PLAN DE FORMATION · 2026

Cursus Computer Science en trois ans

Un parcours de type licence (BSc) en informatique : de la première ligne de Python à l'algorithmique, au génie logiciel et à l'apprentissage automatique, avec des cours universitaires de Coursera et des manuels gratuits.

Année 1	Année 2	Année 3
Fondations	Programmation et algorithmique	Génie logiciel et ouvertures
384 h	**379 h**	**438 h**

1201 h	6	68	11	14
de cours en tronc commun	semestres	cours	universités	manuels de référence

Données relevées sur coursera.org en novembre 2026 · Disponibilité des manuels vérifiée à la même date

Sommaire

1. Présentation du cursus
2. Vue d'ensemble et charge de travail
3. Année 1 · Fondations (S1, S2)
4. Année 2 · Programmation et algorithmique (S3, S4)
5. Année 3 · Génie logiciel et ouvertures (S5, S6)
6. Options
7. Bibliographie
8. Méthode et avertissements

1. Présentation du cursus

Ce cursus reproduit, en autodidacte, la structure d'une licence d'informatique : trois années, cinq domaines et une montée en abstraction continue. Il s'adresse à un débutant motivé qui a le niveau de mathématiques de fin de lycée et peut consacrer une dizaine d'heures par semaine à ses études.

LES CINQ DOMAINES

- **Programmation.** Python, C, Java, Go, JavaScript.
- **Mathématiques.** preuves, combinatoire, graphes, arithmétique, algèbre linéaire, statistique.
- **Algorithmique.** structures de données, conception et analyse d'algorithmes, complexité.
- **Génie logiciel.** conception, architecture, cycle de vie agile, tests, sécurité.
- **Ouvertures.** concurrence, web et apprentissage automatique.

LES PRINCIPES QUI FIXENT L'ORDRE

- **Programmer d'abord.** Python (S1) donne l'outil pour tout le reste, y compris les mathématiques discrètes, qui s'appuient sur de petits programmes.
- **Les mathématiques avant l'algorithmique.** Preuves, combinatoire et graphes (S2) précèdent l'analyse des algorithmes de Stanford (S4).
- **Du bas niveau vers l'abstraction.** Le C (S3) montre la mémoire et la machine ; Java (S3, S4) apporte les objets et les structures de données ; le génie logiciel (S5) organise le tout.
- **Le métier avant les ouvertures.** Conception, cycle de vie et tests (S5) viennent avant la sécurité, la concurrence, le web et l'apprentissage automatique (S6).
- **Un projet par semestre.** Chaque semestre se termine par un capstone ou un projet qui réunit ses compétences.

LANGAGES ET OUTILS

Python (S1, S2, S6), C (S3), Java (S3 à S5), Go et JavaScript (S6). Rice utilise CodeSkulptor dans le navigateur ; les autres cours demandent d'installer un compilateur C, un JDK Java et Go.

2. Vue d'ensemble et charge de travail

ANNÉE	SEM.	CONTENU	HEURES
Année 1 Fondations **384 h**	**S1**	**Programmer et penser algorithmiquement** Python for Everybody Specialization (79 h) Fundamentals of Computing Specialization (98 h)	177 h
	S2	**Mathématiques pour l'informatique** Introduction to Discrete Mathematics for Computer Science (116 h) Linear Algebra from Elementary to Advanced (34 h) Statistics with Python (57 h)	207 h
Année 2 Programmation et algorithmique **379 h**	**S3**	**C et Java** Introductory C Programming Specialization (84 h) Java Programming and Software Engineering Fundamentals (84 h)	168 h
	S4	**Structures de données et algorithmes** Object Oriented Java Programming: Data Structures and Beyond (154 h) Algorithms Specialization (57 h)	211 h
Année 3 Génie logiciel et ouvertures **438 h**	**S5**	**Génie logiciel** Software Design and Architecture Specialization (52 h) Software Development Lifecycle Specialization (63 h) Software Testing and Automation Specialization (95 h)	210 h
	S6	**Sécurité, concurrence, web et IA** Secure Software Design Specialization (60 h) Programming with Google Go (34 h) Web Design for Everybody: Basics of Web Development & Coding Specialization (78 h) Machine Learning: Theory and Hands-on Practice with Python (56 h)	228 h
Total du tronc commun			**1 201 h**
Options (facultatives)			+ 788 h

