programmation

4 Réseaux
41 Architecture, gestion, sécurité
42 Protocoles, QoS, multimédia, mobilité
43 Métrologie, évaluation de performances

5 Bioinformatique
51 Inférence et analyse de séquences/réseaux
52 Stockage et fouille
53 Modélisation et simulation (molécules, dynamique des réseaux)

Nomenclature CNU 27

6 Systèmes informatiques
61 Systèmes d'exploitation et intergiciels
62 Modèles, spécifications, validation, vérification
63 Systèmes critiques, embarqués, temps réel
64 Systèmes répartis et distribués
65 Sécurité des systèmes informatiques

7 Génie logiciel et programmation
71 Ingénierie des exigences, méthodes de développement, gestion des processus logiciels
72 Ingénierie pilotée par les modèles
73 Approches formelles, spécification, vérification, preuve, validation, test
74 Architecture logicielle (composants, lignes de produits, services)
75 Méthodes de programmation et paradigmes
76 Langages, compilation, génération de code, interprétation

Nomenclature CNU 27

8 Intelligence Artificielle
81 Apprentissage
82 Logique, raisonnement, ingénierie des connaissances
83 Planification
84 TALN
85 Contraintes, méta-heuristiques
86 Systèmes multi-agents, modélisation cognitive

9 Images et géométrie, scènes, parole, signaux
91 Analyse d'objets, d'images, de scènes
92 Traitement de données audiovisuelles et multimédia
93 Perception et vision par ordinateur, réalité augmentée
94 Informatique graphique, synthèse d'images

Nomenclature CNU 27

10 Communication homme-machine
A1 Environnements Informatiques pour l'Apprentissage (EIAO, EIAH)
A2 Communication homme-machine (compagnons artificiels, affect, dialogue)
A3 Analyse de documents
A4 Interaction homme-machine (interface, travail coopératif, collecticiel)

11 Architecture des machines
B1 Architecture des ordinateurs, processeurs, multi-processeurs, systèmes mémoire
B2 Méthodes de conception, de vérification et de test de matériel
B3 Architectures spécialisées, systèmes numériques intégrés sur la puce, systèmes embarqués

Nomenclature CNU 27

12 Informatique industrielle
C1 Architecture dédiée, conception, systèmes sur puces ou embarqués
C2 Systèmes temps réel, ordonnancement, systèmes d'exploitation
C3 Modèles pour les systèmes à événements discrets

13 Modélisation-simulation pour les systèmes complexes (systèmes artificiels et naturels)
D1 Formalismes de modélisation
D2 Simulation distribuée
D3 Vérification, validation de modèles de simulation
D4 Transformations de modèles, génération de code à partir des modèles
D5 Couplages de modèles, interactions entre systèmes discrets

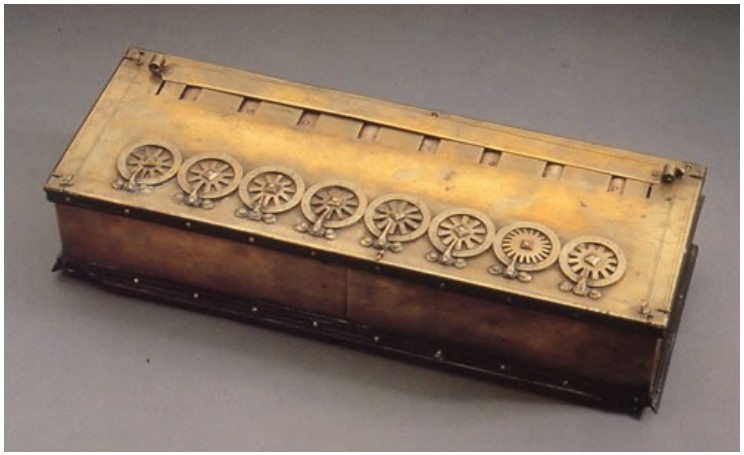
Histoire de l’ informatique

université de BORDEAUX

Histoire de l'Informatique

D'après « Une très brève histoire de l'informatique », traduction d'Anne Dicky de « A Very Brief History of Computer Science », texte écrit en 1995 et revu en 1999 par Jeffrey Shallit pour ses étudiants de l'Université de Waterloo (Canada).

