Programme de formation Informatique logicielle (ISCL)

Mathématiques 1 (MAT1) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Mathématiques 1 (MAT1)	150		6*							
Mathématiques discrètes (MAD)	90		4*							
Tutorat MAT1 (TMAT1)	fac.									
Unité préparatoire de mathématiques (UPMath)	fac.	12								
Programmation (PRG) 9 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Programmation 1 (PRG1)	270		12*							
Tutorat PRG1 (TPRG1)	fac.									
Unité préparatoire d'informatique (UPInfo)	fac.	12								
Systèmes logiques et science des données (SLD) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Introduction à la science des données (ISD)	120		4*							
Systèmes logiques (SYL)	120		4*							
Unité préparatoire de systèmes logiques (UPSysLog)	fac.	12								
Communication 1 (COM1) 4 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Anglais 1 (ENG1)	60			3*						
Expression et communication (EXP)	60		2							
Algorithmes et structures de données (ASD) 6 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Algorithmes et structures de données (ASD)	180			8*						
Tutorat ASD (TASD)	fac.									
Architecture et sécurité informatique (ASI) 7 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Architectures des ordinateurs (ARO)	105			4*						
Introduction à la sécurité de l'information (ISI)	105			4*						
Mathématiques 2 (MAT2) 5 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Mathématiques 2 (MAT2)	150			6*						
Tutorat MAT2 (TMAT2)	fac.									
Programmation et réseaux informatiques (PRI) 7 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Programmation 2 (PRG2)	105			4*						
Réseaux informatiques (RXI)	105			4*						
Tutorat PRG2 (TPRG2)	fac.									
Communication 2 (COM2) 3 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Anglais 2 (ENG2)	60				16					
Design thinking and sprint (DTS)	30				8					
Projet d'informatique (PIN) 3 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Projet d'informatique (PIN)	90				16					
Bases de données et applications internet (BDA) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Bases de données relationnelles (BDR)	150					6*				
Développement d'applications internet (DAI)	90					4*				
Mathématiques 3 (MAT3) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Mathématiques 3 (MAT3)	120					4				
Probabilités et statistiques (PST)	120					4				
Programmation orienté objet (POO) 5 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Programmation orientée objet (POO)	150					6*				

Systèmes d'exploitation et concurrence (SEC) 7 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Programmation concurrente (PCO)	105					4*				
Systèmes d'exploitation (SYE)	105					4*				
Conception orientée objet (COO) 6 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Modèles de conception réutilisables (MCR)	90						3			
Programmation orientée objet avancée (POA)	90						3*			
Développement logiciel et responsabilités (DLR) 5 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Ethique et aspects légaux (EAL)	45						2			
Processus de développement en ingénierie logicielle (PDL)	105						4			
Graphes et réseaux de neurones (GRN) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Apprentissage par réseaux de neurones artificiels (ARN)	90						4*			
Graphes et réseaux (GRE)	150						6*			
Technologies Cloud et Web (TCW) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Cloud Computing (CLD)	90						4			
Technologies web (WEB)	150						6*			
Projet de groupe (PDG) 6 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Projet de groupe (PDG)	180							35		
Architectures multi-tiers (AMT) 5 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Applications multi-tiers (AMT)	150								6*	
Développement mobile et sécurité logicielle (DML) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Développement d'applications Android (DAA)	120								4	
Sécurité logicielle haut niveau (SLH)	120								4*	
Simulation, optimisation et paradigmes de programmation (SOP) 8 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Paradigmes et langages de programmation (PLP)	120								4*	
Simulation et optimisation (SIO)	120								4	
Systèmes et données distribués (SDD) 6 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Méthodes d'accès aux données (MAC)	90								4	
Systèmes distribués et répartis (SDR)	90								4	
Enseignements à choix IL (XIL) 15 ECTS | Seuil de compensation: 3.0 | Crédits fondamentaux min: 9	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Unités à choix	--									x
Innovation Crunch Time (CRUNCH) 2 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Innovation Crunch Time (CRH)	60									3.33
Travail de Bachelor (TB) 15 ECTS | Seuil de compensation: 3.0 | Seuil de répétition: 4.0	Coef.	E1	S1	S2	E2	S3	S4	E3	S5	S6
Travail de Bachelor pour TIC (TB-TIC)	450									10
Périodes par semaine			32	33	40	32	32	35	30	13

Légende :

- HES d'été E1, E2, E3 : 3 semaines
- Semestre S1, S2, S3, S4, S5, S6 : 16 semaines
- fac. : module ou unité facultatif
- Seuil de compensation : Toute note d'unité inférieure au seuil entraine l'échec du module.
- * : unités avec examen final

Mathématiques 1 (Mathematics 1)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Mathématiques 1 (Mathematics 1)
Code	:	MAT1
Année académique	:	2024-2025
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
MAT1	150	Mathématiques 1
MAD	90	Mathématiques discrètes

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
MAT1	Cours		96							
MAD	Cours		64							

2. Organisation

Crédits ECTS	:	8
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☐ Avoir validé les modules : Néant
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☒ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

À l'issue de ce module, l'étudiant·e sera capable de :

- utiliser le langage mathématique et son formalisme pour décrire et modéliser des systèmes de complexité raisonnable ;
- utiliser les outils du calcul différentiel pour modéliser et résoudre des problèmes du monde de l'ingénieur ;
- manipuler des matrices et en particulier de les utiliser pour résoudre des systèmes d'équations linéaires ;
- présenter la résolution d'un problème en appliquant et respectant les principes fondamentaux de la rédaction scientifique.

5. Contenu et formes d'enseignement

Mathématiques 1

Le but de cette unité est de donner à l'étudiant·e les notions importantes du calcul différentiel et du calcul matriciel.

Forme(s) d'enseignement : Cours

Mathématiques discrètes

Cette unité introduit les notions de base des mathématiques discrètes à la base de nombreux chapitres des mathématiques et de l'informatique. Elle a également pour but de développer et d'améliorer l'analyse des problèmes scientifiques et la rédaction rigoureuse et justifiée de leurs solutions.

Forme(s) d'enseignement : Cours

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Mathématiques 1 (MAT1) [poids: 150]

Note finale = moyenne cours x 0.5 + moyenne examen x 0.5

Mathématiques discrètes (MAD) [poids: 90]

Note finale = moyenne cours x 0.5 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{90 \times MAD + 150 \times MAT1}{240}$$

7. Modalités de remédiations

☐ Pas de remédiation

☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Mathématiques 1

- Calcul différentiel: Stewart. ANALYSE concepts et contextes. Vol 1. fonctions d'une variable. De boeck.
- Calcul matriciel: David-C-Lay. Algèbre linéaire: Théorie, exercices & aplications. Pearson

Mathématiques discrètes

- Kenneth H. Rosen, Mathématiques discrètes, édition révisée, 2006, Chenelière, Montréal.
- Ronald L. Graham, Donald Ervin Knuth, Oren Patashnik, Concrete Mathematics: A Foundation for Computer Science, 1994, Addison-Wesley.

10. Enseignants

Responsable du module : Jean-François Hêche

Unité	Responsable
Mathématiques 1	Khaled Gafaiti
Mathématiques discrètes	Jean-François Hêche

Programmation
(Computer Programming)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Programmation (Computer Programming)
Code	:	PRG
Année académique	:	2024-2025
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
PRG1	270	Programmation 1

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
PRG1	Cours		96							
	Laboratoire		96							

2. Organisation

Crédits ECTS	:	9
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☐ Avoir validé les modules : Néant
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☒ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

A la fin de ce module l'étudiant·e maitrisera les bases de la programmation procédurale en C++ et sera capable de :

- écrire des programmes simples qui interagissent avec l'utilisateur via la console standard ou via des fichiers ;
- choisir à bon escient les structures de contrôle les plus appropriées pour les branches et boucles de son code ;
- choisir les types de données appropriés, notamment en comprenant les avantages et les limites des types entiers, non signés et réels ;
- manipuler les données de son programme en les stockant dans des variables, des tableaux, des structures ou des classes ;
- manipuler des données textuelles via des chaines de caractères ou les flux d'entrée/sortie ;
- structurer son code avec des fonctions - éventuellement surchargées ou génériques - des opérateurs surchargés ou des classes, éventuellement génériques, mais sans aborder la notion d'héritage ou de polymorphisme ;
- rendre son code compréhensible et réutilisable via sa structure, le choix des noms de constantes, variables, fonctions et classes, en commentant le code et en documentant les interfaces ;
- utiliser les algorithmes de la librairie standard (STL) y compris la notion d'itérateur sur des tableaux ;
- comprendre les différents types d'allocation de mémoire disponibles (automatique, statique et dynamique) et leurs conséquences en terme de gestion des ressources ;
- gérer les erreurs lors de la compilation, du déboggage, et de l'exécution du code via les assertions et les exceptions ;
- collaborer en petit groupe pour développer un programme à plusieurs, y compris les bases du contrôle des versions du code via un outil tel que git.

5. Contenu et formes d'enseignement

Programmation 1

Cette unité permet à l'étudiant·e d'acquérir une solide connaissance de la programmation C++.

Les notions suivantes y sont notamment présentées : types de données, variables, constantes, opérateurs, instructions *if* et *switch*, boucles, fonctions, compilation séparée, tableaux classiques à une dimension, classes *vector* et *array,* librairie *algorithm*, chaînes de caractères (classe *string*), bases de programmation orientée objet, généricité et exceptions.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Programmation 1 (PRG1) [poids: 270]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité PRG1

7. Modalités de remédiations

☐ Pas de remédiation
☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Programmation 1

1. Programmer en C++ moderne, Claude Delannoy, Eyrolles 2019

2. Big C++, Cay S. Horstmann, Wiley 2017

10. Enseignants

Responsable du module : Olivier Cuisenaire

Unité	**Responsable**
Programmation 1	Olivier Cuisenaire

Systèmes logiques et science des données (Logic systems and data)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Systèmes logiques et science des données (Logic systems and data)
Code	:	SLD
Année académique	:	2024-2025
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
ISD	120	Introduction à la science des données
SYL	120	Systèmes logiques

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
ISD	Cours		32							
	Laboratoire		32							
SYL	Cours		32							
	Laboratoire		32							

2. Organisation

Crédits ECTS	:	8
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☐ Avoir validé les modules	:	Néant
☒ Avoir suivi ou suivre en parallèle les modules	:	Programmation (PRG)
☐ Pas de prérequis		

4. Compétences visées / Objectifs généraux d'apprentissage

A l'issue du module, l'étudiant-e sera capable de :

- Expliquer les modes de représentation des principaux types de données ;
- Utiliser les principaux dispositifs logiques et arithmétiques des systèmes de traitement de l'information (portes logiques, bascules, registres, circuits arithmétiques de base) ;
- Décrire et expliquer les modes de représentation des systèmes combinatoires et séquentiels (algèbre de Boole, tables de vérité, tables de Karnaugh, tables d'états, graphes des états) ;
- Utiliser des méthodes de synthèse et de simplification des systèmes combinatoires et séquentiels.
- délimiter les problèmes qui relèvent de l'apprentissage automatique ;
- distinguer les différents types de problèmes d'apprentissage et les paradigmes qui s'y rattachent (apprentissage supervisé et apprentissage non-supervisé) ;
- formaliser des énoncés de problèmes en termes de tâches d'apprentissage et mettre en place une méthode d'apprentissage appropriée ;
- utiliser des librairies du calcul scientifique conçues pour analyser les données disponibles et concevoir des modèles avec ;
- savoir interpréter les modèles créés à partir des données en faisant preuve d'esprit critique.

