

# Mastère Spécialisé

## Développement de Logiciels



## Mastère Spécialisé Développement de Logiciels

- » Modalité: en ligne
- » Durée: 12 mois
- » Qualification: TECH Université Technologique
- » Intensité: 16h/semaine
- » Horaire: à votre rythme
- » Examens: en ligne

Accès au site web: [www.techtitute.com/fr/informatique/master/master-developpement-logiciels](http://www.techtitute.com/fr/informatique/master/master-developpement-logiciels)

# Sommaire

01

Présentation

---

*page 4*

02

Objectifs

---

*page 8*

03

Compétences

---

*page 14*

04

Structure et contenu

---

*page 18*

05

Méthodologie

---

*page 32*

06

Diplôme

---

*page 40*

# 01

# Présentation

Pour participer à l'un des domaines les plus prometteurs du secteur des technologies de l'information, le professionnel doit être formé scientifiquement et technologiquement, ainsi que préparé à faire face, de manière efficace, aux défis qui se présentent dans l'exercice professionnel de l'ingénierie logicielle. Ce Mastère Spécialisé est orienté pour atteindre une haute maîtrise de Développement de Logiciels, à travers les dernières avancées et développements dans ce domaine, au moyen d'une méthodologie d'étude d'un impact maximal et d'une flexibilité extraordinaire.



“

*Acquérir les connaissances les plus complètes en ingénierie logicielle, dans la formation la plus actualisée du marché éducatif en ligne et commencer à travailler sur les développements dans ce domaine professionnel dynamique"*

Avec l'avancée des nouvelles technologies, les Logiciels sont devenus un élément extrêmement important dans le monde d'aujourd'hui. Ces dernières années, la nécessité de pouvoir développer des produits Logiciels présentant les fonctionnalités et la qualité appropriées, tout en respectant les délais et le budget établis, est devenue évidente.

Ce programme est destiné aux personnes désireuses d'atteindre un niveau de connaissances plus élevé en matière de Développement Logiciel. L'objectif principal est de permettre aux étudiants d'appliquer les connaissances acquises dans ce Mastère Spécialisé dans le monde réel, dans un environnement de travail qui reproduit les conditions que l'on peut trouver dans le futur, de manière rigoureuse et réaliste.

Saisissez l'occasion de suivre cette formation dans un format 100% en ligne, sans avoir à renoncer à ses obligations, et en facilitant le retour à l'université. actualise ses connaissances et atteignant qualification de Mastère Spécialisé pour continuer à se développer personnellement et professionnellement.

Vous acquerez des connaissances approfondies dans le domaine de l'ingénierie logicielle, mais aussi dans le domaine de l'informatique et de la structure des ordinateurs, y compris les bases mathématiques, statistiques et physiques essentielles en ingénierie.

Saisissez l'occasion et suivez cette formation dans un format 100% en ligne, sans devoir renoncer à vos obligations, et en facilitant votre retour à l'université. Mettez à jour vos connaissances et obtenez votre Mastère Spécialisé pour continuer à vous épanouir personnellement et professionnellement.

Ce **Mastère Spécialisé en Développement de Logiciels** contient le programme éducatif le plus complet et le plus actuel du marché. Ses principales caractéristiques sont:

- » Développement de 100 scénarios simulés présentés par des experts en Développement de Logiciels
- » Les contenus graphiques, schématiques et éminemment pratiques avec lesquels ils sont conçus fournissent des informations scientifiques et pratiques sur le Développement de Logiciels
- » Actualités sur les derniers développements en matière de Développement de Logiciels
- » Contient des exercices pratiques où le processus d'auto-évaluation peut être réalisé pour améliorer l'apprentissage
- » Système d'apprentissage interactif fondé sur la méthode des cas et son application à la vie réelle
- » Tout cela sera complété par des cours théoriques, des questions à l'expert, des forums de discussion sur des sujets controversés et un travail de réflexion individuel
- » Disponibilité des contenus à partir de tout appareil fixe ou portable doté d'une connexion internet



*Ce programme vous permettra d'apprendre la structure de base d'un ordinateur et de ses logiciels, afin d'améliorer vos compétences"*

“

*Apprenez tout ce dont vous avez besoin pour travailler avec les langages de programmation en toute sécurité, en intégrant à vos connaissances l'interprétation et la conception d'algorithmes de base pour travailler en programmation"*

Le corps enseignant comprend des professionnels du monde du Développement de Logiciels, qui apportent l'expérience de leur travail à cette formation, ainsi que des spécialistes reconnus issus de grandes entreprises et d'universités prestigieuses.

Grâce à son contenu multimédia développé avec les dernières technologies éducatives, les spécialistes bénéficieront d'un apprentissage situé et contextuel. Ainsi, ils se formeront dans un environnement simulé qui leur permettra d'apprendre en immersion et de s'entraîner dans des situations réelles.

La conception de ce programme est basée sur l'Apprentissage par Problèmes. Ainsi le professionnel devra essayer de résoudre les différentes situations de pratique professionnelle qui se présentent à lui tout au long du programme. Pour ce faire, le professionnel sera assisté d'un système vidéo interactif innovant créé par des experts reconnus en Développement de Logiciels une grande expérience de l'enseignement.