-3000 : abaques babyloniennes
-80 : le mécanisme d'Antikythera, servait à prédire les mouvements des astres.
1641 : machine à additionner de Blaise Pascal
Leibniz (1646-1716) préconise l'usage du système binaire pour les machines à calculer



Histoire de l'Informatique

Joseph-Marie Jacquard (1752-1834) inventa un métier à tisser dont les motifs étaient indiqués par des cartons perforés
Charles Babbage (1791-1871) eut l'idée d'incorporer dans une machine à calculer des cartes du métier Jacquard, dont la lecture séquentielle donnerait des instructions et des données à sa machine. Ada Lovelace l'aide à concevoir les « diagrammes » pour faire fonctionner la machine et publie le premier algorithme destiné à être exécuté sur une machine.

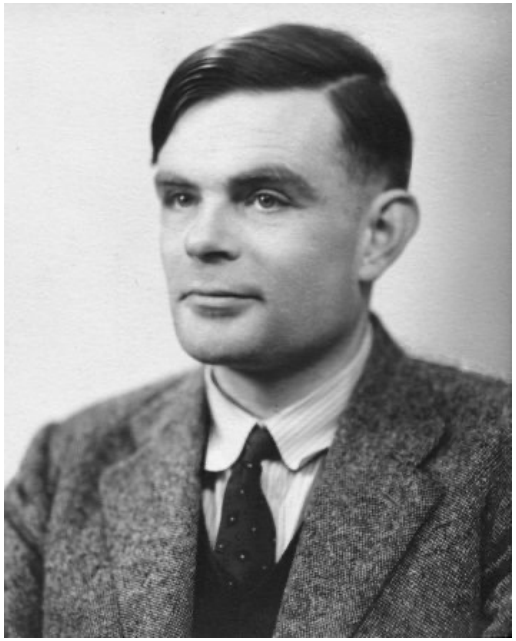
Histoire de l'Informatique

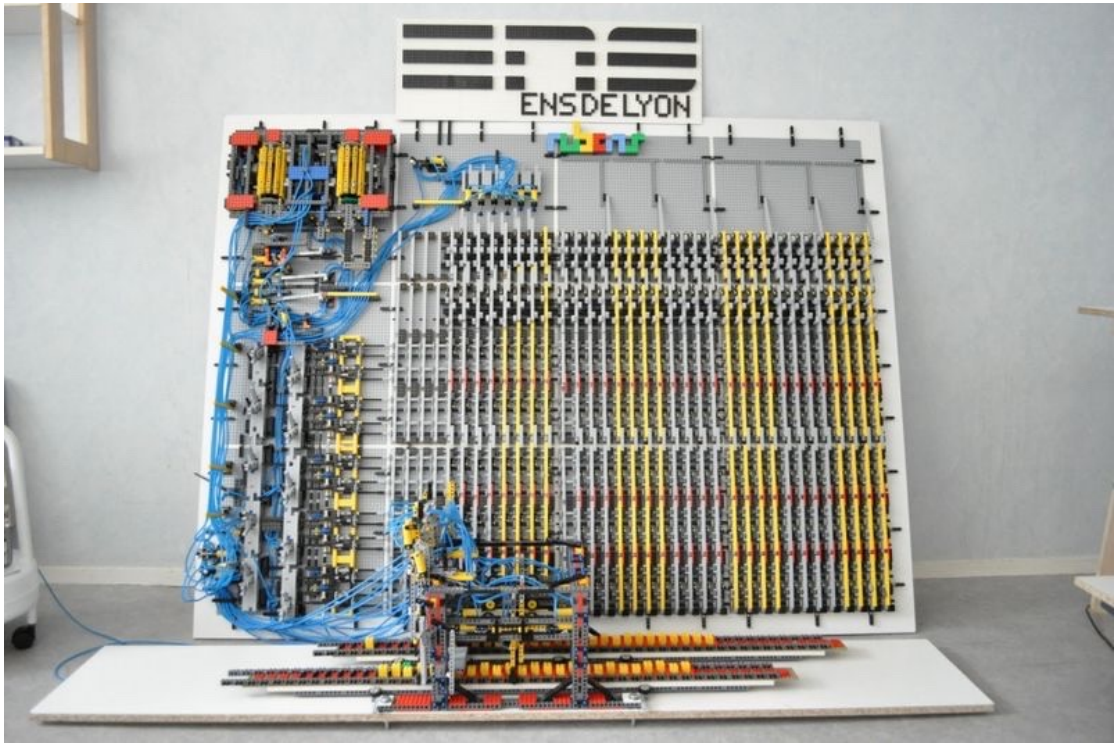
En 1928, le mathématicien David Hilbert (1862-1943) posa trois questions au Congrès International des Mathématiciens :
(1)Les mathématiques sont-elles complètes ? (tout énoncé mathématique peut-il être soit prouvé, soit réfuté ?)
(2)Les mathématiques sont-elles cohérentes ? (peut-on être sûr que des raisonnements valides ne conduiront pas à des absurdités ?)
(3)Les mathématiques sont-elles décidables ? (existe-t-il un algorithme pouvant dire de n'importe quel énoncé mathématique s'il est vrai ou faux ?) Cette dernière question est connue sous le nom de *Entscheidungsproblem*.