5. Contenu et formes d'enseignement

Introduction à la science des données

La "science des données" vise à produire des méthodes automatisées d'analyse de données massives et de sources plus ou moins complexes afin d'en extraire des informations potentiellement utiles.

Pour cela, l'ingénieur·e des données met en œuvre et exploite des méthodes pour le traitement, l'acquisition, le stockage et l'analyse de grands volumes de données, qui résultent des capteurs et des systèmes d'information de plus en plus complexes qui nous entourent. L'ingénieur·e des données vise également à interpréter ces données et à formuler des prédictions par le biais de l'apprentissage automatique (machine learning), puis à proposer des visualisations porteuses de sens.

Cette unité sert d'introduction au domaine et aux bases méthodologiques, et fournit les outils de base pour le développement de solutions.

Forme(s) d'enseignement : Cours, Laboratoire

Systèmes logiques

Contenu :

- introduction aux systèmes numériques ;
- présentation des systèmes de numération et représentation des nombres dans l'ordinateur ;
- présentation des fonctions logiques (AND, OR, NOT, etc) et utilisation de l'algèbre de Boole pour la simplification des expressions complexes ;
- conception des systèmes combinatoires ;
- explication détaillée de la méthodologie de conception et d'analyse des systèmes séquentiels.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Introduction à la science des données (ISD) [poids: 120]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Systèmes logiques (SYL) [poids: 120]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = $\frac{120 \times SYL + 120 \times ISD}{240}$

7. Modalités de remédiations

☐ Pas de remédiation
☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Introduction à la science des données

- Azencott, Chloé-Agathe. Introduction au machine learning. Dunod, 2019.
- Domingos, Pedro. The master algorithm: How the quest for the ultimate learning machine will remake our world. Basic Books, 2015.
- VanderPlas, Jake. Python data science handbook: Essential tools for working with data. " O'Reilly Media, Inc.", 2016 (https://jakevdp.github.io/PythonDataScienceHandbook/).

Systèmes logiques

- A. Nketsa et D. Delauzun, "Systèmes électroniques numériques complexes", Ed. Ellipses, 2012
- T. Floyd, "Digital Fundamentals", Prentice Hall, 2014
- R. Bryant, D. O'Hallaron, "Computer systems: a programmers perspective", Prentice Hall, 2015

10. Enseignants

Responsable du module : Andres Perez Uribe

Unité	Responsable
Introduction à la science des données	Andres Perez Uribe
Systèmes logiques	Romuald Mosqueron

Communication 1

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Communication 1
Code : COM1
Année académique : 2024-2025
Type de formation : Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
ENG1	60	Anglais 1
EXP	60	Expression et communication

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
ENG1	Cours			48						
EXP	Cours		32							

2. Organisation

Crédits ECTS : 4
Langue(s) principale(s) d'enseignement : Français, Anglais

3. Prérequis

☐ Avoir validé les modules : Néant
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☒ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Etre en mesure de s'exprimer de manière pertinente et efficace en anglais et en français dans son domaine professionnel :

- comprendre la langue anglaise et s'exprimer de manière appropriée ;
- réaliser des présentations orales structurées et convaincantes.

5. Contenu et formes d'enseignement

Anglais 1

L'étudiant·e sera mis·e en situation pour utiliser et approfondir sa connaissance de la langue anglaise, ce qui lui permettra d'être à l'aise pour bien comprendre la langue dans des situations professionnelles et de s'exprimer efficacement et couramment. La forme d'enseignement est le cours.

Forme(s) d'enseignement : Cours

Expression et communication

Cett unité est destinée à sensibiliser l'étudiant-e sur l'importance de la communication orale et écrite dans la vie professionnelle et privée.

Forme(s) d'enseignement : Cours

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Anglais 1 (ENG1) [poids: 60]

Note finale = moyenne cours x 0.5 + moyenne examen x 0.5

Expression et communication (EXP) [poids: 60]

Note finale = moyenne cours x 1

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{60 \times ENG1 + 60 \times EXP}{120}$$

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Anglais 1

Cambridge English Grammar and Vocabulary (CUP) - for B2 First ou C1 Advanced selon le niveau de l'étudiant.e

Expression et communication

- Ammiar, B., Kohneh-Chahri, O., & Petitbon, F. (2019). *La boîte à outils du coaching?*(3ème). Dunod.
- Fisher, R., Ury, W., & Patton, B. (2006). *Comment réussir une négociation*. Seuil.
- Gillet-Goinard, F., & Maimi, L. (2020). *La boîte à outils pour animer vos réunions*. Dunod. https://www.dunod.com/entreprise-economie/boite-outils-pour-animer-vos-reunions
- Joule, R.-V., & Beauvois, J.-L. (2017). *Petit traité de manipulation à l'usage des honnêtes gens*.
- Schmid Mast, M., Palese, T., & Tur, B. (2019). *Leaderspritz. Le cocktail du leadership interpersonnel—La recette des managers pour communiquer, comprendre, convaincre*. EPFL Press. https://www.epflpress.org/produit/920/9782889152926/
- Stimec, A., & Benitah, A. (2019). *La boîte à outils du dialogue en entreprise*. Dunod.
- Wood, S. E., Green Wood, E., Boyd, D., & Hétu, F. (2015). *L'univers de la psychologie* (2ème). ERPI.

10. Enseignants

Responsable du module : Myriam Malherbe

Unité	Responsable
Anglais 1	Georgina Cretegny
Expression et communication	Myriam Malherbe

Algorithmes et structures de données (Algorithms and Data Structures)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Algorithmes et structures de données (Algorithms and Data Structures)
Code	:	ASD
Année académique	:	2024-2025
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
ASD	180	Algorithmes et structures de données

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
ASD	Cours			64						
	Laboratoire			64						

2. Organisation

Crédits ECTS	:	6
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☒ Avoir validé les modules : Programmation (PRG)
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

A l'issue de ce module, l'étudiant·e sera capable de :

- comprendre la notion de complexité et pouvoir l'appliquer au choix d'un algorithme ou de structures de données ;
- comprendre les principes de la récursivité et ses applications classiques (e.g. Hanoi, permutation, minimax) et concevoir une solution récursive à un problème ;
- choisir un algorithme de tri en fonction de ses caractéristiques (stabilité, complexités spatiale et temporelle, types des données à trier), en expliquer son fonctionnement et le mettre en œuvre ;
- choisir une structure de données (linéaire, arbre, graphe, tables de hachage) en fonction de ses caractéristiques (complexités spatiales et temporelles, opérations disponibles), en expliquer son fonctionnement et la mettre en œuvre ;
- expliquer et justifier les choix réalisés pour résoudre un problème, critiquer une solution proposée ;
- connaître les structures de données existantes et savoir choisir et implémenter la structure de donnée la plus adaptée à une problématique concrète (STL, C++) ;
- connaître les algorithmes de base pour manipuler des structures de données et savoir les réutiliser dans des situations plus complexes en les adaptant et en les combinant (STL, C++).

5. Contenu et formes d'enseignement

Algorithmes et structures de données

Cette unité d'enseignement permet à l'étudiant-e de se familiariser, d'utiliser et de concevoir des algorithmes et des structures de données de base nécessaires pour la conception d'un programme. Elle introduit les notions de complexité d'un algorithme et de type de données abstrait. Elle présente la récursivité et les différents algorithmes de tri. Les structures linéaires avec leurs différentes particularités sont étudiées. Les arbres, et en particulier, les algorithmes pour les arbres binaires de recherche sont approfondis. Les fonctions de hachage, la résolution de collisions et des applications sont présentés. Cette unité finit par une introduction aux graphes ainsi que par la présentation d'algorithmes de base permettant leur manipulation.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Algorithmes et structures de données (ASD) [poids: 180]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité ASD

7. Modalités de remédiations

☐ Pas de remédiation

☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Algorithmes et structures de données

1. "Algorithms", 4/E, Robert Sedgewick and Kevin Wayne, Addison-Wesley Professional, 2011, ISBN 0132762560, 9780132762564.
2. " Introduction to algorithms", Thomas Cormen, Charles Leiserson, Ronald Rivest, Clifford Stein, Third Edition, MIT Press, 2009, ISBN-10: 0262033844.
3. "Algorithmes: Notions de base", Thomas Cormen, Éditeur Dunod, 2013, ISBN 2100702904, 9782100702909.

10. Enseignants

Responsable du module : Laura Elena Raileanu

Unité	Responsable
Algorithmes et structures de données	Laura Elena Raileanu

Architecture et sécurité informatique

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Architecture et sécurité informatique
Code : ASI
Année académique : 2024-2025
Type de formation : Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
ARO	105	Architectures des ordinateurs
ISI	105	Introduction à la sécurité de l'information

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
ARO	Cours			32						
	Laboratoire			32						
ISI	Cours			32						
	Laboratoire			32						

2. Organisation

Crédits ECTS : 7
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☐ Avoir validé les modules : Néant

☒ Avoir suivi ou suivre en parallèle les modules : Programmation et réseaux informatiques (PRI), Systèmes logiques et science des données (SLD)

☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Le développement de solutions et systèmes informatiques ne se limite pas au développement logiciel pur. Il faut également tenir compte de l'environnement dans lequel fonctionnera le système. L'architecture du matériel ainsi que la sécurité en sont des éléments importants. Ces domaines sont devenus incontournables pour comprendre les principes de fonctionnement des logiciels au sein du matériel et d'en sécuriser le contenu ou l'accès. Ce module permet d'en acquérir les compétences de base.