*Une formation qui vous permettra de comprendre le fonctionnement d'un programme informatique et d'intervenir sur tous les éléments essentiels d'un programme informatique.*

*Apprenez à connaître les derniers systèmes de données sur le marché, apprenez à concevoir des algorithmes avancés et tous les aspects qu'un professionnel hautement compétent doit maîtriser.*



# 02

## Objectifs

L'objectif de cette formation est de fournir aux professionnels travaillant dans le développement logiciel les connaissances et les compétences nécessaires pour exercer leur activité en utilisant les protocoles et les techniques les plus avancés actuellement disponibles. Grâce à une approche de travail entièrement adaptable à l'étudiant, ce Mastère Spécialisé vous amènera progressivement à acquérir les compétences qui vous pousseront vers un niveau professionnel beaucoup plus élevé.



```
!!$_GET[type]) echo "current";  
type=1#text_margin">  
</div>  
ang'] == 'rus') ed
```

“

*Vous plongerez dans le domaine du calcul et de la structure des ordinateurs, sujets essentiels pour tout développeur de logiciels"*



## Objectifs généraux

---

- » Former scientifiquement et technologiquement, ainsi que préparer à la pratique professionnelle du génie Logiciel, le tout avec une formation transversale et polyvalente adaptée aux nouvelles technologies et aux innovations dans ce domaine
- » Obtenir des connaissances approfondies dans le domaine du génie Logiciel, mais aussi dans le domaine de l'informatique et de la structure des ordinateurs, y compris les bases mathématiques, statistiques et physiques essentielles en ingénierie

“

*Atteignez le niveau de connaissances que vous souhaitez et maîtrisez le développement logiciel grâce à cette formation de haut niveau”*





## Objectifs spécifiques

---

### Module 1. Principes fondamentaux de la programmation

- » Comprendre la structure de base d'un ordinateur, les Logiciels et les langages de programmation à usage général
- » Apprendre à concevoir et à interpréter des algorithmes, qui constituent la base nécessaire au développement de programmes informatiques
- » Comprendre les éléments essentiels d'un programme informatique, tels que les différents types de données, les opérateurs, les expressions, les instructions, les entrées/sorties et les instructions de contrôle
- » Comprendre les différentes structures de données disponibles dans les langages de programmes polyvalents, tant statiques que dynamiques, et acquérir les connaissances essentielles pour la manipulation des fichiers
- » Comprendre les différentes techniques de test des logiciels et l'importance de générer une bonne documentation en même temps qu'un bon code source
- » Apprendre les concepts de base du langage de Programmation C++, l'un des langages de programmation les plus utilisés dans le monde

### Module 2. Structure des données

- » Apprenez les principes fondamentaux de la programmation C++, notamment les classes, les variables, les expressions conditionnelles et les objets
- » Comprendre les types de données abstraits, les types de structures de données linéaires, les structures de données hiérarchiques simples et complexes et leur mise en œuvre en C++
- » Comprendre le fonctionnement des structures de données avancées autres que les structures habituelles
- » Comprendre la théorie et la pratique liées à l'utilisation des tas et des files d'attente prioritaires
- » Apprendre le fonctionnement des tables de *Hash*, en tant que types de données abstraites et fonctions
- » Comprendre la théorie des Graphes, ainsi que les algorithmes et concepts avancés des Graphes

### Module 3. Algorithme et complexité

- » Apprendre les principales stratégies de conception d'algorithmes, ainsi que les différentes méthodes et mesures de calcul d'algorithmes
- » Apprendre les principaux algorithmes de tri utilisés dans le développement de Logiciels
- » Comprendre le fonctionnement des différents algorithmes avec les arbres, *Heaps* et les Graphes
- » Comprendre le fonctionnement des algorithmes *Greedy*, leur stratégie et des exemples de leur utilisation dans les principaux problèmes connus. Nous apprendrons également l'utilisation des algorithmes gourmands sur les graphes
- » Nous apprendrons les principales stratégies de recherche de chemin minimum, avec l'approche des problèmes essentiels du domaine et des algorithmes pour leur résolution
- » Comprendre la technique du *Backtracking* et ses principales utilisations, ainsi que d'autres techniques alternatives

### Module 4. Bases de données

- » Apprenez les différentes applications et finalités des systèmes de bases de données, ainsi que leur fonctionnement et leur architecture
- » Comprendre le modèle relationnel, de sa structure et de ses opérations à l'algèbre relationnelle étendue
- » Apprenez en profondeur ce que sont les bases de données SQL, comment elles fonctionnent, la définition des données et la création de requêtes, des plus basiques aux plus avancées et complexes
- » Apprenez à concevoir des bases de données à l'aide du modèle entité-relationnel, à créer des diagrammes et à connaître les caractéristiques du modèle E-R étendu
- » Approfondir la conception des bases de données relationnelles, en analysant les différentes formes normales et les algorithmes de décomposition
- » Poser les bases pour comprendre le fonctionnement des bases de données NoSQL, ainsi que présenter la base de données MongoDB

### Module 5. Bases de données avancées

- » Présenter les différents systèmes de bases de données actuellement sur le marché
- » Apprendre l'utilisation de XML et des bases de données pour le web
- » Comprendre le fonctionnement des bases de données avancées telles que les bases de données parallèles et distribuées
- » Comprendre l'importance de l'indexation et de l'association dans les systèmes de bases de données
- » Comprendre le fonctionnement des systèmes de traitement et d'extraction transactionnels
- » Acquérir des connaissances relatives aux bases de données non relationnelles et à l'exploration de données

### Module 6. Conception d'algorithmes avancés

- » Approfondissez la conception avancée d'algorithmes, en analysant les algorithmes récursifs et de division et de conquête, ainsi qu'en effectuant des analyses amorties
- » Comprendre les concepts de la Programmation dynamique et les algorithmes pour les problèmes de PN
- » Comprendre le fonctionnement de l'optimisation combinatoire, ainsi que les différents algorithmes de randomisation et les algorithmes parallèles
- » Connaître et comprendre le fonctionnement des différentes méthodes de recherche locale et de recherche de candidats
- » Apprendre les mécanismes de vérification formelle des programmes et des programmes itératifs, notamment la logique du premier ordre et le système formel de Hoare
- » Apprenez le fonctionnement de certaines des principales méthodes numériques telles que la méthode de bisection, la méthode de Newton Raphson et la méthode de la sécante

## Module 7. Interaction Homme-machine

- » Acquérir des connaissances solides en matière d'interaction homme-machine et de création d'interfaces utilisables
- » Comprendre l'importance de la convivialité des applications et pourquoi il est nécessaire d'en tenir compte lors de la conception de nos Logiciels
- » Comprendre les différents types de diversité humaine, les limites qu'ils impliquent et comment adapter les interfaces en fonction des besoins spécifiques de chacun d'entre eux
- » Apprendre le processus de conception d'une interface, de l'analyse des besoins à l'évaluation, en passant par les différentes étapes intermédiaires nécessaires à la création d'une interface adaptée
- » Connaître les différentes directives d'accessibilité, les normes qui les établissent et les outils qui permettent de les évaluer
- » Comprendre les différentes méthodes d'interaction avec l'ordinateur, en utilisant des périphériques et des dispositifs