En 1931, Kurt Gödel (1906-1978) répondit à deux de ces questions. Il démontra que tout système formel suffisamment puissant est soit incohérent, soit incomplet. De plus, si un système d'axiomes est cohérent, cette cohérence ne peut être prouvée en n'utilisant que les axiomes. La troisième question restait ouverte, en remplaçant « vrai » par « prouvable » (existe-t-il un algorithme pour dire si une assertion peut être prouvée ?)

Histoire de l'Informatique

En 1936, Alan Turing (1912-1954) résolut l'Entscheidungsproblem en construisant un modèle formel de calculateur - la machine de Turing - et en prouvant qu'une telle machine ne pouvait pas résoudre certains problèmes, en particulier le problème d'arrêt : étant donné un programme, peut-on dire s'il termine pour n'importe quelle valeur des données ?





Histoire de l'Informatique

Exemple : supposons qu'il existe une fonction estNulle ayant comme paramètres une fonction f de $\mathbb{N}$ vers $\mathbb{N}$ et un entier n, et qui renvoie vrai si f(n) est nul, faux sinon.
Il est relativement facile par l'absurde de montrer qu'une telle fonction ne peut exister !
Supposons que estNulle(int f(int), int n) existe et soit auto la fonction définie par :

```
auto(int n) :
    si estNulle(auto, n) alors
        renvoyer 1
    sinon
        renvoyer 0
```

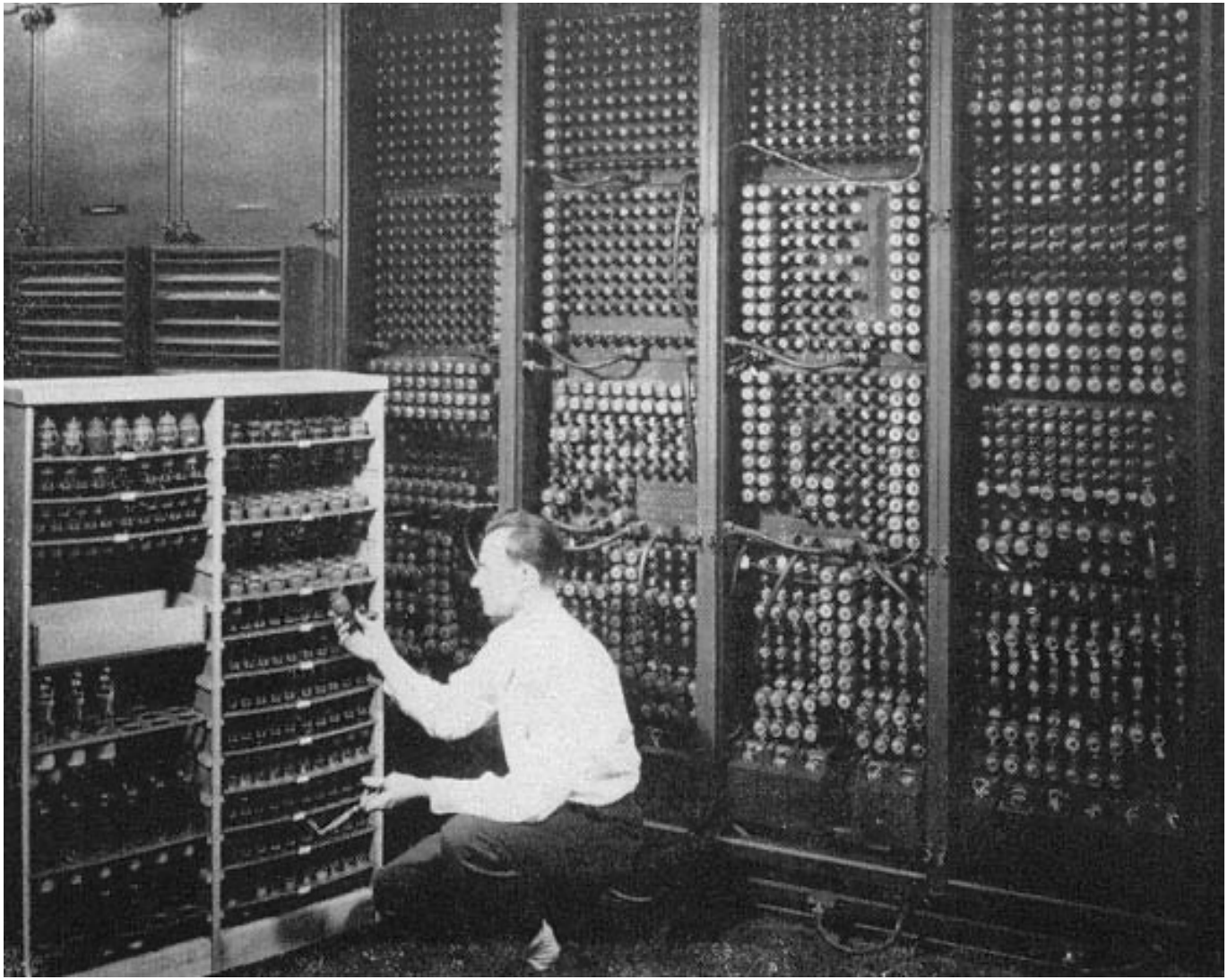
Quel sera le résultat de auto(n), quelle que soit la valeur de n choisie ?