5. Contenu et formes d'enseignement

Architectures des ordinateurs

Cette unité est organisée comme suit :

- présentation des éléments de base d'un ordinateur, leurs caractéristiques, leurs performances et leurs interactions ;
- description bloc par bloc de l'architecture d'un processeur ;
- explication détaillée de la technique de pipeline ;
- explication de l'utilisation des mémoires caches pour l'amélioration des performances ;
- présentation des divers types de mémoire et des technologies ;
- présentation et pratique d'un jeu partiel d'instructions en langage machine (assembleur) ;
- conception, par des travaux de laboratoire, d'un processeur de type ARM à l'aide du logiciel de simulation Logisim.

Forme(s) d'enseignement : Cours, Laboratoire

Introduction à la sécurité de l'information

Cette unité donne les bases de la sécurité de l'information avec notamment des aspects de sécurité organisationnelle, physique, légale et technique. Le cours donne un aperçu des menaces et protections, avec notamment les codes malveillants, sécurité logicielle, Web, réseaux, cryptographie et bien d'autres.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0

Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Architectures des ordinateurs (ARO) [poids: 105]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Introduction à la sécurité de l'information (ISI) [poids: 105]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{105 \times ISI + 105 \times ARO}{210}$$

7. Modalités de remédiations

☐ Pas de remédiation

☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Architectures des ordinateurs

- Systèmes électroniques numériques complexes / Alexandre Nketsa, Damien Delauzun. Ellipses, Technosup 2012.
- ARM assembly language; fundamentals and techniques / Hohl, William. Taylor & Francis 2009
- ARM System-on-Chip Architecture / Steve Furber . Pearson 2000

Introduction à la sécurité de l'information

10. Enseignants

Responsable du module : Romuald Mosqueron

Unité	Responsable
Architectures des ordinateurs	Romuald Mosqueron
Introduction à la sécurité de l'information	Sylvain Pasini

Mathématiques 2
(Mathematics 2)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Mathématiques 2 (Mathematics 2)
Code	:	MAT2
Année académique	:	2024-2025
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
MAT2	150	Mathématiques 2

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
MAT2	Cours			96						

2. Organisation

Crédits ECTS	:	5
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☒ Avoir validé les modules : Mathématiques 1 (MAT1)
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

À l'issue de ce module, l'étudiant·e sera capable de :

- utiliser le langage mathématique et son formalisme pour décrire et modéliser des systèmes de complexité raisonnable ;
- présenter la résolution d'un problème en appliquant et respectant les principes fondamentaux de la rédaction scientifique ;
- utiliser les outils du calcul intégral pour résoudre des problèmes du monde de l'ingénieur ;
- étudier des structures combinatoires simples et de résoudre des problèmes de dénombrement et de probabilités discrètes ;
- modéliser et de manipuler analytiquement les objets classiques de la géométrie du plan et de l'espace.

5. Contenu et formes d'enseignement

Mathématiques 2

Cette unité présente les notions de base du calcul intégral ainsi que les techniques fondamentales pour le traitement analytique de la géométrie du plan et de l'espace. Un chapitre est également consacré aux techniques usuelles de dénombrement et à leur application au calcul de la probabilité d'un événement dans une expérience aux issues élémentaires équiprobables.

Forme(s) d'enseignement : Cours

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Mathématiques 2 (MAT2) [poids: 150]

Note finale = moyenne cours x 0.5 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité MAT2

7. Modalités de remédiations

☐ Pas de remédiation
☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Mathématiques 2

- Kenneth H. Rosen, Mathématiques discrètes, édition révisée, 2006, Chenelière, Montréal.
- James Stewart, Analyse 1 concepts et contextes - fonctions d'une variable, 2011, De Boeck Supérieur, Bruxelles.
- Earl W. Swokowski, Analyse, 5ème édition, 1993, De Boeck Supérieur, Bruxelles.
- Earl W. Swokowski, Jeffrey A. Cole, Trigonométrie, géométrie vectorielle et géométrie analytique, 2007, LEP Loisirs et Pédagogie, Lausanne.

10. Enseignants

Responsable du module : Jean-François Hêche

Unité	Responsable
Mathématiques 2	Jean-François Hêche

Programmation et réseaux informatiques (Computer Programming and Networks)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Programmation et réseaux informatiques (Computer Programming and Networks)
Code	:	PRI
Année académique	:	2024-2025
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
PRG2	105	Programmation 2
RXI	105	Réseaux informatiques

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
PRG2	Cours			32						
	Laboratoire			32						
RXI	Cours			32						
	Laboratoire			32						

2. Organisation

Crédits ECTS	:	7
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☐ Avoir validé les modules	:	Néant
☒ Avoir suivi ou suivre en parallèle les modules	:	Programmation (PRG)
☐ Pas de prérequis		

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module enseigne le langage C et donne une introduction aux réseaux informatiques, tels que les réseaux LAN/WAN physiques (ordinateurs, serveurs, équipement réseau) ou les infrastructures virtualisées (containers Docker interconnectés par un réseau virtualisé).

A l'issue de ce module, l'étudiant·e sera capable de :

- maîtriser la syntaxe du langage C ;
- concevoir des programmes dans ce langage ;
- mettre en place des réseaux IP LAN et WAN ;
- analyser et dépanner le fonctionnement de l'infrastructure réseau ;
- déployer des applications client-serveur sur une infrastructure et analyser/dépanner leur comportement.

5. Contenu et formes d'enseignement

Programmation 2

Cette unité permet à l'étudiant·e d'acquérir une solide connaissance de la programmation C.

Les notions suivantes y sont notamment présentées : préprocesseur, macros, compilation conditionnelle, types de base, opérateurs arithmétiques et logiques, structures et types composés, conversion de types (transtypage), utilisation des pointeurs, allocation dynamique et restitution de la mémoire, fonctions de manipulation mémoire, structures dynamiques (listes chaînées), chaînes de caractères, fonctions de manipulation de chaînes de caractères, notions avancées sur les pointeurs (pointeurs sur fonction), gestion des fichiers de type texte et binaire, accès séquentiel et direct aux fichiers.

Forme(s) d'enseignement : Cours, Laboratoire

Réseaux informatiques

Cette unité donne une introduction aux réseaux informatiques, tels que les réseaux LAN/WAN physiques (ordinateurs, serveurs, équipemenent réseau) ou les infrastructures virtualisées (containers Docker interconnectés par un réseau virtualisé).

Elle permettra aux étudiant·e·s d'apprendre à mettre en place des réseaux IP ainsi que d'analyser et dépanner le fonctionnement de l'infrastructure réseau.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Programmation 2 (PRG2) [poids: 105]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Réseaux informatiques (RXI) [poids: 105]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{105 \times PRG2 + 105 \times RXI}{210}$$

7. Modalités de remédiations

☐ Pas de remédiation
☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Programmation 2

Le guide complet du langage C, Claude Delannoy, Eyrolles 2020

Réseaux informatiques

10. Enseignants

Responsable du module : Olivier Cuisenaire

Unité	Responsable
Programmation 2	Daniel Rossier
Réseaux informatiques	Jürgen Ehrensberger

Communication 2

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Communication 2
Code : COM2
Année académique : 2025-2026
Type de formation : Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
ENG2	60	Anglais 2
DTS	30	Design thinking and sprint

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
ENG2	Cours				48					
DTS	Projet				24					

2. Organisation

Crédits ECTS : 3
Langue(s) principale(s) d'enseignement : Français, Anglais

3. Prérequis

☒ Avoir validé les modules : Communication 1 (COM1)
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet de renforcer les compétences en communication développées en première année :

- dans des situations professionnelles, être en mesure de comprendre et de s'exprimer de manière pertinente et efficace en anglais ;
- réaliser des présentations écrites et orales permettant de transmettre le message-clé de manière synthétique et convaincante.

Ce module permet également d'élargir le développement des "soft skills" par l'expérimentation d'un processus d'innovation centré sur les utilisateurs. Celui-ci a pour but d'amener de nouvelles solutions à des problématiques ou des challenges spécifiques définis par une entreprise mandante.

5. Contenu et formes d'enseignement

Anglais 2

ENG2 constitue la suite de ENG1.

L'étudiant·e sera mis·e en situation pour utiliser et approfondir sa connaissance de la langue anglaise, ce qui lui permettra d'être à l'aise pour bien comprendre la langue dans des situations professionnelles et de s'exprimer efficacement et couramment. La forme d'enseignement est le cours.

Forme(s) d'enseignement : Cours

Design thinking and sprint

Dans les milieux industriels et de la recherche, une base de "hard skills" (compétences scientifiques et techniques) solide telle que celle de ISC est grandement appréciée. Cependant, les "soft skills" (compétences relationnelles et humaines) sont de plus en plus demandées, car il est important de savoir interagir, coopérer et communiquer au sein des groupes et à travers les entités d'une entreprise.

L'unité DTS vise à renforcer les "soft skills" par le biais des approches suivantes :

- le "design thinking" est un processus de pensée créative qui met l'utilisateur final ou le client au centre de la réflexion, afin d'amener des solutions innovantes ou originales à des problématiques ou des challenges spécifiques. Le facteur humain est essentiel dans la démarche. Les phases principales sont "Empathize" (comprendre les besoins de l'utilisateur final ou du client), "Define" (définir et reformuler le problème), "Ideate" (réfléchir par brainstorming aux solutions possibles), "Prototype" (prototypage d'une solution), "Test" (obtenir un retour du public cible), selon une série de cycles itératifs ;
- le "innovation sprint" est un marathon créatif qui cadre temporellement un processus de "design thinking", typiquement sur une durée de 1 à 5 jours. Ces sprints se conduisent par équipes de quelques personnes.

L'unité DTS est déclinée en 3 jours d'innovation sprint, où des modules théoriques de design thinking seront dispensés tout au long du sprint. Un thème général sera proposé, à partir duquel seront déclinés des problématiques spécifiques à solutionner. Les étudiant·e·s seront regroupés par équipes de 4 ou 5 personnes, qui collaboreront à des solutions originales aux problématiques proposées. Durant les 3 jours de DTS, l'accent principal est mis essentiellement sur les étapes "Empathize", "Define", "Ideate", et partiellement sur les étapes "Prototype" et "Test".

Un coach (enseignant·e) sera affecté à chaque équipe et servira comme mentor (accompagnateur) dans la démarche de design thinking.