## Module 8. Programmation avancée

- » Approfondir les connaissances en Programmation, notamment en ce qui concerne la programmation orientée objet, et les différents types de relations entre les classes existantes
- » Apprendre les différents modèles de conception pour les problèmes orientés objet
- » Apprendre la Programmation événementielle et le développement d'interfaces utilisateur avec Qt
- » Acquérir les connaissances essentielles de la Programmation concurrente, des processus et des threads
- » Apprenez à gérer l'utilisation des threads et de la synchronisation, ainsi que la résolution des problèmes courants dans le cadre de la Programmation concurrente
- » Comprendre l'importance de la documentation et des tests dans le développement de Logiciels

## Module 9. Développement d'applications de réseau

- » Connaître les caractéristiques du langage de balisage HTML et son utilisation dans la création de sites web, ainsi que les feuilles de style CSS
- » Apprenez à utiliser le langage de Programmation JavaScript, orienté navigateur, et certaines de ses principales fonctionnalités
- » Comprendre les concepts de la Programmation orientée composants et de l'architecture des composants
- » Apprenez à utiliser le *Framework* pour *Frontend* Bootstrap pour la conception de sites Web
- » Comprendre la structure du modèle contrôleur-vue dans le développement de sites Web dynamiques
- » Connaître l'architecture orientée services et les bases du protocole HTTP

## Module 10. Ingénierie logicielle

- » Poser les bases du génie Logiciel et de la modélisation, en apprenant les principaux processus et concepts
- » Comprendre le processus Logiciel et les différents modèles pour son développement, y compris les technologies agiles
- » Comprendre l'ingénierie des exigences, leur développement, leur élaboration, leur négociation et leur validation
- » Apprenez la modélisation des exigences et les différents éléments tels que les scénarios, les informations, les classes d'analyse, le flux, le comportement et les modèles
- » Comprendre les concepts et les processus de la conception de Logiciels, en apprenant également l'architecture de conception, la conception au niveau des composants et la conception basée sur des modèles
- » Connaître les principales normes relatives à la qualité des Logiciels et à la gestion de projet

# 03

## Compétences

Après avoir passé les évaluations du Mastère Spécialisé en Développement de Logiciels, aura acquis les compétences professionnelles nécessaires pour réaliser un travail de qualité et pourra acquerra également de nouvelles compétences et techniques qui lui permettront de compléter les connaissances qu'il possède déjà.



“

*Augmentez vos compétences en matière de Développement de Logiciels et passez au niveau supérieur en tant que professionnel dans ce domaine en constante évolution"*



## Compétence générale

---

» Répondre aux besoins actuels dans le domaine de Développement de Logiciels

“

*Un programme exceptionnel par sa densité, son exhaustivité et la manière dont il est enseigné, qui vous permettra de progresser rapidement et efficacement”*





## Compétences spécifiques

---

- » Être capable de comprendre la structure de base d'un ordinateur, des logiciels et des langages de Programmation à usage général
- » Savoir appliquer les principes fondamentaux de la programmation C++, notamment les classes, les variables, les expressions conditionnelles et les objets
- » Connaissance approfondie des principales stratégies de conception d'algorithmes, ainsi que les différentes méthodes et mesures de calcul d'algorithmes
- » Connaissance des différentes applications et finalités des systèmes de bases de données, ainsi que leur fonctionnement et leur architecture
- » Être capable de présenter les différents systèmes de bases de données actuellement sur le marché
- » Savoir analyser en analysant les algorithmes récurrents et de division et de conquête, ainsi qu'en effectuant des analyses amorties
- » Utiliser les connaissances sur l'interaction homme-machine et la création d'interfaces utilisables dans l'exercice quotidien de la profession
- » Avoir une connaissance approfondie de la Programmation
- » Connaître les caractéristiques du langage de balisage HTML et son utilisation dans la création de sites web, ainsi que les feuilles de style CSS
- » Être capable d'appliquer les principaux processus et concepts des bases de l'ingénierie Logicielle et de la modélisation

# 04

## Structure et contenu

La structure des contenus a été conçue par une équipe de professionnels de l'ingénierie informatique dans le but de permettre aux étudiants du Mastère Spécialisé d'apprendre efficacement et rapidement. À cette fin, les contenus ont été organisés de manière à ce que l'apprentissage soit intensif et constant, en essayant de maintenir la motivation en fonction du sentiment de progrès de l'élève.

```
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
  
<?if ($send==2) {?>  
<td align="right" colspan="2">  
<input type="button" value="<?=checkLangItem("contact-text-1")?>" />  
</td>  
</tr>  
<tr>  
<td align="right" colspan="2">  
<div class="text-2">  
<?=checkLangItem("contact-arpakai")?>  
</div>  
</td>
```

```
<?=checkLangItem("contact-text-1")?></div>
```

```
onclick="window.location='<?=kontakti.ph
```

```
sci/index.php?sid=' + Math.r
```

“

*Un programme éducatif visant à acquérir une compétence complète en matière de développement de logiciels, qui vous propulsera à un nouveau niveau professionnel”*