Histoire de l'Informatique

En 1944, à Harvard, Howard Aiken (1900-1973) construisit le calculateur électromécanique Mark I, avec l'aide d'IBM.
Alan Turing, en Angleterre, travaillait à décoder la machine allemande Enigma; les Anglais construisirent un calculateur, le Colossus, pour aider au décryptage.
En 1939, à l'Université d'Iowa, John Atanasoff (1904-1995) et Clifford Berry conçurent et réalisèrent l'ABC, un calculateur électronique pour résoudre des systèmes d'équations linéaires, mais il ne fonctionna jamais correctement.

Atanasoff discuta de son invention avec John Mauchly (1907-1980), qui, plus tard, avec John Eckert (1919-1995), conçut et réalisa l'ENIAC, un calculateur électronique destiné à l'origine aux calculs balistiques. On ne sait pas très bien quelles idées Atanasoff transmit à Mauchly; le mérite d'avoir inventé le premier ordinateur revient-il à Atanasoff ou à Mauchly et Eckert ? Ce fut le sujet de batailles juridiques, c'est encore celui d'un débat historique. L'ENIAC fut construit à l'Université de Pennsylvanie, et terminé en 1946.

Histoire de l'Informatique



Histoire de l'Informatique

En 1944, Mauchly, Eckert, et John von Neumann (1903-1957) travaillaient à la conception d'un ordinateur électronique, l'EDVAC. Eckert et Mauchly reprirent ces idées pour construire l'UNIVAC.
Pendant ce temps, en Allemagne, Konrad Zuse (1910-1995) construisait le premier calculateur programmable universel (non spécialisé), le Z3 (1941).
En Angleterre, Maurice Wilkes (1913-2010) construisit l'EDSAC (à partir de l'EDVAC). Frederic C. Williams (1911-1977) et son équipe construisirent le Manchester Mark I, dont une version fut opérationnelle dès juin 1948. Certains considèrent cette machine comme le premier ordinateur à programme en mémoire (architecture dite de Von Neumann).
L'invention du transistor en 1947 par John Bardeen, Walter Brattain et William Shockley transforma l'ordinateur, et permit la révolution du microprocesseur. Pour cette découverte, ils reçurent le Prix Nobel de Physique en 1956.
Jay Forrester (1918-2016) inventa vers 1949 la mémoire à noyau magnétique.

Histoire de l'Informatique

Grace Hopper (1906-1992) inventa la notion de compilateur (1951). (Quelques années plus tôt, elle avait trouvé le premier bug de l'histoire de l'informatique, une phalène entrée dans le Mark II de Harvard.)
Elle a aussi participé à la conception du langage COBOL en 1959.