Les délivrables principaux que chaque équipe devra produire sont orientés vers les documents de communication orale et écrite qui permettent de vendre les idées (par exemple à des décideurs ou des investisseurs) dans un temps très cours :

- « one-pager » écrit: un document de synthèse de 1 page A4 (type flyer) "notre solution en 1 page" ;
- « pitch vidéo »: une capsule vidéo (clip) "notre solution en 1 minute".

Forme(s) d'enseignement : Projet

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Anglais 2 (ENG2) [poids: 60]

Note finale = moyenne cours x 1

Design thinking and sprint (DTS) [poids: 30]

Note finale = moyenne projet x 1

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = $\frac{30 \times DTS + 60 \times ENG2}{90}$

7. Modalités de remédiations

☒ Pas de remédiation
☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Anglais 2

Bibliographie

Cambridge English Grammar and Vocabulary - B2 First or C1 Advanced selon le niveau de l'étudiant-e

Design thinking and sprint

- Design thinking: https://fr.wikipedia.org/wiki/Design_thinking
- Design sprint: https://fr.wikipedia.org/wiki/Design_sprint
- Pitch: https://fr.wikipedia.org/wiki/Elevator_pitch

10. Enseignants

Responsable du module : Myriam Malherbe

Unité	Responsable
Anglais 2	Georgina Cretegny
Design thinking and sprint	Vincent Peiris

Projet d'informatique
(Software Development Project)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Projet d'informatique (Software Development Project)
Code	:	PIN
Année académique	:	2025-2026
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
PIN	90	Projet d'informatique

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
PIN	Projet				48					

2. Organisation

Crédits ECTS	:	3
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☒ Avoir validé les modules : Algorithmes et structures de données (ASD)
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet aux étudiant·e·s de :

- travailler en groupe ;
- maitriser des outils de développement collaboratif ;
- développer une UI en C++ ;
- être introduits à la programmation événementielle.

5. Contenu et formes d'enseignement

Projet d'informatique

Cette unité se déroule sous forme de projet de développement d'une application avec GUI en C++. Elle est réalisée en petits groupes (typiquement 3 personnes) et se termine par la présentation du résultat final ainsi qu'une compétition entre les groupes qui compare l'efficacité de leurs solutions.

Forme(s) d'enseignement : Projet

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Projet d'informatique (PIN) [poids: 90]

Note finale = moyenne projet x 1

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité PIN

7. Modalités de remédiations

☒ Pas de remédiation
☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Projet d'informatique

10. Enseignants

Responsable du module : Olivier Cuisenaire

Unité	Responsable
Projet d'informatique	Olivier Cuisenaire

Bases de données et applications internet (Databases and internet applications)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Bases de données et applications internet (Databases and internet applications)
Code	:	BDA
Année académique	:	2025-2026
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
BDR	150	Bases de données relationnelles
DAI	90	Développement d'applications internet

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
BDR	Cours					48				
	Laboratoire					48				
DAI	Cours					32				
	Laboratoire					32				

2. Organisation

Crédits ECTS : 8
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☐ Avoir validé les modules : Néant
☒ Avoir suivi ou suivre en parallèle les modules : Programmation et réseaux informatiques (PRI), Programmation orienté objet (POO)

☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet de maitriser les concepts de bases liés à la création d'applications connectées : conception et mise en œuvre de bases de données relationnelles et développement d'applications communiquant à travers le réseau.

5. Contenu et formes d'enseignement

Bases de données relationnelles

L'objectif de cette unité est de maîtriser les éléments essentiels pour la conception, la mise en œuvre et l'utilisation des bases de données relationnelles, plus précisement :

- concevoir les schémas conceptuels et relationnels de base de données avec une démarche d'ingénieur ;
- comprendre et utiliser le modèle relationnel, les contraintes d'intégrité et l'algèbre relationnelle ;
- implémenter une base de données sur un systéme de gestion de base de données (SGBD) relationnelles et utiliser le langage SQL pour la définition, la manipulation et le contrôle des données ;
- décrire les formes normales et les appliquer pour vérifier la qualité d'une base de données ;
- utiliser les bases de données au travers un programme Java via l'API JDBC.

Forme(s) d'enseignement : Cours, Laboratoire

Développement d'applications internet

Après avoir suivi les cours d'introduction à la programmation, ce cours enseigne la réalisation d'applications qui communiquent à travers le réseau. Presque toutes les applications modernes sont de ce type: applications Web ou mobiles, applications client-serveur, applications peer-to-peer, etc. Le cours se concentre principalement les aspect de programmation (Java et JavaScript), mais traite aussi des aspects de génie logiciel (spécification, tests unitaires, mesure de performances), d'outils (Git, Docker) et d'infrastructure (load balancing, redondance).

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Bases de données relationnelles (BDR) [poids: 150]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Développement d'applications internet (DAI) [poids: 90]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{150 \times BDR + 90 \times DAI}{240}$$

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Bases de données relationnelles

- "Database systems, models, languages, design, and application programming", R. Elmasri, S. Navathe, 6th edition, 2015, Pearson education. ISBN: 0133970779, 9780133970777.
- "Database systems the complete book", Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom, 2009, Prentice Hall. ISBN: 0131873253, 9780131873254.
- "An introduction to database systems" , DATE, C. J.,Addison Wesley, 2004. ISBN: 0321197844, 9780321197849
- "Bases de donneés", G. Gardarin, Eyrolles, 2005.
- "Système de gestion bases de données", KORTH, F., SILBERSCHATZ A., McGraw-Hill, 1989.

Développement d'applications internet

https://docs.oracle.com/javase/tutorial/networking/index.html

10. Enseignants

Responsable du module : Bertil Chapuis

Unité	Responsable
Bases de données relationnelles	Nastaran Fatemi
Développement d'applications internet	Jürgen Ehrensberger

Mathématiques 3
(Mathematics 3)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Mathématiques 3 (Mathematics 3)
Code	:	MAT3
Année académique	:	2025-2026
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
MAT3	120	Mathématiques 3
PST	120	Probabilités et statistiques

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
MAT3	Cours					64				
PST	Cours					64				

2. Organisation

Crédits ECTS	:	8
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☒ Avoir validé les modules : Mathématiques 2 (MAT2)
☒ Avoir suivi ou suivre en parallèle les modules : Systèmes logiques et science des données (SLD)
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet de développer des compétences essentielles dans plusieurs domaines des mathématiques appliquées aux sciences de l'ingénieur en exposant des outils aussi centraux que les nombres complexes, les équations différentielles, la théorie des probabilités ou, encore, l'analyse statistique des données.

5. Contenu et formes d'enseignement

Mathématiques 3

Cette unité offre une présentation des nombres complexes, des équations différentielles ainsi que de quelques applications de ces outils mathématiques.

Forme(s) d'enseignement : Cours

Probabilités et statistiques

De nos jours, les données jouent un rôle de plus en plus prépondérant dans les entreprises, sociétés et organisations. Elles circulent rapidement en flux et sont créées sous diverses formes (structurées ou non) en énormes quantités (mégadonnées) et bien souvent de manière automatisée. Dans cette unité, il est proposé d'introduire des méthodes statistiques pour extraire des informations pertinentes cachées dans les données et d'être familiarisé au calcul des probabilités en l'illustrant par différentes applications (filtre bayesiens anti-spams, introduction au codage avec ou sans bruit, fiabilité...). Pour y parvenir, est utilisé **R**, logiciel libre de statistique qui a fait ses preuves dans l'industrie et dans le milieu académique.

Forme(s) d'enseignement : Cours

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Mathématiques 3 (MAT3) [poids: 120]

Note finale = moyenne cours x 1

Probabilités et statistiques (PST) [poids: 120]

Note finale = moyenne cours x 1

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{120 \times PST + 120 \times MAT3}{240}$$

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Mathématiques 3

- James Stewart, Analyse 1 concepts et contextes - fonctions d'une variable, 2011, De Boeck Supérieur, Bruxelles.
- Earl W. Swokowski, Analyse, 5ème édition, 1993, De Boeck Supérieur, Bruxelles.

Probabilités et statistiques

- Dalgaard, P. (2008). Introductory Statistics with **R**. Second Edition. New York: Springer.
- Daly, F., Hand, D.J., Jones, M.C., Lunn, A.D., and McConway, K.J. (1995). Elements of Statistics. Addison-Wesley, Harlow, England.
- Drouilhet, R., Lafaye de Micheaux, P., et Liquet, B. (2011). Le logiciel **R** : Maîtriser le langage, Effectuer des analyses statistiques. Paris: Springer-Verlag.
- Gonick, L. & Smith, W. (1993). The Cartoon Guide to Statistics. HarperCollins,New-York.
- Maindonald, J. & Braun, J. (2010). Data Analysis and Graphics Using **R**. Third Edition. Cambridge: Cambridge University Press.
- Morgenthaler, S. (2014). Introduction à la Statistique (4ème édition). Presses Polytechniques et Universitaires Romandes, PPUR, Lausanne.
- Ross, S. M. (2014). Initiation aux Probabilités (Traduction de la 9ème édition américaine). Presses Polytechniques et Universitaires Romandes, PPUR, Lausanne.
- Wild, C.J. & Seber, G. A.F. (2000). Chance Encounter, A First Course in Data Analysis and Inference, Wiley, New York.

10. Enseignants

Responsable du module : Jean-François Hêche

Unité	Responsable
Mathématiques 3	Jean-François Hêche
Probabilités et statistiques	Jacques Zuber

Programmation orienté objet (Object oriented programming)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Programmation orienté objet (Object oriented programming)
Code	:	POO
Année académique	:	2025-2026
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
POO	150	Programmation orientée objet

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
POO	Cours					48				
	Laboratoire					48				

2. Organisation

Crédits ECTS	:	5
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☒ Avoir validé les modules : Algorithmes et structures de données (ASD)
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet d'acquérir les compétences permettant de modéliser, concevoir et réaliser des applications orienté objet.