## Module 1. Principes fondamentaux de la programmation

- 1.1. Introduction à la programmation
  - 1.1.1. Structure de base d'un ordinateur
  - 1.1.2. Software
  - 1.1.3. Langages de programmation
  - 1.1.4. Cycle de vie d'une application logicielle
- 1.2. Conception d'algorithmes
  - 1.2.1. Résolution de problèmes
  - 1.2.2. Techniques descriptives
  - 1.2.3. Éléments et structure d'un algorithme
- 1.3. Éléments d'un programme
  - 1.3.1. Origine et caractéristiques du langage C++
  - 1.3.2. L'environnement de développement
  - 1.3.3. Concept du programme
  - 1.3.4. Types de données fondamentales
  - 1.3.5. Opérateurs
  - 1.3.6. Expressions
  - 1.3.7. Phrases
  - 1.3.8. Entrée et sortie de données
- 1.4. Déclarations de contrôle
  - 1.4.1. Phrases
  - 1.4.2. Branches
  - 1.4.3. Boucles
- 1.5. Abstraction et modularité: fonctions
  - 1.5.1. Conception modulaire
  - 1.5.2. Concept de fonction et d'utilité
  - 1.5.3. Définition d'une fonction
  - 1.5.4. Flux d'exécution dans un appel de fonction
  - 1.5.5. Prototypes d'une fonction
  - 1.5.6. Retour des résultats
  - 1.5.7. Appel d'une fonction: paramètres
  - 1.5.8. Passage de paramètres par référence et par valeur
  - 1.5.9. Identifiant du champ d'application
- 1.6. Structures de données statiques
  - 1.6.1. *Tableaux*
  - 1.6.2. Les tableaux. Polyèdres
  - 1.6.3. Recherche et tri
  - 1.6.4. Cordes Fonctions d'E/S pour les chaînes de caractères
  - 1.6.5. Structures Jonctions
  - 1.6.6. Nouveaux types de données
- 1.7. Structures de données dynamiques: pointeurs
  - 1.7.1. Concept. Définition du pointeur
  - 1.7.2. Opérateurs et opérations avec des pointeurs
  - 1.7.3. *Tableaux* de pointeurs
  - 1.7.4. Pointeurs et *Tableaux*
  - 1.7.5. Pointeurs vers des chaînes de caractères
  - 1.7.6. Pointeurs vers des structures
  - 1.7.7. Indications multiples
  - 1.7.8. Pointeurs vers les fonctions
  - 1.7.9. Passage de fonctions, de structures et de *tableaux* comme paramètres de fonction
- 1.8. Fichiers
  - 1.8.1. Concepts de base
  - 1.8.2. Opérations avec des fichiers
  - 1.8.3. Types de fichiers
  - 1.8.4. Organisation des fichiers
  - 1.8.5. Introduction aux fichiers C++
  - 1.8.6. Traitement des fichiers
- 1.9. Récursion
  - 1.9.1. Définition de la récursion
  - 1.9.2. Types de récursions
  - 1.9.3. Avantages et inconvénients
  - 1.9.4. Considérations
  - 1.9.5. Conversion récursive itérative
  - 1.9.6. La pile de récursion

- 1.10. Tests et documentation
  - 1.10.1. Test du programme
  - 1.10.2. Test boîte blanche
  - 1.10.3. Tests en boîte noire
  - 1.10.4. Outils de test
  - 1.10.5. Documentation du logiciel

## Module 2. Structure des données

- 2.1 Introduction à la programmation C++
  - 2.1.1. Classes, constructeurs, méthodes et attributs
  - 2.1.2. Variables
  - 2.1.3. Expressions conditionnelles et boucles
  - 2.1.4. Objets
- 2.2. Types de données abstraites (ADT)
  - 2.2.1. Types de données
  - 2.2.2. Structures de base et TAD
  - 2.2.3. Vecteurs et *Matrices*
- 2.3. Structures de données linéaires
  - 2.3.1. Définition de la Liste des TAD
  - 2.3.2. Listes liées et listes doublement liées
  - 2.3.3. Listes ordonnées
  - 2.3.4. Listes C++
  - 2.3.5. Pile TAD
  - 2.3.6. File d'attente TAD
  - 2.3.7. Pile et File d'attente C++
- 2.4. Structures de données hiérarchique
  - 2.4.1. Arbre TAD
  - 2.4.2. Chemins d'accès
  - 2.4.3. Arbres n-aires
  - 2.4.4. Arbres binaires
  - 2.4.5. Arbres de recherche binaires
- 2.5. Structures de données hiérarchiques: arbres complexes
  - 2.5.1. Des arbres parfaitement équilibrés ou de hauteur minimale
  - 2.5.2. Arbres à trajets multiples
  - 2.5.3. Références bibliographiques

- 2.6. Monticules et file d'Attente prioritaire
  - 2.6.1. Monticules TAD
  - 2.6.2. File d'Attente prioritaire TAD
- 2.7. Tables *Hash*
  - 2.7.1. TAD Table *Hash*
  - 2.7.2. Fonctions *Hash*
  - 2.7.3. Fonction *Hash* en tables *Hash*
  - 2.7.4. Redispersion
  - 2.7.5. Tables *Hash* ouvertes
- 2.8. Réseaux
  - 2.8.1. Graphique TAD
  - 2.8.2. Types de Graphiques
  - 2.8.3. Représentation graphique et opérations de base
  - 2.8.4. Conception du Réseau
- 2.9. Algorithmes graphiques et concepts Graphiques avancés
  - 2.9.1. Problèmes de Graphiques
  - 2.9.2. Algorithmes de parcours
  - 2.9.3. Algorithmes de recherche ou de cheminement
  - 2.9.4. Autres algorithmes
- 2.10. Autres structures de données
  - 2.10.1. Sets
  - 2.10.2. *Matrices* parallèles
  - 2.10.3. Tableaux de symboles
  - 2.10.4. *Essais*