HEURES DE COURS PAR SEMESTRE



RYTHME CONSEILLÉ

Sur 40 semaines de travail par an, le cursus demande environ **10 h par semaine en année 1**, **9 h en année 2** et **11 h en année 3**. Les durées sont celles annoncées par Coursera ; prévoir en plus 20 à 30 % de temps pour les lectures et les projets.

ANNÉE 1 · 384 HEURES

Fondations

Programmer en Python, penser de façon algorithmique, et acquérir les mathématiques de l'informatique

S1

ANNÉE 1 · 177 HEURES · ≈ 9 H PAR SEMAINE SUR 20 SEMAINES

Programmer et penser algorithmiquement

> Apprendre à programmer en Python, du premier script aux programmes interactifs, puis aborder la récursivité, la recherche et les graphes.

COMPÉTENCES VISÉES

- Variables, boucles, fonctions, structures de données Python
- Fichiers, web, API et bases SQLite
- Programmation événementielle et orientée objet
- Tests, probabilités et combinatoire appliquées aux programmes
- Recherche, récursivité, graphes et complexité en introduction

PROJET DE FIN DE SEMESTRE

L'examen final de Fundamentals of Computing, puis un jeu de stratégie en Python avec une IA minimax et une suite de tests.

POINTS D'ATTENTION

- Suivre Fundamentals of Computing en parallèle de la seconde moitié de Python for Everybody.
- Rice utilise CodeSkulptor, un environnement Python dans le navigateur.

PROGRAMME

1

SPÉCIALISATION · 5 COURS · 79 H

Python for Everybody Specialization

University of Michigan | Débutant | Python, SQL, SQLite

Prérequis annoncés : No prerequisites; designed as an on-ramp to programming

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Programming for Everybody (Getting Started with Python)	Débutant	17 h	4,8 (233 775)
2	Python Data Structures	Débutant	20 h	4,9 (97 365)
3	Using Python to Access Web Data	Débutant	19 h	4,8 (44 591)
4	Using Databases with Python	Débutant	14 h	4,8 (21 659)
5	Capstone: Retrieving, Processing, and Visualizing Data with Python	Débutant	9 h	4,7 (14 483)

2

SPÉCIALISATION · 7 COURS · 98 H

Fundamentals of Computing Specialization

Rice University | Débutant | Python

Prérequis annoncés : High school mathematics; no programming required

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	An Introduction to Interactive Programming in Python (Part 1)	—	19 h	4,8 (3 322)
2	An Introduction to Interactive Programming in Python (Part 2)	Débutant	16 h	4,9 (1 214)
3	Principles of Computing (Part 1)	Intermédiaire	19 h	4,7 (584)
4	Principles of Computing (Part 2)	Intermédiaire	16 h	4,8 (365)
5	Algorithmic Thinking (Part 1)	Intermédiaire	14 h	4,6 (385)
6	Algorithmic Thinking (Part 2)	Intermédiaire	13 h	4,7 (220)
7	The Fundamentals of Computing Capstone Exam	Intermédiaire	1 h	4,8 (116)

LECTURES COMPLÉMENTAIRES

Python for Everybody

Charles R. Severance

Manuel officiel de la spécialisation de Michigan.

Gratuit, CC BY 4.0 · www.py4e.com/book

Think Python, 3rd edition

Allen B. Downey

Une approche plus rigoureuse des fonctions, de la récursivité et des classes.