5. Contenu et formes d'enseignement

Programmation orientée objet

Cette unité permet d'acquérir et de pouvoir mettre en oeuvre les concepts de base de la conception et de la programmation orientée objet : diagrammes de classes UML, classes, interfaces, propriétés statiques, héritage, classes et méthodes abstraites, polymorphisme, liaison dynamique, types énumérés, égalité d'objets, copie d'objets, classes internes et programmation évenementielle, exceptions et généricité.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Programmation orientée objet (POO) [poids: 150]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité POO

7. Modalités de remédiations

☒ Pas de remédiation
☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Programmation orientée objet

10. Enseignants

Responsable du module : Pier Donini

Unité	Responsable
Programmation orientée objet	Pier Donini

Systèmes d'exploitation et concurrence (Operating systems and concurrent programming)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Systèmes d'exploitation et concurrence (Operating systems and concurrent programming)
Code	:	SEC
Année académique	:	2025-2026
Type de formation	:	Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
PCO	105	Programmation concurrente
SYE	105	Systèmes d'exploitation

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
PCO	Cours					32				
	Laboratoire					32				
SYE	Cours					32				
	Laboratoire					32				

2. Organisation

Crédits ECTS : 7
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☒ Avoir validé les modules : Programmation et réseaux informatiques (PRI)
☒ Avoir suivi ou suivre en parallèle les modules : Architecture et sécurité informatique (ASI)
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Les systèmes d'exploitation (OS) constituent le coeur d'un système informatique ; ils permettent l'exécution d'applications performantes et riches en fonctionnalité.

Ce module permet à l'étudiant·e de se familiariser, d'une part, avec les mécanismes internes d'un OS intervenant dans la sécurisation, la performance, la réactivité et la portabilité des applications et, d'autre part, avec la notion de concurrence nécessaire à la gestion de ressources tant matérielles que logicielles qui doivent être partagées entre les applications.

Il permet également d'acquérir les compétences requises pour le développement d'applications multitâches ainsi que pour la gestion adéquate des interactions entre applications et les différents services d'un système d'exploitation.

5. Contenu et formes d'enseignement

Programmation concurrente

A l'issue de ce cours l'étudiant-e aura acquis des connaissances liées à la programmation multi-tâches. Il-elle saura décrire une tâche et développer des applications mettant en oeuvre les mécanismes de synchronisation que sont les verrous, les sémaphores et les variables conditions. Il-elle aura en outre acquis une bonne connaissance des paradigmes suivants: producteurs-consommateurs, lecteurs-rédacteurs, moniteurs.

Forme(s) d'enseignement : Cours, Laboratoire

Systèmes d'exploitation

Cette unité donne aux étudiant·e·s un aperçu détaillé des mécanismes internes aux systèmes d'exploitation en approchant les différents sous-systèmes. Les architectures principales des OS, la gestion des processus et des threads, la gestion mémoire et les systèmes de fichiers constituent les principaux sujets de ce cours. Les laboratoires permettent aux étudiant·e·s de bien comprendre les interactions entre le noyau d'un OS et les applications utilisateurs.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Programmation concurrente (PCO) [poids: 105]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Systèmes d'exploitation (SYE) [poids: 105]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{105 \times PCO + 105 \times SYE}{210}$$

7. Modalités de remédiations

☒ Pas de remédiation
☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Programmation concurrente

Systèmes d'exploitation

10. Enseignants

Responsable du module : Daniel Rossier

Unité	Responsable
Programmation concurrente	Yann Thoma
Systèmes d'exploitation	Daniel Rossier

Conception orientée objet (Object oriented conception)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Conception orientée objet (Object oriented conception)
Code	:	COO
Année académique	:	2025-2026
Type de formation	:	Bachelor

Niveau

☐ Module de base
☒ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
MCR	90	Modèles de conception réutilisables
POA	90	Programmation orientée objet avancée

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
MCR	Cours						16			
	Laboratoire						32			
POA	Cours						24			
	Laboratoire						24			

2. Organisation

Crédits ECTS : 6
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☒ Avoir validé les modules : Programmation et réseaux informatiques (PRI), Programmation orienté objet (POO)
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet d'acquerir les compétences permettant d'expliquer et de mettre en oeuvre les concepts avancés de modélisation et de programmation orientée objet.

5. Contenu et formes d'enseignement

Modèles de conception réutilisables

Cette unité d'enseignement permet de décrire et implémenter les principaux modèles de conception réutilisables (*design patterns*), de reconnaître les situations où ils peuvent être mis en oeuvre et de concevoir des applications qui les utilisent.

Forme(s) d'enseignement : Cours, Laboratoire

Programmation orientée objet avancée

Cette unité d'enseignement permet d'acquérir et de mettre en oeuvre les concepts de base de la programmation orientée objet en C++, ainsi que ceux avancés de C++ (généricité, liste d'initialiseurs, iterateurs, for_each, foncteurs, pointeurs intelligents) et de Java (interfaces fonctionnelles, lambda expressions, streams).

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Modèles de conception réutilisables (MCR) [poids: 90]

Note finale = moyenne cours x 0.6 + moyenne laboratoire x 0.4

Programmation orientée objet avancée (POA) [poids: 90]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{90 \times POA + 90 \times MCR}{180}$$

7. Modalités de remédiations

☒ Pas de remédiation
☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Modèles de conception réutilisables

Programmation orientée objet avancée

10. Enseignants

Responsable du module : Pier Donini

Unité	Responsable
Modèles de conception réutilisables	Pier Donini
Programmation orientée objet avancée	Pier Donini

Développement logiciel et responsabilités

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Développement logiciel et responsabilités
Code : DLR
Année académique : 2025-2026
Type de formation : Bachelor

Niveau

☒ Module de base
☐ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
EAL	45	Ethique et aspects légaux
PDL	105	Processus de développement en ingénierie logicielle

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
EAL	Cours						32			
PDL	Cours						32			
	Laboratoire						32			

2. Organisation

Crédits ECTS : 5
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☒ Avoir validé les modules : Programmation orienté objet (POO)
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module offre aux étudiant·e·s une formation complète couvrant les aspects techniques et éthiques du développement logiciel. D'une part, il vise à former les étudiant·e·s aux méthodes et processus de développement logiciel, en les exposant aux pratiques courantes d'ingénierie logicielle et aux défis techniques qu'elles impliquent. D'autre part, ce module engage les étudiant·e·s dans une réflexion approfondie sur les responsabilités éthiques et légales liées à la pratique professionnelle en ingénierie logicielle. Les étudiant·e·s apprendront à concilier les exigences techniques avec des considérations éthiques et légales, leur permettant ainsi de prendre des décisions éclairées et responsables dans leur future carrière.

5. Contenu et formes d'enseignement

Ethique et aspects légaux

L'ingénieur·e ISC est confronté·e à de nombreux choix éthiques et légaux dans le contexte de sa pratique professionnelle. Les formes que peuvent prendre les risques légaux et les conflits éthiques potentiels sont variées et changeantes, rendant difficile de fixer des repères quant au comportement à adopter. Cette unité vise à amorcer une réflexion devant conduire à la détermination d'une ligne de conduite personnelle responsable fondée sur les principes de l'éthique prenant en compte le contexte juridique et légal.

Forme(s) d'enseignement : Cours

Processus de développement en ingénierie logicielle

Cette unité permet de découvrir le processus de développement logiciel dans son ensemble, c'est-à-dire, au travers des activités de spécification, de conception, d'implémentation, de test et de maintenance. L'agencement de ces activités varie d'un projet à l'autre et, en fonction du contexte, le processus de développement peut être exécuté en cascade ou de manière incrémentale et itérative. L'objectif du cours est de se familiariser avec des pratiques d'ingénierie logicielle (XP, UML, CI/CD, etc.) qui permettent à une équipe de s'organiser, de collaborer et de répondre aux besoins formulés par des utilisateur·rice·s en respectant des contraintes de coûts, de délais et de qualité.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Ethique et aspects légaux (EAL) [poids: 45]

Note finale = moyenne cours x 1

Processus de développement en ingénierie logicielle (PDL) [poids: 105]

Note finale = moyenne cours x 0.67 + moyenne laboratoire x 0.33

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{45 \times EAL + 105 \times PDL}{150}$$

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Ethique et aspects légaux

Processus de développement en ingénierie logicielle

- A Philosophy of Software Design, John K. Ousterhout, Yaknyam Press, 2018.
- Software Engineering, Ian Sommerville, Pearson, 2016.
- Learning Agile: Understanding Scrum, XP, Lean, and Kanban, par Andrew Stellman et Jennifer Greene, O'Reilly, 2014.
- UML distilled, par Martin Fowler, Addison Wesley, 2013.
- Applying UML and Design patterns, par Craig Larman, Prentice Hall, 2004.
- Object Oriented Modeling & Design with UML, par R. Blaha & James Rumbaugh, Pearson, 2004.
- RUP, XP, architectures et outils : Industrialiser le processus de développement, par Pierre-Yves Cloux, 2003 Dunod.

10. Enseignants

Responsable du module : Bertil Chapuis

Unité	Responsable
Ethique et aspects légaux	Eric Bruyndonckx
Processus de développement en ingénierie logicielle	Bertil Chapuis

Graphes et réseaux de neurones (Graphs and neural networks)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Graphes et réseaux de neurones (Graphs and neural networks)
Code	:	GRN
Année académique	:	2025-2026
Type de formation	:	Bachelor

Niveau

☐ Module de base
☒ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
ARN	90	Apprentissage par réseaux de neurones artificiels
GRE	150	Graphes et réseaux

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
ARN	Cours						32			
	Laboratoire						32			
GRE	Cours						64			
	Laboratoire						32			

2. Organisation

Crédits ECTS	:	8
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☒ Avoir validé les modules	:	Algorithmes et structures de données (ASD), Mathématiques 3 (MAT3), Systèmes logiques et science des données (SLD)
☐ Avoir suivi ou suivre en parallèle les modules	:	Néant
☐ Pas de prérequis		

4. Compétences visées / Objectifs généraux d'apprentissage

A l'issue de ce module, l'étudiant·e sera capable de :