## Module 3. Algorithme et complexité

- 3.1. Introduction aux stratégies de conception d'algorithmes
  - 3.1.1. Récursion
  - 3.1.2. Diviser pour mieux régner
  - 3.1.3. Autres stratégies
- 3.2. Efficacité et analyse des algorithmes
  - 3.2.1. Mesures d'efficacité
  - 3.2.2. Taille de l'entrée de mesure
  - 3.2.3. Mesure du temps d'exécution
  - 3.2.4. Pire, meilleur et moyen cas
  - 3.2.5. Notation asymptotique
  - 3.2.6. Critères d'analyse mathématique des algorithmes non récursifs
  - 3.2.7. Analyse mathématique des algorithmes récursifs
  - 3.2.8. Analyse empirique des algorithmes
- 3.3. Algorithmes de tri
  - 3.3.1. Concept de tri
  - 3.3.2. Triage des bulles
  - 3.3.3. Tri par sélection
  - 3.3.4. Triage par insertion
  - 3.3.5. Fusionner le tri (Merge Sort)
  - 3.3.6. Tri rapide (QuickSort)
- 3.4. Algorithmes avec arbres
  - 3.4.1. Concept d'arbre
  - 3.4.2. Arbres binaires
  - 3.4.3. Allées d'arbres
  - 3.4.4. Représentation des expressions
  - 3.4.5. Arbres binaires ordonnés
  - 3.4.6. Arbres binaires équilibrés
- 3.5. Algorithmes avec *Heaps*
  - 3.5.1. Les *Heaps*
  - 3.5.2. L'algorithme HeapSort
  - 3.5.3. Files d'attente prioritaires

- 3.6. Algorithmes graphiques
  - 3.6.1. Représentation
  - 3.6.2. Voyage en largeur
  - 3.6.3. Profondeur de déplacement
  - 3.6.4. Disposition topologique
- 3.7. Algorithmes *Greedy*
  - 3.7.1. La stratégie *Greedy*
  - 3.7.2. Éléments de la stratégie *Greedy*
  - 3.7.3. Change de devises
  - 3.7.4. Le problème du voyageur
  - 3.7.5. Problème de sac à dos
- 3.8. Recherche de chemins minimaux
  - 3.8.1. Le problème du chemin minimal
  - 3.8.2. Arcs et cycles négatifs
  - 3.8.3. Algorithme de Dijkstra
- 3.9. Algorithmes *Greedy* sur les Graphes
  - 3.9.1. L'arbre à chevauchement minimal
  - 3.9.2. L'algorithme de Prim
  - 3.9.3. L'algorithme de Kruskal
  - 3.9.4. Analyse de la complexité
- 3.10. *Backtracking*
  - 3.10.1. Le *Backtracking*
  - 3.10.2. Techniques alternatives

## Module 4. Bases de données

- 4.1. Applications et objectifs des systèmes de bases de données
  - 4.1.1. Applications de différents systèmes de bases de données
  - 4.1.2. Objectif des différents systèmes de base de données
  - 4.1.3. Vue des données
- 4.2. Base de données et architecture
  - 4.2.1. Base de données relationnelle
  - 4.2.2. Conception de la base de données
  - 4.2.3. Bases de données à base d'objets et semi-structurées
  - 4.2.4. Stockage des données et requêtes
  - 4.2.5. Gestion des transactions
  - 4.2.6. Extraction et analyse de données
  - 4.2.7. Architecture des bases de données
- 4.3. Le modèle relationnel: structure, opérations et algèbre relationnelle étendue
  - 4.3.1. La structure des BD relationnelles
  - 4.3.2. Opérations fondamentales dans l'algèbre relationnelle
  - 4.3.3. Autres opérations dans l'algèbre relationnelle
  - 4.3.4. Opérations d'algèbre relationnelle étendues
  - 4.3.5. Valeurs nulles
  - 4.3.6. Modification de la base de données
- 4.4. SQL (I)
  - 4.4.1. Qu'est-ce que SQL?
  - 4.4.2. La définition des données
  - 4.4.3. La structure de base des requêtes SQL
  - 4.4.4. Opérations de réglage
  - 4.4.5. Fonctions d'agrégation
  - 4.4.6. Valeurs nulles
- 4.5. SQL (II)
  - 4.5.1. Sous-requêtes imbriquées
  - 4.5.2. Requêtes complexes
  - 4.5.3. Vues
  - 4.5.4. Curseurs
  - 4.5.5. Requêtes complexes
  - 4.5.6. Déclencheurs

- 4.6. La conception des bases de données et le modèle E-R
  - 4.6.1. Aperçu du processus de conception
  - 4.6.2. Le modèle entité-relation
  - 4.6.3. Restrictions
- 4.7. Diagramme entité relations
  - 4.7.1. Diagramme entité relations
  - 4.7.2. Aspects de la conception des relations entre entités
  - 4.7.3. Ensembles d'entités faibles
- 4.8. Le modèle entité-relation étendu
  - 4.8.1. Caractéristiques du modèle E-R étendu
  - 4.8.2. Conception d'une base de données
  - 4.8.3. Réduction aux schémas relationnels
- 4.9. Conception de bases de données relationnelles
  - 4.9.1. Caractéristiques des bonnes conceptions relationnelles
  - 4.9.2. Domaines atomiques et première forme normale (1FN)
  - 4.9.3. Décomposition à l'aide de dépendances fonctionnelles
  - 4.9.4. Théorie de la dépendance fonctionnelle
  - 4.9.5. Algorithmes de décomposition
  - 4.9.6. Décomposition à l'aide de dépendances multivaluées
  - 4.9.7. Autres formes normales
  - 4.9.8. Processus de conception des bases de données
- 4.10. Bases de données NoSQL
  - 4.10.1. Que sont les bases de données NoSQL?
  - 4.10.2. Analyse des différentes options NoSQL et de leurs caractéristiques
  - 4.10.3. MongoDB