Histoire de l'Informatique

John Backus et son équipe écrivirent le premier compilateur FORTRAN en avril 1957. LISP (List Processing), un langage de traitement de listes pour l'intelligence artificielle, fut inventé par John McCarthy vers 1958. Alan Perlis, John Backus, Peter Naur et leurs associés développèrent Algol (Algorithmic Language) en 1959. Jack Kilby (Texas Instruments) et Robert Noyce (Fairchild Semiconductor) inventèrent les circuits intégrés en 1959.
Edsger Dijkstra (1930-2002) trouva un algorithme efficace pour résoudre le problème des plus courts chemins dans un graphe, à titre de démonstration pour l'ARMAC en 1956. Il trouva aussi un algorithme efficace de recherche d'un arbre recouvrant de poids minimal, afin de minimiser le câblage du X1. (Dijkstra est célèbre pour ses déclarations caustiques et péremptoires).

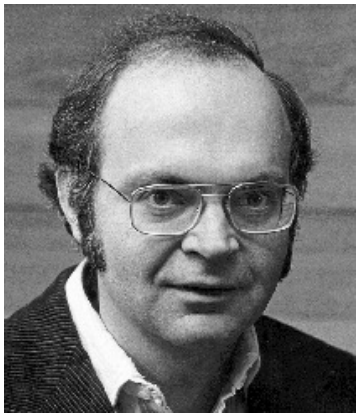
Histoire de l'Informatique

Dans un célèbre article de la revue Mind, en 1950, Alan Turing décrivit le test de Turing, l'une des premières avancées en intelligence artificielle. Il proposait une définition de la « pensée » ou de la « conscience » relative à un jeu : un examinateur pose des questions par écrit à un interlocuteur situé dans la pièce voisine, et doit décider, au vu des réponses, si son interlocuteur est une machine ou un être humain.

S'il est incapable de répondre, on peut raisonnablement dire que l'ordinateur « pense ». En 1952, Alan Turing fut arrêté pour outrage aux bonnes moeurs après qu'une plainte pour cambriolage eut révélé sa liaison avec Arnold Murray. L'homosexualité affichée était tabou dans l'Angleterre des années 1950, et on obligea Turing à suivre un « traitement » hormonal qui le rendit impuissant et lui fit pousser des seins. Le 7 juin 1954, Turing se suicida en mangeant une pomme enrobée de cyanure.

Histoire de l'Informatique

Dans les années 1960, l'informatique devint une discipline à part entière. Le premier département d'informatique fut créé en 1962 à l'Université de Purdue; le premier Ph.D. d'informatique fut délivré à Richard Wexelblat par l'Université de Pennsylvanie, en décembre 1965.
Les années 1960 virent émerger la théorie des automates et des langages formels : on peut notamment citer Noam Chomsky et Michael Rabin.
Vers la fin de la décennie, on commença à construire ARPAnet, le précurseur d'Internet.
Ted Hoff (né en 1937) et Federico Faggin (Intel) conçurent le premier microprocesseur en 1969-1971.
Donald Knuth (né en 1938), auteur du traité *The Art of Computer Programming*, posa des fondements mathématiques rigoureux pour l'analyse des algorithmes.



Histoire de l'Informatique

Dans les années 1970, les travaux d'Edgar Codd (1924-2003) sur les bases de données relationnelles permirent une avancée majeure dans la théorie des bases de données. Codd reçut le Turing Award en 1961.
Le système d'exploitation Unix fut développé aux Bell Laboratories par Ken Thompson (né en 1943) et Dennis Ritchie (1941-2011). Brian Kernighan et Ritchie développèrent C, un important langage de programmation.
On vit apparaître de nouveaux langages, tels que Pascal (inventé par Niklaus Wirth) et Ada (réalisé par une équipe dirigée par Jean Ichbiah). La première architecture RISC fut commencée par John Cocke en 1975, chez IBM. Vers cette époque, des projets analogues démarrèrent à Berkeley et Stanford. Les années 1970 virent aussi naître les super-ordinateurs. Seymour Cray (1925-1996) conçut le CRAY-1, qui apparut en mars 1976; il pouvait exécuter 160 millions d'opérations par seconde. Le Cray XMP sortit en 1982. Cray Research (à présent repris par Silicon Graphics) continue à construire des ordinateurs géants.