- comprendre le fonctionnement des réseaux de neurones et des différentes architectures ;
- comprendre les techniques d'apprentissage pour les réseaux de neurones (algorithme de rétropropagation du gradient),
- mettre en place un pipeline de modélisation à l'aide des réseaux de neurones ;
- utiliser des librairies de Machine Learning conçues pour créer et entraîner des réseaux de neurones (p.ex., Perceptron multicouche, réseaux convolutifs, réseaux récurrents) ;
- mettre en place des applications intégrant des réseaux de neurones et savoir évaluer leurs performances en faisant preuve d'esprit critique.
- définir et utiliser correctement les notions de théorie des graphes du cours (graphe « orienté/non orienté », « arête/arc », « chaîne/chemin », « cycle/circuit», « biparti », « complet », « simple », « eulérien », etc.) et les appliquer à la caractérisation d'un graphe donné ;
- passer d'une représentation d'un graphe à une autre (graphique, matricielle, par tableaux et listes) et discuter les avantages et inconvénients des représentations informatiques classiques ;
- décrire les problèmes classiques de la théorie des graphes et énoncer, appliquer et discuter le fonctionnement des algorithmes de résolution étudiés :
    - parcours de graphes (en largeur ou en profondeur et leurs applications : tri topologique, composantes connexes/fortement connexes), algorithme de Tarjan,
    - arbres et arborescences optimaux (arbres recouvrants de poids minimal, algorithme de Kruskal et de Prim, arborescences recouvrantes de poids minimal, algorithme de Chu-Liu, chaînes et chemins de section optimale),
    - plus courts chemins (algorithmes de Bellman-Ford et de Dijkstra, de Johnson, de Floyd-Warshall, ...),
    - ordonnancement (tri topologique, graphe potentiels-tâches, chemin critique, applications et extensions),
    - flots dans les réseaux (flot de valeur maximale, coupe de capacité minimale, algorithme de Ford-Fulkerson, flot à coût minimum, algorithme de Busacker et Gowen, variantes (problèmes de transbordement) et applications à des problèmes particuliers : couplages, postier chinois, ... ;
- modéliser une situation donnée à l'aide d'un graphe, identifier le(s) problème(s) de graphes à résoudre, établir une démarche de résolution, choisir l'algorithme étudié le plus adapté à la résolution de chaque problème et, selon les cas, les résoudre en appliquant la démarche retenue (ce point constitue l'un des objectifs majeurs du cours) ;
- comprendre et appliquer des algorithmes de graphes présentés sous forme d'un pseudocode, discuter leur complexité ;
- développer ou adapter des pseudocodes pour la résolution de problèmes proches de ceux étudiés ;
- mettre en oeuvre et tester des algorithmes étudiés ou développés et les appliquer à la résolution de problèmes d'optimisation ou d'aide à la décision.

5. Contenu et formes d'enseignement

Apprentissage par réseaux de neurones artificiels

Les réseaux de neurones artificiels et l'apprentissage profond sont parmi les sous-domaines les plus actifs de l'apprentissage automatique et de l'intelligence artificielle à l'heure actuelle. Les techniques de Deep Learning sont basées sur les réseaux neuronaux. Elles sont au cœur d'une large gamme d'applications applications impressionnantes, allant de la reconnaissance d'objets dans les images, de la traduction automatique, au jeu de Go. Cette unité se concentre sur les principes de fonctionnement des réseaux de neurones, leur implémentation (en Python), ainsi que leur entraînement et leur utilisation. Les étudiant-e-s apprendront les concepts fondamentaux des réseaux de neurones et du Deep Learning et développeront une bonne compréhension de la mise en place de solutions et d'applications intégrant ces algorithmes.

Forme(s) d'enseignement : Cours, Laboratoire

Graphes et réseaux

Les graphes offrent un outil à la fois visuel, intuitif et puissant pour la modélisation et l'optimisation de nombreux problèmes d'aide à la décision. Dans ce cours les étudiant-e-s seront familiarisé-e-s aux notions et concepts fondamentaux de la théorie des graphes ainsi qu'aux algorithmes classiques de résolution de problèmes tels que les recouvrements optimaux, les plus courts chemins ou les flots. Des exercices de mise en situation permettront également aux étudiant-e-s d'entraîner et d'améliorer leur capacité à la modélisation mathématique et informatique de cas concrets.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Apprentissage par réseaux de neurones artificiels (ARN) [poids: 90]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Graphes et réseaux (GRE) [poids: 150]

Note finale = moyenne cours x 0.4 + moyenne laboratoire x 0.1 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{90 \times ARN + 150 \times GRE}{240}$$

7. Modalités de remédiations

☒ Pas de remédiation
☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Apprentissage par réseaux de neurones artificiels

- Bishop, C. M. (1995). Neural networks for pattern recognition. Oxford University Press.
- Chollet, F. (2018). Deep learning with Python (Vol. 361). New York: Manning.
- Géron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media.
- Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning. Cambridge: MIT Press.

Graphes et réseaux

- Ravindra K. Ahuja, Thomas L. Magnanti, James B. Orlin, Network Flows: Theory, Algorithms, and Applications, 1993, Prentice Hall.
- Thomas Cormen, Charles Leiserson, Ronald Rivest, Clifford Stein, Introduction à l'algorithmique, 2002, Sciences sup, Dunod, Paris.
- Reinhard Diestel, Graph Theory, 2012, Springer-Verlag, Heidelberg.
- Michel Gondran, Michel Minoux, Graphes et algorithmes, 2009, Lavoisier, Cachan.
- Jon Kleinberg, Éva Tardos, Algorithm Design, 2005, Pearson.
- Robert Sedgewick, Kevin Wayne , Algorithms, 2011, Addison-Wesley.

10. Enseignants

Responsable du module : Andres Perez Uribe

Unité	Responsable
Apprentissage par réseaux de neurones artificiels	Andres Perez Uribe
Graphes et réseaux	Jean-François Hêche

Technologies Cloud et Web

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Technologies Cloud et Web
Code : TCW
Année académique : 2025-2026
Type de formation : Bachelor

Niveau

☐ Module de base
☒ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
CLD	90	Cloud Computing
WEB	150	Technologies web

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
CLD	Cours						32			
	Laboratoire						32			
WEB	Cours						48			
	Laboratoire						48			

2. Organisation

Crédits ECTS : 8
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☐ Avoir validé les modules : Néant
☒ Avoir suivi ou suivre en parallèle les modules : Bases de données et applications internet (BDA)
☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet de maîtriser les concepts de base liés à la création d'applications Web déployées sur le cloud : programmation d'applications Web en HTML/CSS/JavaScript et déploiement automatisé d'applications évolutives sur le cloud.

5. Contenu et formes d'enseignement

Cloud Computing

Choisir pour une application à déployer dans le cloud le modèle de service et le modèle de déploiement appropriés. Concevoir une architecture et déployer une application distribuée évolutive sur un cloud Infrastructure-en-tant-que-Service et Plateforme-en-tant-que-Service. Choisir pour une application le modèle de stockage cloud approprié. Connaître les logiciels Open Source pour mettre en oeuvre un cloud privé.

Forme(s) d'enseignement : Cours, Laboratoire

Technologies web

Cette unité est fortement orientée vers la pratique : l'objectif est d'appliquer les principes et les technologies présentés en développant des applications Web de plus en plus conséquentes. Dans une première phase, des exercices guidés sont utilisés pour solidifier les connaissances des étudiant-e-s relatives au langage et aux librairies de base. Dans une deuxième phase, les étudiant·e·s améliorent une application et explorent des libraires avancées. Il s'agit ici de faire preuve d'imagination pour proposer des fonctionnalités innovantes.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Cloud Computing (CLD) [poids: 90]

Note finale = moyenne cours x 0.67 + moyenne laboratoire x 0.33

Technologies web (WEB) [poids: 150]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{150 \times WEB + 90 \times CLD}{240}$$

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Cloud Computing

- Thomas Erl, Zaigham Mahmood, Ricardo Puttini, *Cloud Computing: Concepts, Technology & Architecture*, 2013-05-16, Prentice Hall.
- Bill Wilder, *Cloud Architecture Patterns*, 2012-09, O'Reilly Media.
- Jurg van Vliet, *Flavia Paganelli, Programming Amazon EC2*, 2011-02, O'Reilly Media.
- Lucas Carlson, *Programming for PaaS, 2013-07*, O'Reilly Media.
- Mathieu Zarouk, *Cloud Computing - Maîtrisez la plate-forme AWS - Amazon Web Services*, 2013-01-09, Eni.

Technologies web

- "Eloquent Javascript", Marijn Haverbeke
- "JavaScript: The Good Parts", Douglas Crockford
- "Web Development with Node and Express", Ethan Brown

10. Enseignants

Responsable du module : Bertil Chapuis

Unité	Responsable
Cloud Computing	Marcel Graf
Technologies web	Bertil Chapuis

Architectures multi-tiers (Multitiered architectures)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Architectures multi-tiers (Multitiered architectures)
Code	:	AMT
Année académique	:	2026-2027
Type de formation	:	Bachelor

Niveau

☐ Module de base
☒ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
AMT	150	Applications multi-tiers

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
AMT	Cours								48	
	Laboratoire								48	

2. Organisation

Crédits ECTS	:	5
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☒ Avoir validé les modules : Bases de données et applications internet (BDA), Programmation orienté objet (POO)

☐ Avoir suivi ou suivre en parallèle les modules : Néant

☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet aux étudiant·e·s de développer une compréhension approfondie des architectures d'applications multi-tiers. Les étudiant·e·s acquerront des compétences clés en matière de conception, de développement, et d'optimisation d'applications à travers l'étude des principes généraux des architectures multi-tiers, l'utilisation de technologies spécifiques telles que Jakarta EE, et l'application des pratiques de génie logiciel comme l'intégration et la livraison continues.

5. Contenu et formes d'enseignement

Applications multi-tiers

Le premier objectif de cette unité est de présenter les concepts généraux d'une architecture multi-tiers (principes qui s'appliquent donc aux différentes technologies, que ce soit dans le monde Java, dans le monde C# ou dans d'autres). Le rôle d'un ORM, l'injection de dépendances, le modèle de conception MVC, les APIs de type RPC et les APIs de type REST seront notamment étudiés dans ce cadre.

Le deuxième objectif est d'étudier une implémentation particulière de ces concepts. A cette fin, les technologies qui composent la plateforme Jakarta EE ou ASP.NET Core seront étudiées et mises en pratique.