## Module 5. Bases de données avancées

- 5.1. Introduction à différents systèmes de base de données
  - 5.1.1. Rappel historique
  - 5.1.2. Bases de données hiérarchique
  - 5.1.3. Bases de données réseaux
  - 5.1.4. Bases de données relationnelles
  - 5.1.5. Bases de données non relationnelles
- 5.2. XML et bases de données pour le web
  - 5.2.1. Validation des documents XML
  - 5.2.2. Transformations de documents XML
  - 5.2.3. Stockage des données XML
  - 5.2.4. Bases de données relationnelles XML
  - 5.2.5. SQL/XML
  - 5.2.6. Bases de données natives XML
- 5.3. Bases de données parallèles
  - 5.3.1. Systèmes parallèles
  - 5.3.2. Architectures de bases de données parallèles
  - 5.3.4. Parallélisme des requêtes
  - 5.3.5. Parallélisme des requêtes
  - 5.3.6. Conception du système parallèle
  - 5.3.7. Traitement parallèle en SQL
- 5.4. Bases de données distribuées
  - 5.4.1. Systèmes distribués
  - 5.4.2. Stockage distribué
  - 5.4.3. Disponibilité
  - 5.4.4. Traitement distribué des requêtes
  - 5.4.5. Fournisseurs de bases de données distribuées
- 5.5. Indexation et association
  - 5.5.1. Indices ordonnés
  - 5.5.2. Index denses et épars
  - 5.5.3. Indices multiniveaux
  - 5.5.4. Mise à jour de l'index
  - 5.5.5. Association statique
  - 5.5.6. Comment utiliser les index dans les bases de données

- 5.6. Introduction au traitement transactionnel
  - 5.6.1. États d'une transaction
  - 5.6.2. Mise en œuvre de l'atomicité et de la durabilité
  - 5.6.3. Séquentialité
  - 5.6.4. Récupérabilité
  - 5.6.5. Mise en œuvre de l'isolation
- 5.7. Systèmes de récupération
  - 5.7.1. Classification des défauts
  - 5.7.2. Structures de stockage
  - 5.7.3. Récupération et atomicité
  - 5.7.4. Récupération basée sur l'historique
  - 5.7.5. Transactions et récupérations simultanées
  - 5.7.6. Haute disponibilité dans les bases de données
- 5.8. Exécution et traitement des requêtes
  - 5.8.1. Coût d'une requête
  - 5.8.2. Opération de sélection
  - 5.8.3. Triage
  - 5.8.4. Introduction à l'optimisation des requêtes
  - 5.8.5. Suivi des performances
- 5.9. Bases de données non relationnelles
  - 5.9.1. Bases de données orientées documents
  - 5.9.2. Bases de données orientées Graphes
  - 5.9.3. Bases de données clés-valeurs
- 5.10. Data Warehouse, OLAP et data mining
  - 5.10.1. Composants de l'entrepôt de données
  - 5.10.2. Architecture d'un data Warehouse
  - 5.10.3. OLAP
  - 5.10.4. Fonctionnalités d'exploration de données
  - 5.10.5. Autres types d'exploitation minière

## Module 6. Conception d'algorithmes avancés

- 6.1. Analyse des algorithmes récursifs et des algorithmes "diviser pour régner"
  - 6.1.1. Poser et résoudre des équations de récursion homogènes et non-homogènes
  - 6.1.2. Aperçu de la stratégie "diviser pour régner"
- 6.2. Analyse amortie
  - 6.2.1. Analyse des agrégats
  - 6.2.2. La méthode de comptabilisation
  - 6.2.3. La méthode du potentiel
- 6.3. Programmation dynamique et algorithmes pour les problèmes NP
  - 6.3.1. Caractéristiques de la programmation dynamique
  - 6.3.2. Retour en arrière: *Backtracking*
  - 6.3.3. Branchements et élagage
- 6.4. Optimisation combinatoire
  - 6.4.1. Représentation du problème
  - 6.4.2. Optimisation 1D
- 6.5. Algorithmes de randomisation
  - 6.5.1. Exemples d'algorithmes de randomisation
  - 6.5.2. Le théorème de Buffon
  - 6.5.3. Algorithme de Monte Carlo
  - 6.5.4. Algorithme de Las Vegas
- 6.6. Recherche locale et recherche de candidats
  - 6.6.1. *Montée en Gradient*
  - 6.6.2. *Hill Climbing*
  - 6.6.3. *Simulated Annealing*
  - 6.6.4. *Tabu Search*
  - 6.6.5. Recherche de candidats
- 6.7. Vérification formelle des programmes
  - 6.7.1. Spécification d'abstractions fonctionnelles
  - 6.7.2. Le langage de la logique du premier ordre
  - 6.7.3. Le système formel de Hoare

- 6.8. Vérification itérative des programmes
  - 6.8.1. Règles du système formel de Hoare
  - 6.8.2. Concept d'invariant d'itération
- 6.9. Méthodes numériques
  - 6.9.1. La méthode de bisection
  - 6.9.2. La méthode Newton Raphson
  - 6.9.3. La méthode sécante
- 6.10. Algorithmes parallèles
  - 6.10.1. Opérations binaires parallèles
  - 6.10.1. Opérations parallèles avec les réseaux
  - 6.10.3. Le parallélisme dans diviser pour mieux régner
  - 6.10.4. Le parallélisme dans la programmation dynamique

## Module 7. Interaction Homme-machine

- 7.1. Introduction à l'interaction homme-machine
  - 7.1.1. Qu'est-ce que l'interaction homme-machine?
  - 7.1.2. Relations entre l'interaction homme-machine et les autres disciplines
  - 7.1.3. L'interface utilisateur
  - 7.1.4. Utilisabilité et accessibilité
  - 7.1.5. Expérience de utilisateur et conception centrée sur l'utilisateur
- 7.2. Informatique et interaction: paradigmes d'interface utilisateur et d'interaction
  - 7.2.1. Interaction
  - 7.2.2. Paradigmes d'interaction et styles d'interaction
  - 7.2.3. Évolution des interfaces utilisateur
  - 7.2.4. Interfaces utilisateur classiques: WIMP/GUI, commandes, voix, réalité virtuelle
  - 7.2.5. Interfaces utilisateur innovantes: mobiles, portables, collaboratives, BCI
- 7.3. Le facteur humain: aspects psychologiques et cognitifs
  - 7.3.1. L'importance du facteur humain dans l'interaction
  - 7.3.2. Traitement de l'information humaine
  - 7.3.3. L'entrée et la sortie d'informations: visuelles, auditives et tactiles
  - 7.3.4. Perception et attention
  - 7.3.5. Connaissance et modèles mentaux: représentation, organisation et acquisition