Histoire de l'Informatique

Il y eut aussi des progrès importants en algorithmique et en théorie de la complexité. En 1971, Steve Cook publia son article fondamental sur la NP-complétude, et, peu après, Richard Karp montra que de nombreux problèmes combinatoires naturels étaient NP-complets.

Whit Diffie et Martin Hellman publièrent un article fondant la théorie de cryptographie à clef publique; le système de cryptage RSA fut inventé par Ronald Rivest, Adi Shamir, et Leonard Adleman.

En 1979, trois étudiants de Caroline du Nord développèrent un serveur de nouvelles distribué qui finalement devint Usenet.

Histoire de l'Informatique

Quelques dates récentes :
1980 : loi « Informatique et Libertés »
1981 : IBM PC, Ms-DOS
1982 : protocole TCP/IP et création du mot « Internet »
1983 : langage C++
1984 : création de la Free Software Fondation, du projet GNU et de la licence GPL, MacIntosh
1985 : norme IEEE 754 sur les nombres flottants, CD-ROM, Windows 1.0
1989 : création du World-Wide-Web par Tim Berners-Lee
1991 : protocole http, premier noyau Linux
1993 : Mosaic : premier navigateur web
1994 : création du World-Wide-Web Consortium W3C
1996 : protocole USB

Histoire de l'Informatique

2014

A l'échelle de la photo de gauche : -

Contient 100 000 fois plus de données.

Débats actuels

Femmes et informatique

https://www.societe-informatique-de-france.fr/revue-de-presse-femmes-info/

Informatique et éthique

CERNA : Commission de réflexion sur l'Ethique de la Recherche en sciences et technologies du Numérique d'Allistene

https://www.allistene.fr/cerna/

- Algorithmes et code publiés : voitures autonomes, Parcoursup
- Protection des données personnelles

Informatique dans l'enseignement

Outils vs Science

Représentations en Informatique

université de BORDEAUX

Informatique = Science du traitement automatique de l'information

On cherche à résoudre un problème P à l'aide d'un traitement automatique.
Il faut :

1) modéliser les données du problème P ➔ *Structure de données* ;
2) trouver une méthode de résolution du problème P, en fonction de la représentation des données ➔ *Algorithmes* ;
3) traduire la structure de données et les algorithmes en fonction de la machine cible ➔ *Programmation*.

Exemples de données à modéliser :

- nombres,
- lettres,
- texte,
- images
- autres …

Nombres et Texte

université de BORDEAUX

Nombres

Représentation binaire

Entiers naturels

- sur 1 octet (8 bits) : 00000000, 00000001, 00000010, … ➔ entre 0 et 2^8 - 1 = 255
- sur 2 octets (16 bits) : entre 0 et 2^{16} - 1 = 65535
- sur 4 octets (32 bits) : entre 0 et 2^{32} - 1 = 4 294 967 295

Entiers relatifs

- le bit de poids fort représente le signe (0 = positif, 1 = négatif)
- complément à 2 :
    - prendre le nombre positif en binaire,
    - inverser chaque bit
    - rajouter 1
- sur 1 octet : -3 = $(11111101)_2$, -2 = $(11111110)_2$, -1 = (11111111), 0 = $(00000000)_2$, 1 = $(00000001)_2$, 2 = $(00000010)_2$, … ➔ entre -2^7 et $2^7 – 1$, soit entre -128 et +127

Nombres réels : on représente le signe, la mantisse (un entier naturel), et une puissance de 10. Exemple -12,5 : -1 x 125 x 10^{-1}

Codage / Décodage d'un nombre en base *b*

Soit *n* un nombre écrit dans une base *b.*

Remarque : les chiffres utilisés pour *n* vont de 0 à b-1 (par exemple de 0 à 9 pour la base 10, de 0 à 1 pour la base 2, de 0 à f pour la base 16).

En base 16, le chiffre 10 est noté a, le chiffre 11 : b, … , le chiffre 15 : f

Si $n = c_k\ c_{k-1}\ c_{k-2} \ldots c_0$, *alors la valeur de n est* $\sum_{i=0 \text{ à } k} c_i \times b^i$

Par exemple, $127 = 7 \times 10^0 + 2 \times 10^1 + 1 \times 10^2 = 7 + 2 \times 10 + 1 \times 100$

Décodage d'un nombre en base *b*

Pour décoder *n*, on applique l'algorithme suivant :