Gratuit, CC BY-NC-SA 4.0 · allendowney.github.io/ThinkPython

S2

ANNÉE 1 · 207 HEURES · ≈ 10 H PAR SEMAINE SUR 20 SEMAINES

Mathématiques pour l'informatique

> Savoir prouver, compter et raisonner sur des graphes, maîtriser l'algèbre linéaire de base et les statistiques nécessaires à l'informatique moderne.

COMPÉTENCES VISÉES

- Preuves : récurrence, absurde, invariants
- Combinatoire, probabilités discrètes, théorie des graphes
- Arithmétique modulaire et cryptographie RSA
- Systèmes linéaires, matrices, valeurs propres
- Statistique descriptive et inférentielle en Python

PROJET DE FIN DE SEMESTRE

Le projet Delivery Problem de UCSD (voyageur de commerce), puis une petite implémentation de RSA avec ses tests.

POINTS D'ATTENTION

- Discrete Mathematics demande des bases de Python : le S1 les fournit.
- Linear Algebra ne demande pas de calcul différentiel.

PROGRAMME

1

SPÉCIALISATION · 5 COURS · 116 H

Introduction to Discrete Mathematics for Computer Science

University of California San Diego | Débutant | Python

Prérequis annoncés : Basic math (fractions, squares), basic Python (functions, loops, recursion)

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Mathematical Thinking in Computer Science	Débutant	42 h	4,4 (2 289)
2	Combinatorics and Probability	Débutant	23 h	4,6 (874)
3	Introduction to Graph Theory	Débutant	21 h	4,5 (1 073)
4	Number Theory and Cryptography	Débutant	17 h	4,5 (610)
5	Delivery Problem	Débutant	13 h	4,7 (376)

2

SPÉCIALISATION · 3 COURS · 34 H

Linear Algebra from Elementary to Advanced

Johns Hopkins University | Débutant

Prérequis annoncés : Basic algebra; no calculus needed

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Linear Algebra: Linear Systems and Matrix Equations	Débutant	10 h	4,7 (183)
2	Linear Algebra: Matrix Algebra, Determinants, & Eigenvectors	—	15 h	4,8 (79)
3	Linear Algebra: Orthogonality and Diagonalization	Intermédiaire	9 h	4,8 (56)

3

SPÉCIALISATION · 3 COURS · 57 H

Statistics with Python

University of Michigan | Débutant | Python, Jupyter, Matplotlib, NumPy, pandas, Seaborn, Statsmodels

Prérequis annoncés : High school algebra; basic Python recommended

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Understanding and Visualizing Data with Python	Débutant	20 h	4,7 (2 741)
2	Inferential Statistical Analysis with Python	Intermédiaire	22 h	4,6 (938)
3	Fitting Statistical Models to Data with Python	Intermédiaire	15 h	4,4 (718)

LECTURES COMPLÉMENTAIRES

Mathematics for Computer Science

E. Lehman, F. T. Leighton, A. R. Meyer (MIT)

La référence pour les preuves, la combinatoire et les graphes.

Gratuit, CC BY-SA · ocw.mit.edu/courses/6-1200j-mathematics-for-computer-science-spring-2024/pages/readings

Linear Algebra

Jim Hefferon

Manuel complet avec exercices corrigés.

Gratuit · hefferon.net/linearalgebra

OpenIntro Statistics

D. Diez, M. Çetinkaya-Rundel, C. Barr

Pour Statistics with Python.

PDF gratuit · www.openintro.org/book/os

ANNÉE 2 · 379 HEURES

Programmation et algorithmique

Programmer près de la machine en C, programmer objet en Java, puis concevoir des structures de données et des algorithmes efficaces

S3

ANNÉE 2 · 168 HEURES · ≈ 8 H PAR SEMAINE SUR 20 SEMAINES

C et Java

> Comprendre la mémoire, les pointeurs et la compilation avec le C, puis la programmation orientée objet avec Java.