- 7.4. Le facteur humain: les limitations sensorielles et physiques
  - 7.4.1. Diversité fonctionnelle, handicap et déficience
  - 7.4.2. Diversité visuelle
  - 7.4.3. La diversité auditive
  - 7.4.4. Diversité cognitive
  - 7.4.5. Diversité des moteurs
  - 7.4.6. Le cas des immigrants numériques
- 7.5. Le processus de conception (I): analyse des besoins pour la conception de l'interface utilisateur
  - 7.5.1. Conception centrée sur l'utilisateur
  - 7.5.2. Qu'est-ce que l'analyse des besoins?
  - 7.5.3. Collecte d'informations
  - 7.5.4. Analyse et interprétation des informations
  - 7.5.5. Analyse de l'utilisabilité et de l'accessibilité
- 7.6. Le processus de conception (II): Prototypage et analyse des tâches
  - 7.6.1. Design conceptuel
  - 7.6.2. Prototypage
  - 7.6.3. Analyse hiérarchique des tâches
- 7.7. Le processus de conception (III): évaluation
  - 7.7.1. L'évaluation dans le processus de conception: objectifs et méthodes
  - 7.7.2. Méthodes d'évaluation sans utilisateurs
  - 7.7.3. Méthodes d'évaluation avec les utilisateurs
  - 7.7.4. Normes et standards d'évaluation
- 7.8. Accessibilité: définition et lignes directrices
  - 7.8.1. Accessibilité et conception universelle
  - 7.8.2. Initiative WAI et directives WCAG
  - 7.8.3. Directives WCAG 2.0 et 2.1
- 7.9. Accessibilité: évaluation et diversité fonctionnelle
  - 7.9.1. Outils d'évaluation de l'accessibilité du Web
  - 7.9.2. Accessibilité et diversité fonctionnelle
- 7.10. L'ordinateur et l'interaction : périphériques et dispositifs
  - 7.10.1. Dispositifs et périphériques traditionnels
  - 7.10.2. Dispositifs alternatifs et périphériques
  - 7.10.3. Téléphones mobiles et tablettes
  - 7.10.4. Diversité fonctionnelle, interaction et périphériques

## Module 8. Programmation avancée

- 8.1. Introduction à la programmation orientée objet
  - 8.1.1. Introduction à la programmation orientée objet
  - 8.1.2. Conception de la classe
  - 8.1.3. Introduction à UML pour la modélisation des problèmes
- 8.2. Relations entre les classes
  - 8.2.1. Abstraction et héritage
  - 8.2.2. Concepts d'héritage avancés
  - 8.2.3. Polymorphisme
  - 8.2.4. Composition et agrégation
- 8.3. Introduction aux patrons de conception pour les problèmes orientés objet
  - 8.3.1. Que sont les modèles de conception?
  - 8.3.2. Modèle *Factory*
  - 8.3.4. Modèle *Singleton*
  - 8.3.5. Modèle *Observer*
  - 8.3.6. Modèle *Composite*
- 8.4. Exceptions
  - 8.4.1. Quelles sont les exceptions?
  - 8.4.2. Capture et traitement des exceptions
  - 8.4.3. Lancer d'exceptions
  - 8.4.4. Création d'exceptions
- 8.5. Interfaces utilisateur
  - 8.5.1. Introduction à Qt
  - 8.5.2. Positionnement
  - 8.5.3. Que sont les événements?
  - 8.5.4. Événements: définition et saisie
  - 8.5.5. Développement d'interfaces utilisateurs
- 8.6. Introduction à la programmation concurrente
  - 8.6.1. Introduction à la programmation concurrente
  - 8.6.2. Le concept de processus et de fil conducteur
  - 8.6.3. Interaction entre processus ou threads
  - 8.6.4. Threads en C++
  - 8.6.6. Avantages et inconvénients de la programmation concurrente
- 8.7. Gestion et synchronisation des threads
  - 8.7.1. Cycle de vie du fil
  - 8.7.2. La classe *Thread*
  - 8.7.3. Programmation du fil
  - 8.7.4. Groupes de fils
  - 8.7.5. Fils démoniaques
  - 8.7.6. Synchronisation
  - 8.7.7. Mécanismes de verrouillage
  - 8.7.8. Mécanismes de communication
  - 8.7.9. Moniteurs
- 8.8. Problèmes courants de la programmation concurrente
  - 8.8.1. Le problème du producteur-consommateur
  - 8.8.2. Le problème des lecteurs et des écrivains
  - 8.8.3. Le problème du dîner des philosophes
- 8.9. Documentation et test des logiciels
  - 8.9.1. Pourquoi est-il important de documenter les logiciels?
  - 8.9.2. Documentation sur la conception
  - 8.9.3. Utilisation des outils de documentation
- 8.10. Tests de logiciels
  - 8.10.1. Introduction aux tests logiciels
  - 8.10.1. Types de tests
  - 8.10.2. Tests unitaires
  - 8.10.3. Test d'intégration
  - 8.10.4. Test de validation
  - 8.10.6. Test du système

## Module 9. Développement d'applications de réseau

- 9.1. Langages de balisage HTML5
  - 9.1.1. Les bases du HTML
  - 9.1.2. Nouveaux éléments HTML 5
  - 9.1.3. Formulaires: nouveaux contrôles
- 9.2. Introduction aux feuilles de style CSS
  - 9.2.1. Premiers pas avec CSS
  - 9.2.2. Introduction à CSS3
- 9.3. Langage de script du navigateur: JavaScript
  - 9.3.1. Les bases du JavaScript
  - 9.3.2. DOM
  - 9.3.3. Événements
  - 9.3.4. JQuery
  - 9.3.5. Ajax
- 9.4. Concept de la programmation orientée vers les composants
  - 9.4.1. Contexte
  - 9.4.2. Composants et interfaces
  - 9.4.3. États d'un composant
- 9.5. Architecture des composants
  - 9.5.1. Architectures actuelles
  - 9.5.2. Intégration et déploiement des composants
- 9.6. *Framework Frontend*: Bootstrap
  - 9.6.1. Conception avec grille
  - 9.6.2. Formulaires
  - 9.6.3. Composants
- 9.7. Contrôleur de modèle et de vue
  - 9.7.1. Méthodes de développement Web
  - 9.7.2. Modèle de conception: MVC
- 9.8. Technologies de la Grille d'information
  - 9.8.1. Augmentation des ressources informatiques
  - 9.8.2. Concept de la technologie des Grilles