Soit $n = c_k\ c_{k-1}\ c_{k-2} \ldots c_0$ en base *b*

n ← 0

Pour i allant de k à 0 faire :

n ← n * b + c_i

Retourner n

Exemple : pour n = 127 en base 10

n = 0

i = 2 : n ← 0 * 10 + 1 donc n = 1

i = 1 : n ← 1 * 10 + 2 donc n = 12

i = 0 : n ← 12 * 10 + 7 donc n = 127

Décodage d'un nombre en base *b*

Pour décoder *n*, on applique l'algorithme suivant :

Soit $n = c_k\ c_{k-1}\ c_{k-2} \ldots c_0$ en base *b*

n ← 0

Pour i allant de k à 0 par pas de -1 faire :

n ← n * b + c_i

Retourner n

Exemple : pour n = 1001 en base 2

n = 0

i = 3 : n ← 0 * 2 + 1 donc n = 1

i = 2 : n ← 1 * 2 + 0 donc n = 2

i = 1 : n ← 2 * 2 + 0 donc n = 4

i = 0 : n ← 4 * 2 + 1 donc n = 9

Donc $(1001)_2 = (9)_{10}$

Décodage d'un nombre en base *b*

Pour décoder *n*, on applique l'algorithme suivant :

Soit $n = c_k\ c_{k-1}\ c_{k-2} \ldots c_0$ en base *b*

n ← 0

Pour i allant de k à 0 par pas de -1 faire :

n ← n * b + c_i

Retourner n

Exemple : pour n = a3f en base 16

n = 0

i = 2 : n ← 0 * 16 + 10 donc n = 10

i = 1 : n ← 10 * 16 + 3 donc n = 163

i = 0 : n ← 163 * 16 + 15 donc n = 2623

Donc $(a3f)_{16} = (2623)_{10}$

Codage d'un nombre en base *b*

Pour coder *n* en base *b* on applique l'algorithme suivant :

Le symbole % représente le modulo, donc a % b est le reste de la division entière de a par b.

Le symbole // représente la division entière.

i ← 0

Tant que n > 0 faire :

    c[i] ← n % b

    n ← n // b

    i ← i + 1

Retourner c

Codage d'un nombre en base *b*

i ← 0
Tant que n > 0 faire :
    c[i] ← n % b
    n ← n // b
    i ← i + 1
Retourner c
i ← 0

Exemple pour coder n = 127 en base 10

n = 127, i = 0 :

c[0] ← 7, n ← 12, i ← 1

c[1] ← 2, n ← 1, i ← 2

c[2] ← 1, n ← 0, i ← 3

Donc c = {1, 2, 7}

Codage d'un nombre en base *b*

n ← 0
i ← 0
Tant que n > faire :
    c[i] ← n % b
    n ← n // b
    i ← i + 1
Retourner c
n ← 0
i ← 0

Exemple pour coder n = 13 en base 2

n = 13, i = 0 :

c[0] ← 1, n ← 6, i ← 1

c[1] ← 0, n ← 3, i ← 2

c[2] ← 1, n ← 1, i ← 3

c[3] ← 1, n ← 0, i ← 4 Donc c = {1, 1, 0, 1}

On peut vérifier que $1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 1x8 + 1x4 + 1x1 = 13$

Codage d'un nombre en base *b*

i ← 0

Tant que n > 0 faire :

    c[i] ← n % b

    n ← n // b

    i ← i + 1

Retourner c

Exemple pour n = 2623 en base 16

n = 2623, i = 0 :

c[0] ← f, n ← 163, i ← 1

c[1] ← 3, n ← 10, i ← 2

c[2] ← a, n ← 0, i ← 3

Donc c = {a, 3, f}

On peut vérifier que $10 \times 16^2 + 3 \times 16^1 + 15 \times 16^0 = 2623$

Caractères

Code ASCII (American Standard Code for Information Interchange)
Correspondance entre un caractère et un entier compris entre 0 et 127
95 caractères affichables et 33 caractères de contrôle : retour arrière, saut de page …

caractères affichables (dont l'espace !) :

```
 !"#$%&'()*+,-./
0123456789
:;<=>? @
ABCDEFGHIJKLMNOPQRSTUVWXYZ
[\]^_`
abcdefghijklmnopqrstuvwxyz
{|}~
```

Remarques

• La valeur du chiffre codé `c` est `c-48` (48 = code de 0).
• Pour passer d'une majuscule à la minuscule correspondante, il faut rajouter 32, soit la différence entre les valeurs associées à '`a`' (97) et '`A`'(65).
• Pas de caractères accentués.