COMPÉTENCES VISÉES

- Algorithmes en sept étapes, écriture, test et débogage en C
- Pointeurs, tableaux, récursivité et gestion de la mémoire
- Interaction avec le système : fichiers, arguments, mémoire dynamique
- Java, classes, listes et principes de conception
- Pages web simples en HTML, CSS et JavaScript

PROJET DE FIN DE SEMESTRE

Le système de recommandation du dernier cours de Duke, puis un mini-shell Unix en C avec redirections et tubes.

POINTS D'ATTENTION

- Commencer par le C : la gestion de la mémoire éclaire ensuite le fonctionnement de Java.
- Le premier cours de Duke porte sur le web (HTML, CSS, JavaScript) avant Java.

PROGRAMME

1

SPÉCIALISATION · 4 COURS · 84 H

Introductory C Programming Specialization

Duke University · Débutant · C

Prérequis annoncés : No prior programming experience required

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Programming Fundamentals	Débutant	19 h	4,7 (7 054)
2	Writing, Running, and Fixing Code in C	Débutant	20 h	4,3 (837)
3	Pointers, Arrays, and Recursion	Débutant	21 h	4,3 (352)
4	Interacting with the System and Managing Memory	Débutant	24 h	4,2 (182)

2

SPÉCIALISATION · 5 COURS · 84 H

Java Programming and Software Engineering Fundamentals

Duke University | Débutant | Java, JavaScript, HTML, CSS, BlueJ, CodePen

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Programming Foundations with JavaScript, HTML and CSS	Débutant	33 h	4,6 (14 960)
2	Java Programming: Solving Problems with Software	Débutant	18 h	4,6 (8 095)
3	Java Programming: Arrays, Lists, and Structured Data	Débutant	15 h	4,6 (3 223)
4	Java Programming: Principles of Software Design	Débutant	13 h	4,6 (1 505)
5	Java Programming: Build a Recommendation System	Intermédiaire	5 h	4,7 (661)

LECTURES COMPLÉMENTAIRES

Modern C

Jens Gustedt

Le C moderne, de la syntaxe à la gestion fine de la mémoire.

PDF gratuit, CC BY-NC-ND 4.0 · gustedt.gitlabpages.inria.fr/modern-c

Introduction to Programming Using Java

David J. Eck

Manuel complet de Java 17, en parallèle de Duke.

Gratuit, CC BY-NC-SA 4.0 · math.hws.edu/javanotes

S4

ANNÉE 2 · 211 HEURES · ≈ 11 H PAR SEMAINE SUR 20 SEMAINES

Structures de données et algorithmes

> Choisir et implémenter les bonnes structures de données, puis concevoir et analyser des algorithmes efficaces.

COMPÉTENCES VISÉES

- Programmation orientée objet avancée en Java
- Listes, arbres, tables de hachage, graphes et leurs performances
- Diviser pour régner, tri, recherche et algorithmes randomisés
- Parcours de graphes, plus courts chemins, algorithmes gloutons
- Programmation dynamique et NP-complétude

PROJET DE FIN DE SEMESTRE

Le capstone de UCSD (analyse d'un réseau social en Java), puis un banc d'essai comparant cinq algorithmes de tri et de plus courts chemins.

POINTS D'ATTENTION

- Stanford Algorithms suppose la lecture de preuves : le S2 y prépare.
- Pour un entraînement intensif aux entretiens, voir l'option Data Structures and Algorithms (UCSD).

PROGRAMME

1

SPÉCIALISATION · 5 COURS · 154 H

Object Oriented Java Programming: Data Structures and Beyond

University of California San Diego | Intermédiaire | Java

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Object Oriented Programming in Java	Intermédiaire	40 h	4,7 (5 806)
2	Data Structures and Performance	Intermédiaire	42 h	4,8 (2 251)
3	Advanced Data Structures in Java	Intermédiaire	30 h	4,8 (1 243)
4	Mastering the Software Engineering Interview	Intermédiaire	22 h	4,6 (555)
5	Capstone: Analyzing (Social) Network Data	Débutant	20 h	4,7 (101)