Le troisième objectif est d'étudier les pratiques de génie logiciel qui sont recommandées lors du développement d'applications multi-tiers, telles que la mise en œuvre de chaînes d'intégration et de livraison en continu et l'utilisation de tests automatisés.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Applications multi-tiers (AMT) [poids: 150]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité AMT

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Applications multi-tiers

- https://microprofile.io/
- https://jakarta.ee/
- https://spring.io/
- https://quarkus.io/
- https://helidon.io/

10. Enseignants

Responsable du module : Bertil Chapuis

Unité	Responsable
Applications multi-tiers	Bertil Chapuis

Développement mobile et sécurité logicielle

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Développement mobile et sécurité logicielle
Code : DML
Année académique : 2026-2027
Type de formation : Bachelor

Niveau

☐ Module de base
☒ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
DAA	120	Développement d'applications Android
SLH	120	Sécurité logicielle haut niveau

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
DAA	Cours								32	
	Laboratoire								32	
SLH	Cours								32	
	Laboratoire								32	

2. Organisation

Crédits ECTS : 8
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☒ Avoir validé les modules : Programmation orienté objet (POO)

☒ Avoir suivi ou suivre en parallèle les modules : Bases de données et applications internet (BDA), Systèmes d'exploitation et concurrence (SEC)

☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet d'acquérir les compétences et connaissances nécessaires à la conception de solutions logicielles sécurisées ainsi qu'au développement mobile sur Android, au travers de deux unités :

- Développement android (DVA) introduit les concepts et outils nécessaires au développement d'applications mobiles sur la plateforme Android. Certaines problématiques en lien avec la sécurité et la protection des données seront traitées dans le cadre de cette unité ;
- Sécurité logicielle haut niveau (SLH) présente les principales attaques impactant les logiciels ainsi que les mécanismes de défense les plus courants. Cette unité est également axée sur les bonnes pratiques à mettre en œuvre pour assurer un développement sécurisé.

5. Contenu et formes d'enseignement

Développement d'applications Android

Cette unité permet l'acquisition des compétences nécessaires au développement d'applications mobiles sur Android. La première partie du semestre est dédiée à l'étude et à la prise en main des composants logiciels ainsi que de l'architecture à mettre en place pour réaliser des applications mobiles natives sur Android. La seconde partie du semestre couvre certains aspects fondamentaux du développement mobile, tels que la gestion de l'asynchronisme permettant de communiquer avec un serveur, la mise en place de tests automatisés jusqu'à la distribution de l'application.

Forme(s) d'enseignement : Cours, Laboratoire

Sécurité logicielle haut niveau

Cette unité présente tout d'abord les attaques les plus courantes qui apparaissent dans les logiciels et les étudiant·e·s sont amené·e·s à les implémenter et à connaitre les mécanismes de défense les plus courants. Par la suite, le cours est axé sur les bonnes pratiques de développement sécurisé. Ceci se fait principalement au travers du langage de programmation Rust mais des analyses des autres langages sont proposées tout au long du cours.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Développement d'applications Android (DAA) [poids: 120]

Note finale = moyenne cours x 0.67 + moyenne laboratoire x 0.33

Sécurité logicielle haut niveau (SLH) [poids: 120]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{120 \times DAA + 120 \times SLH}{240}$$

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Développement d'applications Android

- Android Programming: The Big Nerd Ranch Guide - K. Marsicano, B. Gardner, B. Phillips, and C. StewartBig Nerd Ranch - 4th Edition – August 2019
- The Busy Coder's Guide to Android Development - Mark L. Murphy (https://commonsware.com/Android/) - Série de livres sur le développement Android

Sécurité logicielle haut niveau

- Mitre, "Common Weakness Enumeration", https://cwe.mitre.org/
- OWASP, "Cheat sheets", https://cheatsheetseries.owasp.org/

10. Enseignants

Responsable du module : Fabien Dutoit

Unité	Responsable
Développement d'applications Android	Fabien Dutoit
Sécurité logicielle haut niveau	Alexandre Duc

Simulation, optimisation et paradigmes de programmation

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Simulation, optimisation et paradigmes de programmation
Code : SOP
Année académique : 2026-2027
Type de formation : Bachelor

Niveau

☐ Module de base
☒ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
PLP	120	Paradigmes et langages de programmation
SIO	120	Simulation et optimisation

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
PLP	Cours								32	
	Laboratoire								32	
SIO	Cours								32	
	Laboratoire								32	

2. Organisation

Crédits ECTS : 8
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☐ Avoir validé les modules : Néant

☒ Avoir suivi ou suivre en parallèle les modules : Graphes et réseaux de neurones (GRN), Programmation orienté objet (POO)

☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

À l'issue de ce module, l'étudiant·e sera capable de :

- élaborer des algorithmes manipulant des structures linéaires avec un langage fonctionnel ;
- élaborer des algorithmes en utilisant des fonctions d'ordre supérieur ;
- élaborer des algorithmes récursifs avec un langage fonctionnel ;
- définir la grammaire EBNF d'un langage simple ;
- écrire les fichiers de définition qui permettent de générer les programmes d'analyse lexicale et syntaxique d'un tel langage ;
- écrire un interpréteur ou un compilateur simple ;
- définir et implémenter une machine abstraite ;
- expliquer et implémenter différentes méthodes de gestion de la mémoire ainsi que le fonctionnement d'une pile d'exécution ;
- expliquer les notions d'heuristiques et d'approximations pour des problèmes d'optimisation combinatoire difficiles ;
- illustrer les notions d'heuristiques constructives et d'amélioration pour des problèmes classiques (coloration de graphes, voyageur de commerce, ...) ;
- modéliser des problèmes de décision sous forme de programmes linéaires ou de programmes linéaires mixtes ;
- résoudre graphiquement des problèmes linéaires à deux variables ;
- résoudre des programmes linéaires ou des programmes linéaires mixtes à l'aide d'un langage de modélisation (GMPL) et d'un solveur (GLPK/GLPSOL) ;
- motiver l'utilisation de la simulation informatique en illustrant ses avantages au travers d'exemples ;
- adapter les méthodes classiques (fonctions inverses, mélanges, acception-rejet...) à la génération de réalisations de variables aléatoires simples ;
- expliquer le principe de la simulation de Monte-Carlo et le fonctionnement des techniques d'acceptation-rejet et de moyenne d'un échantillon ;
- implémenter des heuristiques classiques, de comparer leurs performances et d'étudier leur complexité ;
- modéliser des problèmes linéaires en utilisant un langage de modélisation puis de les résoudre à l'aide d'un solveur ;
- réaliser une simulation de Monte-Carlo et d'étudier la convergence des estimateurs des performances étudiées et d'établir des intervalles de confiance pour ces estimateurs.

5. Contenu et formes d'enseignement

Paradigmes et langages de programmation

Le cours aborde les concepts principaux des langages de programmation et de la compilation de ces langages en illustrant ces concepts de manière pratique à l'aide d'un langage fonctionnel.

Forme(s) d'enseignement : Cours, Laboratoire

Simulation et optimisation

Les méthodes d'optimisation, aussi bien exactes qu'approchées, constituent un élément central dans les techniques d'aide à la décision. Les études de simulation sont, elles, une étape quasi incontournable lors du déploiement d'un nouveau système ou avant la réorganisation d'un système existant.

Dans cette unité les étudiant·e·s aborderont divers aspects de la simulation informatique stochastique ainsi que de la modélisation de systèmes complexes et de la résolution de problèmes d'optimisation difficiles. Plusieurs approches de résolution, illustrées sur des exemples, seront présentées et développées.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Paradigmes et langages de programmation (PLP) [poids: 120]

Note finale = moyenne cours x 0.3 + moyenne laboratoire x 0.2 + moyenne examen x 0.5

Simulation et optimisation (SIO) [poids: 120]

Note finale = moyenne cours x 0.67 + moyenne laboratoire x 0.33

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{120 \times SIO + 120 \times PLP}{240}$$

7. Modalités de remédiations

☒ Pas de remédiation
☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Paradigmes et langages de programmation

- Miran Lipovaca, Learn You a Haskell for Great Good !, No Starch Press.
- Peter Sestoft, Programming Language Concepts, Springer.

Simulation et optimisation

- Johann Dréo, Alain Pétrowski, Patrick Siarry, Eric Taillard, Métaheuristiques pour l'optimisation difficile, 2003, Eyrolles, Paris.
- Sheldon M. Ross, A course in Simulation, 1990, Prentice Hall.
- Jean-François Hêche, Thomas M. Liebling, Dominique de Werra, Recherche opérationnelle pour ingénieurs, Volume 2, 2003, PPUR presses polytechniques, Lausanne.
- Reuven Y. Rubinstein, Dirk P. Kroese, Simulation and the Monte Carlo Method, 2nd Edition, 2008, John Wiley & Sons.

10. Enseignants

Responsable du module : Jean-François Hêche

Unité	Responsable
Paradigmes et langages de programmation	Bertil Chapuis
Simulation et optimisation	Jean-François Hêche

Systèmes et données distribués

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom : Systèmes et données distribués
Code : SDD
Année académique : 2026-2027
Type de formation : Bachelor

Niveau

☐ Module de base
☒ Module d'approfondissement
☐ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
MAC	90	Méthodes d'accès aux données
SDR	90	Systèmes distribués et répartis

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
MAC	Cours								32	
	Laboratoire								32	
SDR	Cours								32	
	Laboratoire								32	

2. Organisation

Crédits ECTS : 6
Langue(s) principale(s) d'enseignement : Français

3. Prérequis

☒ Avoir validé les modules : Bases de données et applications internet (BDA), Programmation orienté objet (POO), Systèmes d'exploitation et concurrence (SEC)

☐ Avoir suivi ou suivre en parallèle les modules : Néant

☐ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Ce module permet aux étudiant·e·s de se familiariser avec les concepts et les techniques essentiels pour concevoir et interagir avec des systèmes et des données distribués. À travers l'étude des systèmes répartis, les étudiant·e·s apprendront à gérer les défis inhérents aux environnements distribués, tels que la fiabilité et la synchronisation. En parallèle, ils acquerront des compétences en matière d'accès aux données, en se concentrant sur les techniques de stockage et d'analyse de données à grande échelle (big data, map-reduce, NoSql, etc.) et les modèles de recherche d'information et d'indexation. Les travaux pratiques viendront renforcer ces compétences en permettant aux étudiant·e·s de mettre en œuvre des applications concrètes.

5. Contenu et formes d'enseignement

Méthodes d'accès aux données

Cette unité est une introduction aux techniques avancées et récentes d'accès aux données massives. Le but est de présenter les limitations avec les modèles classiques de base de données relationnelles et d'étudier les possibilités de modélisation de données au-delà des bases de données classiques. Dans cet objectif, le cours se focalise sur les besoins des applications utilisant intensivement les données (*Data Intensive Applications*). Plus particulièrement, nous allons étudier les bases de données NoSQL, l'indexation vectoriel employée par les moteurs de recherche textuelle et le modèle map-reduce ainsi que l'analyse des données big data avec Spark.

Forme(s) d'enseignement : Cours, Laboratoire

Systèmes distribués et répartis

Aujourd'hui, la majorité des solutions informatiques font partie d'un système distribué. L'objectif de cette unité est de comprendre les notions fondamentales associées à la conception d'applications réparties, telles que la fiabilité, l'exclusion mutuelles, les horloges logiques ou l'élection d'un leader. Des laboratoires permettront de concevoir, tester et analyser des systèmes répartis.