Passage à Unicode : normes UTF8, UTF16 …

Code ASCII

Hex	Dec	Char		Hex	Dec	Char	Hex	Dec	Char	Hex	Dec	Char
0x00	0	NULL	null	0x20	32	Space	0x40	64	@	0x60	96	`
0x01	1	SOH	Start of heading	0x21	33	!	0x41	65	A	0x61	97	a
0x02	2	STX	Start of text	0x22	34	"	0x42	66	B	0x62	98	b
0x03	3	ETX	End of text	0x23	35	#	0x43	67	C	0x63	99	c
0x04	4	EOT	End of transmission	0x24	36	$	0x44	68	D	0x64	100	d
0x05	5	ENQ	Enquiry	0x25	37	%	0x45	69	E	0x65	101	e
0x06	6	ACK	Acknowledge	0x26	38	&	0x46	70	F	0x66	102	f
0x07	7	BELL	Bell	0x27	39	'	0x47	71	G	0x67	103	g
0x08	8	BS	Backspace	0x28	40	(	0x48	72	H	0x68	104	h
0x09	9	TAB	Horizontal tab	0x29	41	)	0x49	73	I	0x69	105	i
0x0A	10	LF	New line	0x2A	42	*	0x4A	74	J	0x6A	106	j
0x0B	11	VT	Vertical tab	0x2B	43	+	0x4B	75	K	0x6B	107	k
0x0C	12	FF	Form Feed	0x2C	44	,	0x4C	76	L	0x6C	108	l
0x0D	13	CR	Carriage return	0x2D	45	-	0x4D	77	M	0x6D	109	m
0x0E	14	SO	Shift out	0x2E	46	.	0x4E	78	N	0x6E	110	n
0x0F	15	SI	Shift in	0x2F	47	/	0x4F	79	O	0x6F	111	o
0x10	16	DLE	Data link escape	0x30	48	0	0x50	80	P	0x70	112	p
0x11	17	DC1	Device control 1	0x31	49	1	0x51	81	Q	0x71	113	q
0x12	18	DC2	Device control 2	0x32	50	2	0x52	82	R	0x72	114	r
0x13	19	DC3	Device control 3	0x33	51	3	0x53	83	S	0x73	115	s
0x14	20	DC4	Device control 4	0x34	52	4	0x54	84	T	0x74	116	t
0x15	21	NAK	Negative ack	0x35	53	5	0x55	85	U	0x75	117	u
0x16	22	SYN	Synchronous idle	0x36	54	6	0x56	86	V	0x76	118	v
0x17	23	ETB	End transmission block	0x37	55	7	0x57	87	W	0x77	119	w
0x18	24	CAN	Cancel	0x38	56	8	0x58	88	X	0x78	120	x
0x19	25	EM	End of medium	0x39	57	9	0x59	89	Y	0x79	121	y
0x1A	26	SUB	Substitute	0x3A	58	:	0x5A	90	Z	0x7A	122	z
0x1B	27	FSC	Escape	0x3B	59	;	0x5B	91	[	0x7B	123	{
0x1C	28	FS	File separator	0x3C	60	<	0x5C	92	\	0x7C	124	|
0x1D	29	GS	Group separator	0x3D	61	=	0x5D	93	]	0x7D	125	}
0x1E	30	RS	Record separator	0x3E	62	>	0x5E	94	^	0x7E	126	~
0x1F	31	US	Unit separator	0x3F	63	?	0x5F	95	_	0x7F	127	DEL

Texte

Texte simple
suite de caractères terminée par un caractère de contrôle (le plus souvent 0)

Texte formaté : exemple de HTML (HyperText Meta-Language)
Exemple de page HTML

Code

```
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8 »>
<title>Mon Titre</title>
</head>
<body>
<h1>Titre 1</h1><br>
<h3>Titre 3</h3>
<p><b>ABC <i>DEF </i></b><i>IJK </i>LMN</p>
<h3 align="center">Titre 3 centré<br></h3>
<p><a href="http://www.labri.fr/">Exemple de lien</a><br>
<a name="mon_ancre">Exemple d'ancre</a><br>
<img alt="" src="ours.jpg" height="137" width="200"><br>
</p>
<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<p><a href="#mon_ancre">Lien vers l'ancre</a><br></p>
</body>
</html>
```