2

SPÉCIALISATION · 4 COURS · 57 H

Algorithms Specialization

Stanford University (Stanford Online) | Intermédiaire | C, Java, Python

Prérequis annoncés : Programming in at least one language (C, Java or Python); familiarity with proofs (induction, contradiction); discrete probability

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Divide and Conquer, Sorting and Searching, and Randomized Algorithms	Intermédiaire	15 h	4,8 (5 342)
2	Graph Search, Shortest Paths, and Data Structures	Intermédiaire	14 h	4,8 (2 002)
3	Greedy Algorithms, Minimum Spanning Trees, and Dynamic Programming	Intermédiaire	15 h	4,8 (1 279)
4	Shortest Paths Revisited, NP-Complete Problems and What To Do About Them	Intermédiaire	13 h	4,8 (832)

LECTURES COMPLÉMENTAIRES

Open Data Structures

Pat Morin

Structures de données avec leur code en Java.

Gratuit, CC BY 2.5 · opendatastructures.org

Algorithms

Jeff Erickson

Accompagne Stanford Algorithms : récursivité, programmation dynamique, graphes.

Gratuit, CC BY 4.0 · jeffe.cs.illinois.edu/teaching/algorithms

ANNÉE 3 · 438 HEURES

Génie logiciel et ouvertures

Concevoir, organiser et tester du logiciel professionnel, puis la sécurité, la concurrence, le web et l'apprentissage automatique

S5

ANNÉE 3 · 210 HEURES · ≈ 10 H PAR SEMAINE SUR 20 SEMAINES

Génie logiciel

> Passer du programme qui marche au logiciel maintenable : conception, architecture, cycle de vie agile et tests automatisés.

COMPÉTENCES VISÉES

- Conception orientée objet et patrons de conception
- Architecture logicielle et architectures orientées services
- Processus logiciels, agilité et lean
- Tests boîte noire et boîte blanche, analyse automatique
- Tests web et mobiles

PROJET DE FIN DE SEMESTRE

Un projet logiciel mené en binôme sur deux sprints : conception avec patrons, carnet de produit, tests automatisés et intégration continue.

POINTS D'ATTENTION

- Software Design and Architecture utilise Java et Android Studio.
- Deux cours de Software Testing sont notés sous 4 : compléter avec le livre de Google.

PROGRAMME

1

SPÉCIALISATION · 4 COURS · 52 H

Software Design and Architecture Specialization

University of Alberta | Débutant | Java, Android Studio

Prérequis annoncés : Familiarity with object-oriented programming; basic Java

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Object-Oriented Design	Intermédiaire	17 h	4,7 (2 434)
2	Design Patterns	Intermédiaire	15 h	4,7 (1 382)
3	Software Architecture	Intermédiaire	10 h	4,5 (958)
4	Service-Oriented Architecture	Intermédiaire	10 h	4,5 (599)

2

SPÉCIALISATION · 4 COURS · 63 H

Software Development Lifecycle Specialization

University of Minnesota | Intermédiaire

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Software Development Processes and Methodologies	Débutant	21 h	4,7 (2 135)
2	Agile Software Development	Intermédiaire	13 h	4,7 (7 485)
3	Lean Software Development	Débutant	11 h	4,6 (433)
4	Engineering Practices for Building Quality Software	Intermédiaire	18 h	4,5 (476)

3

SPÉCIALISATION · 4 COURS · 95 H

Software Testing and Automation Specialization

University of Minnesota | Intermédiaire | Java, JUnit, Cucumber, Gherkin, Selenium, OWASP

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Introduction to Software Testing	Intermédiaire	30 h	4,3 (839)
2	Black-box and White-box Testing	Intermédiaire	26 h	3,8 (117)
3	Introduction to Automated Analysis	Intermédiaire	19 h	4,2 (87)
4	Web and Mobile Testing with Selenium	Intermédiaire	20 h	3,7 (111)

LECTURES COMPLÉMENTAIRES

Software Engineering at Google

T. Winters, T. Manshreck, H. Wright (eds.)

Les pratiques d'ingénierie logicielle à grande échelle : tests, revue de code, dépendances.