Forme(s) d'enseignement : Cours, Laboratoire

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0

Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Méthodes d'accès aux données (MAC) [poids: 90]

Note finale = moyenne cours x 0.67 + moyenne laboratoire x 0.33

Systèmes distribués et répartis (SDR) [poids: 90]

Note finale = moyenne cours x 0.67 + moyenne laboratoire x 0.33

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

$$\textbf{Note finale} = \frac{90 \times SDR + 90 \times MAC}{180}$$

7. Modalités de remédiations

☒ Pas de remédiation

☐ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Méthodes d'accès aux données

1. "Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems", Martin Kleppmann,. Mars 2017.
2. "Next Generation Databases NoSQL and Big Data", Guy Harrison, 2015.
3. "Database Systems, The Complete Book", 2nd Edition, Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom, 2008.

Systèmes distribués et répartis

Distributed Systems: Principles and Paradigms, par Andrew S.Tanenbaum et Maarten Van Steen, Prentice Hall, 2007.

Reliable and secure distributed programming, par Christian Cachin, Rachid Guerraoui, et Luís Rodrigues, Springer, 2011.

10. Enseignants

Responsable du module : Bertil Chapuis

Unité	Responsable
Méthodes d'accès aux données	Nastaran Fatemi
Systèmes distribués et répartis	Bertil Chapuis

Innovation Crunch Time
(Innovation Crunch Time)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Innovation Crunch Time (Innovation Crunch Time)
Code	:	CRUNCH
Année académique	:	2026-2027
Type de formation	:	Bachelor

Niveau

☐ Module de base
☐ Module d'approfondissement
☐ Module avancé
☒ Module spécialisé

Type

☐ Module principal
☐ Module lié à un module principal
☒ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
CRH	60	Innovation Crunch Time

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
CRH	Projet									50

2. Organisation

Crédits ECTS	:	2
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☐ Avoir validé les modules : Néant
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☒ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

- Participer à un challenge d'innovation en équipe pluridisciplinaires
- Collaborer avec des personnes provenant d'autres horizons (techniques - économiques)
- Travailler sur un sujet proposé par organisation réelle (entreprise, administration publique, association, ...)

5. Contenu et formes d'enseignement

Innovation Crunch Time

En collaboration avec l'UTBM (Université de technologie de Belfort Montbéliard), le CRUNCH est un exercice pédagogique novateur proposé aux étudiant-e-s de 3ème année de Bachelor de toutes filières d'ingénierie et de gestion de la HEIG-VD.

Par équipes interdisciplinaires et sur 5 jours consécutifs, les étudiant-e-s travaillent sur des défis d'innovation, proposés par des partenaires externes (entreprises, associations, collectivités publiques, …) ou des défis entrepreneuriaux, portés par des collègues étudiant-e-s.

Les équipes interdisciplinaires sont formées et appliquent une démarche de *design thinking* en mode sprint, centrée sur l'utilisateur. Encadré par des professeur-e-s sur la méthodologie proposée et par des experts métiers (pour les parties protypage et pitch) cet exercice pédagogique place les étudiant-e-s dans une situation réaliste de travail en équipe sur un sujet donné.

Forme(s) d'enseignement : Projet

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Innovation Crunch Time (CRH) [poids: 60]

Note finale = moyenne projet x 1

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité CRH

7. Modalités de remédiations

☐ Pas de remédiation
☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

Cet exercice permettra la collaboration avec les étudiants de toutes les autres filières de l'école.

9. Bibliographie

Innovation Crunch Time

- Brown, T. (2008). Design thinking. Harvard business review, 86(6), 84.
- Brown, T. (2010). L'esprit design: le design thinking change l'entreprise et la stratégie. Pearson Education France.
- Mauborgne, R., & Kim, W. C. (2015). Stratégie Océan Bleu: Comment créer de nouveaux espaces stratégiques. Pearson.
- Knapp, J., Zeratsky, J., & Kowitz, B. (2017). Sprint: Comment résoudre les problèmes et trouver de nouvelles idées en cinq jours. Editions Eyrolles.
- Osterwalder, A., Pigneur, Y., Bernarda, G., & Smith, A. (2015). La méthode Value proposition design. Paris: Pearson France.

10. Enseignants

Responsable du module : Jean-Marc Seydoux

Unité	Responsable
Innovation Crunch Time	Nathalie Nyffeler

Travail de Bachelor (Bachelor thesis)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Travail de Bachelor (Bachelor thesis)
Code	:	TB
Année académique	:	2026-2027
Type de formation	:	Bachelor

Niveau

☐ Module de base
☐ Module d'approfondissement
☒ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
TB-TIC	450	Travail de Bachelor pour TIC

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
TB-TIC	Projet									x

2. Organisation

Crédits ECTS	:	15
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☐ Avoir validé les modules : Néant
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☒ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Concevoir et réaliser un produit, une application ou un service (les points ci-dessous peuvent varier selon la nature du projet) :

- analyser et affiner un cahier des charges ;
- effectuer une décomposition fonctionnelle ;
- trouver des solutions techniques ;
- effectuer des choix parmi les solutions possibles et les justifier ;
- rédiger des spécifications techniques détaillées ;
- planifier et spécifier en détail la réalisation et la validation ;
- modéliser / simuler ;
- effectuer la conception de détail (calculs, dessins, schémas, programmes, etc.) ;
- développer la solution informatique ;
- intégrer les composants ou produits requis ;
- intégrer la conception en vue du test.

Valider, documenter et promouvoir le développement d'un produit, une application ou un service (les points ci-dessous peuvent varier selon la nature du projet) :

- vérifier / mesurer / tester / caractériser ;
- analyser et critiquer les résultats obtenus ;
- tenir un journal de travail ;
- rédiger un rapport de projet ;
- rédiger une documentation technique ;
- présenter oralement son projet de manière concise.

5. Contenu et formes d'enseignement

Travail de Bachelor pour TIC

Le travail de Bachelor vise à mettre en pratique les connaissances et compétences acquises au cours de la formation, en confrontant l'étudiant·e à un travail d'ingénieur·e dans la conception et la réalisation d'un projet d'ampleur.

Forme(s) d'enseignement : Projet

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Travail de Bachelor pour TIC (TB-TIC) [poids: 450]

Note finale = moyenne projet x 1

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité TB-TIC

7. Modalités de remédiations

☐ Pas de remédiation
☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Travail de Bachelor pour TIC

10. Enseignants

Responsable du module : Pier Donini

Unité	**Responsable**
Travail de Bachelor pour TIC	Pier Donini

Liste des descriptifs de module actuellement indisponibles

- Projet de groupe
- Enseignements à choix IL

Travail de Bachelor (Bachelor thesis)

Domaine	Ingénierie et Architecture
Filière	Informatique et systèmes de communication
Orientation	Informatique logicielle (ISCL)
Mode	Plein temps

1. Intitulé du module

Nom	:	Travail de Bachelor (Bachelor thesis)
Code	:	TB
Année académique	:	2026-2027
Type de formation	:	Bachelor

Niveau

☐ Module de base
☐ Module d'approfondissement
☒ Module avancé
☐ Module spécialisé

Type

☒ Module principal
☐ Module lié à un module principal
☐ Module complémentaire

Caractéristique

☒ Module obligatoire

En cas d'échec définitif à un module défini comme obligatoire pour acquérir le profil de formation correspondant, l'étudiant est exclu de la filière, voire du domaine si le règlement de filière le précise conformément à l'article 25 du règlement sur la formation de base (bachelor et master) en HES-SO.

Organisation temporelle

Les tables contiennent le nombre de périodes par unité et par type d'enseignement. Les valeurs pour le volume de travail correspondent au nombre d'heures totales à fournir par l'étudiant.

Abréviation	Volume	Unité
TB-TIC	450	Travail de Bachelor pour TIC

Semestre		E1	S1	S2	E2	S3	S4	E3	S5	S6
TB-TIC	Projet									x

2. Organisation

Crédits ECTS	:	15
Langue(s) principale(s) d'enseignement	:	Français

3. Prérequis

☐ Avoir validé les modules : Néant
☐ Avoir suivi ou suivre en parallèle les modules : Néant
☒ Pas de prérequis

4. Compétences visées / Objectifs généraux d'apprentissage

Concevoir et réaliser un produit, une application ou un service (les points ci-dessous peuvent varier selon la nature du projet) :

- analyser et affiner un cahier des charges ;
- effectuer une décomposition fonctionnelle ;
- trouver des solutions techniques ;
- effectuer des choix parmi les solutions possibles et les justifier ;
- rédiger des spécifications techniques détaillées ;
- planifier et spécifier en détail la réalisation et la validation ;
- modéliser / simuler ;
- effectuer la conception de détail (calculs, dessins, schémas, programmes, etc.) ;
- développer la solution informatique ;
- intégrer les composants ou produits requis ;
- intégrer la conception en vue du test.

Valider, documenter et promouvoir le développement d'un produit, une application ou un service (les points ci-dessous peuvent varier selon la nature du projet) :

- vérifier / mesurer / tester / caractériser ;
- analyser et critiquer les résultats obtenus ;
- tenir un journal de travail ;
- rédiger un rapport de projet ;
- rédiger une documentation technique ;
- présenter oralement son projet de manière concise.

5. Contenu et formes d'enseignement

Travail de Bachelor pour TIC

Le travail de Bachelor vise à mettre en pratique les connaissances et compétences acquises au cours de la formation, en confrontant l'étudiant·e à un travail d'ingénieur·e dans la conception et la réalisation d'un projet d'ampleur.

Forme(s) d'enseignement : Projet

6. Modalités d'évaluation et de validation

Seuil de compensation entre unités du module : 3.0
Seuil de répétition du module : 4.0

Le calcul de la note finale de chaque unité est détaillé ci-après. Pour chaque unité, sa pondération est indiquée entre crochets après son nom.

Travail de Bachelor pour TIC (TB-TIC) [poids: 450]

Note finale = moyenne projet x 1

Note finale du module

La note du module est calculée à partir des notes des différentes unités du module.

Note finale = note de l'unité TB-TIC

7. Modalités de remédiations

☐ Pas de remédiation
☒ Remédiation possible uniquement lors du premier suivi du module

8. Remarques

9. Bibliographie

Travail de Bachelor pour TIC

10. Enseignants

Responsable du module : Pier Donini

Unité	**Responsable**
Travail de Bachelor pour TIC	Pier Donini