Gratuit en ligne, CC BY-NC-ND 4.0 · abseil.io/resources/swe-book

The Scrum Guide

K. Schwaber, J. Sutherland

Le texte de référence de Scrum, en quelques pages.

Gratuit · scrumguides.org

S6

ANNÉE 3 · 228 HEURES · ≈ 11 H PAR SEMAINE SUR 20 SEMAINES

Sécurité, concurrence, web et IA

> Élargir le profil : concevoir du logiciel sûr, programmer la concurrence en Go, construire pour le web et utiliser l'apprentissage automatique.

COMPÉTENCES VISÉES

- Menaces, abstraction et méthodes de conception sécurisée
- Fonctions, méthodes et concurrence en Go
- HTML, CSS, JavaScript et design responsive
- Apprentissage supervisé, non supervisé et introduction au deep learning

PROJET DE FIN DE SEMESTRE

Le projet de fin de cursus : une application web sécurisée, servie par un service concurrent en Go, avec une fonction de recommandation par apprentissage automatique.

POINTS D'ATTENTION

- Programming with Go demande une expérience de la programmation objet : Java (S3, S4) la fournit.
- Machine Learning: Theory and Hands-on Practice compte peu d'avis ; s'appuyer sur ISLP.

PROGRAMME

1

SPÉCIALISATION · 4 COURS · 60 H

Secure Software Design Specialization

University of Colorado System · Débutant · Java, NetBeans, JUnit, IBM Rhapsody

Prérequis annoncés : Knowing how to code in some language; workplace software development experience helpful

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Software Design as an Element of the Software Development Lifecycle	—	11 h	4,6 (253)
2	Software Design as an Abstraction	Intermédiaire	16 h	4,7 (135)
3	Software Design Methods and Tools	—	14 h	—
4	Software Design Threats and Mitigations	Intermédiaire	19 h	4,6 (94)

2

SPÉCIALISATION · 3 COURS · 34 H

Programming with Google Go

University of California, Irvine · Intermédiaire · Go

Prérequis annoncés : Previous programming experience, ideally object-oriented (C++ or Python)

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Getting Started with Go	Intermédiaire	10 h	4,6 (2 484)
2	Functions, Methods, and Interfaces in Go	Intermédiaire	11 h	4,6 (1 017)
3	Concurrency in Go	Intermédiaire	13 h	4,7 (809)

3

SPÉCIALISATION · 5 COURS · 78 H

Web Design for Everybody: Basics of Web Development & Coding Specialization

University of Michigan | Débutant | HTML, CSS, JavaScript, Bootstrap

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Introduction to HTML5	Débutant	12 h	4,7 (27 930)
2	Introduction to CSS3	Débutant	17 h	4,8 (9 671)
3	Interactivity with JavaScript	Débutant	10 h	4,7 (7 382)
4	Advanced Styling with Responsive Design	Débutant	11 h	4,7 (4 617)
5	Web Design for Everybody Capstone	Débutant	28 h	4,8 (1 790)

4

SPÉCIALISATION · 3 COURS · 56 H

Machine Learning: Theory and Hands-on Practice with Python

University of Colorado Boulder | Intermédiaire | Python

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Introduction to Machine Learning: Supervised Learning	Intermédiaire	23 h	4,5 (67)
2	Introduction to Machine Learning: Unsupervised Learning	Intermédiaire	15 h	4,6 (24)
3	Introduction to Deep Learning	Intermédiaire	18 h	4,5 (24)

LECTURES COMPLÉMENTAIRES

A Tour of Go

The Go Authors

Le tutoriel officiel de Go, concurrence comprise.

Tutoriel officiel gratuit · go.dev/tour

Eloquent JavaScript, 4th edition

Marijn Haverbeke

Pour aller au-delà des bases JavaScript de Web Design for Everybody.

Gratuit, CC BY-NC 3.0 · eloquentjavascript.net

An Introduction to Statistical Learning (Python)

G. James, D. Witten, T. Hastie, R. Tibshirani, J. Taylor

Le manuel de référence pour Machine Learning (CU Boulder).

PDF gratuit · www.statlearning.com

6. Options

Ces spécialisations ne sont pas comptées dans le total. Elles conviennent en fin d'année 3 ou après le cursus, selon l'orientation choisie.

Un entraînement intensif aux structures de données et aux algorithmes, par exercices de programmation notés automatiquement : idéal pour préparer les entretiens techniques.

SPÉCIALISATION · 6 COURS · 182 H

Data Structures and Algorithms Specialization

University of California San Diego | Intermédiaire | Python, Java, C++, JavaScript, Rust

Prérequis annoncés : Basic programming (loops, arrays, stacks, recursion); discrete math (induction, contradiction)

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Algorithmic Toolbox	Intermédiaire	41 h	4,6 (12 588)
2	Data Structures	Intermédiaire	23 h	4,6 (5 580)
3	Algorithms on Graphs	Intermédiaire	55 h	4,7 (2 272)
4	Algorithms on Strings	Intermédiaire	19 h	4,5 (1 091)
5	Advanced Algorithms and Complexity	Avancé	27 h	4,6 (703)
6	Genome Assembly Programming Challenge	Avancé	17 h	4,5 (355)

L'algorithmique avancée de CU Boulder, jusqu'à la programmation linéaire et aux algorithmes quantiques. Éligible à des crédits universitaires.

SPÉCIALISATION · 5 COURS · 199 H

Foundations of Data Structures and Algorithms Specialization

University of Colorado Boulder | Avancé | Python | crédits universitaires

Prérequis annoncés : Some background in calculus, probability theory and Python programming

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Algorithms for Searching, Sorting, and Indexing	Intermédiaire	36 h	4,7 (544)
2	Trees and Graphs: Basics	Avancé	34 h	4,6 (194)
3	Dynamic Programming, Greedy Algorithms	Avancé	38 h	4,7 (282)
4	Approximation Algorithms and Linear Programming	Avancé	46 h	4,8 (67)
5	Advanced Data Structures, RSA and Quantum Algorithms	Avancé	45 h	4,6 (70)

La programmation système moderne et sûre en Rust, avec ingénierie des données et DevOps.

SPÉCIALISATION · 5 COURS · 161 H

Rust Programming Specialization

Duke University | Débutant | Rust, Python, Cargo, Docker, Jenkins, Prometheus, Hugging Face

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Rust Fundamentals	Débutant	40 h	4,0 (255)
2	Data Engineering with Rust	Intermédiaire	64 h	3,5 (72)
3	Rust for DevOps	Intermédiaire	19 h	4,1 (19)
4	Python and Rust with Linux Command Line Tools	Intermédiaire	21 h	5,0 (16)
5	Rust for Large Language Model Operations (LLMOps)	Intermédiaire	17 h	4,3 (15)

La programmation fonctionnelle et parallèle, jusqu'à Spark, avec l'EPFL.

SPÉCIALISATION · 5 COURS · 184 H

Functional Programming in Scala Specialization

École Polytechnique Fédérale de Lausanne | Intermédiaire | Scala, Apache Spark, Apache Hadoop

Prérequis annoncés : At least one year of programming experience

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Functional Programming Principles in Scala	Intermédiaire	56 h	4,8 (8 271)
2	Functional Program Design in Scala	Intermédiaire	35 h	4,5 (3 137)
3	Parallel programming	Intermédiaire	33 h	4,4 (1 844)
4	Big Data Analysis with Scala and Spark	Intermédiaire	28 h	4,6 (2 600)
5	Functional Programming in Scala Capstone	—	32 h	4,4 (547)

Le développement web côté serveur en Python, en prolongement de Web Design for Everybody.

SPÉCIALISATION · 4 COURS · 62 H

Django for Everybody Specialization

University of Michigan | Intermédiaire | Django, Python, HTML, CSS, JavaScript, jQuery, SQL, Linux

Prérequis annoncés : Python for Everybody specialization or equivalent (encouraged)

#	COURS	NIVEAU	DURÉE	NOTE (AVIS)
1	Web Application Technologies and Django	Intermédiaire	12 h	4,7 (1 928)
2	Building Web Applications in Django	Intermédiaire	14 h	4,7 (787)
3	Django Features and Libraries	Intermédiaire	17 h	4,7 (401)
4	Developing and Deploying a Complete Django Web Application	Intermédiaire	19 h	4,8 (295)

7. Bibliographie

Les 14 ouvrages sont consultables gratuitement en ligne, selon les conditions indiquées.

1. **Charles R. Severance**. *Python for Everybody*. Gratuit, CC BY 4.0. www.py4e.com/book
2. **Allen B. Downey**. *Think Python, 3rd edition*. Gratuit, CC BY-NC-SA 4.0. allendowney.github.io/ThinkPython
3. **E. Lehman, F. T. Leighton, A. R. Meyer (MIT)**. *Mathematics for Computer Science*. Gratuit, CC BY-SA. ocw.mit.edu/courses/6-1200j-mathematics-for-computer-science-spring-2024/pages/readings
4. **Jim Hefferon**. *Linear Algebra*. Gratuit. hefferon.net/linearalgebra
5. **D. Diez, M. Çetinkaya-Rundel, C. Barr**. *OpenIntro Statistics*. PDF gratuit. www.openintro.org/book/os
6. **Jens Gustedt**. *Modern C*. PDF gratuit, CC BY-NC-ND 4.0. gustedt.gitlabpages.inria.fr/modern-c
7. **David J. Eck**. *Introduction to Programming Using Java*. Gratuit, CC BY-NC-SA 4.0. math.hws.edu/javanotes
8. **Pat Morin**. *Open Data Structures*. Gratuit, CC BY 2.5. opendatastructures.org
9. **Jeff Erickson**. *Algorithms*. Gratuit, CC BY 4.0. jeffe.cs.illinois.edu/teaching/algorithms
10. **T. Winters, T. Manshreck, H. Wright (eds.)**. *Software Engineering at Google*. Gratuit en ligne, CC BY-NC-ND 4.0. abseil.io/resources/swe-book
11. **K. Schwaber, J. Sutherland**. *The Scrum Guide*. Gratuit. scrumguides.org
12. **The Go Authors**. *A Tour of Go*. Tutoriel officiel gratuit. go.dev/tour
13. **Marijn Haverbeke**. *Eloquent JavaScript, 4th edition*. Gratuit, CC BY-NC 3.0. eloquentjavascript.net
14. **G. James, D. Witten, T. Hastie, R. Tibshirani, J. Taylor**. *An Introduction to Statistical Learning (Python)*. PDF gratuit. www.statlearning.com

8. Méthode et avertissements

- **Source des données.** Les cours, durées, niveaux et notes viennent des pages coursera.org relevées en novembre 2026, et des fichiers « Computer Science Specializations » et « Data Science Specializations Coursera ».
- **Calcul des heures.** Les volumes additionnent les durées annoncées pour chaque cours. Les durées d'en-tête de Coursera, souvent différentes, ne sont pas utilisées.
- **Ce que le cursus ne couvre pas.** Coursera n'offre pas de spécialisation universitaire couvrant les systèmes d'exploitation, les réseaux, les bases de données ou la compilation ; les cours de C, de Go et de Software Testing en abordent une partie.
- **Évolution du catalogue.** Coursera modifie régulièrement ses cours, durées et notes. Vérifier chaque page avant de s'inscrire.
- **Manuels.** La disponibilité gratuite a été vérifiée sur les sites des auteurs et des éditeurs ; les licences diffèrent.