- 9.9. Architecture orientée services
  - 9.9.1. SOA et services web
  - 9.9.2. Topologie des services web
  - 9.9.3. Plateformes de services web
- 9.10. Protocole HTTP
  - 9.10.1. Messages
  - 9.10.2. Sessions persistantes
  - 9.10.3. Système cryptographique
  - 9.10.4. Fonctionnement du protocole HTTPS

## Module 10. Ingénierie logicielle

- 10.1. Introduction au génie logiciel et à la modélisation
  - 10.1.1. La nature des logiciels
  - 10.1.2. La nature unique des WebApps
  - 10.1.3. Ingénierie logicielle
  - 10.1.4. Le processus logiciel
  - 10.1.5. La pratique du génie logiciel
  - 10.1.6. Mythes sur les logiciels
  - 10.1.7. Comment tout commence
  - 10.1.8. Concepts orientés objet
  - 10.1.9. Introduction à UML
- 10.2. Le processus logiciel
  - 10.2.1. Un modèle général de processus
  - 10.2.2. Modèles de processus prescriptifs
  - 10.2.3. Modèles de processus spécialisés
  - 10.2.4. Le processus unifié
  - 10.2.5. Modèles de processus personnels et d'équipe
  - 10.2.6. Qu'est-ce que l'agilité?
  - 10.2.7. Qu'est-ce qu'un processus agile?
  - 10.2.8. Scrum
  - 10.2.9. Boîte à outils du processus Agile

- 10.3. Principes guidant la pratique du génie logiciel
  - 10.3.1. Principes guidant le processus
  - 10.3.2. Principes guidant la pratique
  - 10.3.3. Principes de la communication
  - 10.3.4. Principes de planification
  - 10.3.5. Principes de modélisation
  - 10.3.6. Principes de construction
  - 10.3.7. Principes de déploiement
- 10.4. Comprendre les besoins
  - 10.4.1. Ingénierie des exigences
  - 10.4.2. Établir la base
  - 10.4.3. Détermination des besoins
  - 10.4.4. Développer des cas d'utilisation
  - 10.4.5. Élaboration du modèle d'exigences
  - 10.4.6. Négociation des exigences
  - 10.4.7. Validation des exigences
- 10.5. Modélisation des exigences: scénarios, classes d'information et d'analyse
  - 10.5.1. Analyse des besoins
  - 10.5.2. Modélisation basée sur des scénarios
  - 10.5.3. Modèles UML fournissant le cas d'utilisation
  - 10.5.4. Concepts de modélisation des données
  - 10.5.5. Modélisation basée sur les classes
  - 10.5.6. Diagrammes de classes
- 10.6. Modélisation des exigences: flux, comportement et modèles
  - 10.6.1. Stratégies de modélisation des exigences
  - 10.6.2. Modélisation orientée flux
  - 10.6.3. Diagrammes d'état
  - 10.6.4. Création d'un modèle comportemental
  - 10.6.5. Diagrammes de séquence
  - 10.6.6. Diagrammes de communication
  - 10.6.7. Modèles pour la modélisation des exigences
- 10.7. Concepts de design
  - 10.7.1. La conception dans le contexte du génie logiciel
  - 10.7.2. Le processus de conception
  - 10.7.3. Concepts de design
  - 10.7.4. Concepts de conception orientée objet
  - 10.7.5. Le modèle de conception
- 10.8. Architecture de conception
  - 10.8.1. Architecture logicielle
  - 10.8.2. Genres architecturaux
  - 10.8.3. Styles architecturaux
  - 10.8.4. Conception architecturale
  - 10.8.5. Évolution des conceptions alternatives de l'architecture
  - 10.8.6. Cartographie de l'architecture à l'aide du flux de données
- 10.9. Conception au niveau des composants et des modèles
  - 10.9.1. Qu'est-ce qu'un composant?
  - 10.9.2. Conception de composants basée sur les classes
  - 10.9.3. Réalisation de la conception au niveau des composants
  - 10.9.4. Conception traditionnelle des composants
  - 10.9.5. Développement basé sur les composants
  - 10.9.6. Modèles de conception
  - 10.9.7. Conception de logiciels basée sur des modèles
  - 10.9.8. Modèles architecturaux
  - 10.9.9. Patrons de conception au niveau des composants
  - 10.9.10. Modèles de conception d'interface utilisateur
- 10.10. Qualité des logiciels et gestion de projet
  - 10.10.1. Qualité
  - 10.10.1. Qualité des logiciels
  - 10.10.2. Le dilemme de la qualité des logiciels
  - 10.10.3. Atteindre la qualité des logiciels
  - 10.10.4. Assurance qualité des logiciels
  - 10.10.5. Le spectre de la gestion
  - 10.10.6. Personnel
  - 10.10.7. Le produit
  - 10.10.8. Le processus
  - 10.10.9. Le projet
  - 10.10.10. Principes et pratiques

# 05 Méthodologie

Ce programme de formation offre une manière différente d'apprendre. Notre méthodologie est développée à travers un mode d'apprentissage cyclique: ***le Relearning***.

Ce système d'enseignement est utilisé, par exemple, dans les écoles de médecine les plus prestigieuses du monde et a été considéré comme l'un des plus efficaces par des publications de premier plan telles que le ***New England Journal of Medicine***.



“

*Découvrez Relearning, un système qui renonce à l'apprentissage linéaire conventionnel pour vous emmener à travers des systèmes d'enseignement cycliques: une façon d'apprendre qui s'est avérée extrêmement efficace, en particulier dans les matières qui exigent la mémorisation”*

## Étude de Cas pour mettre en contexte tout le contenu

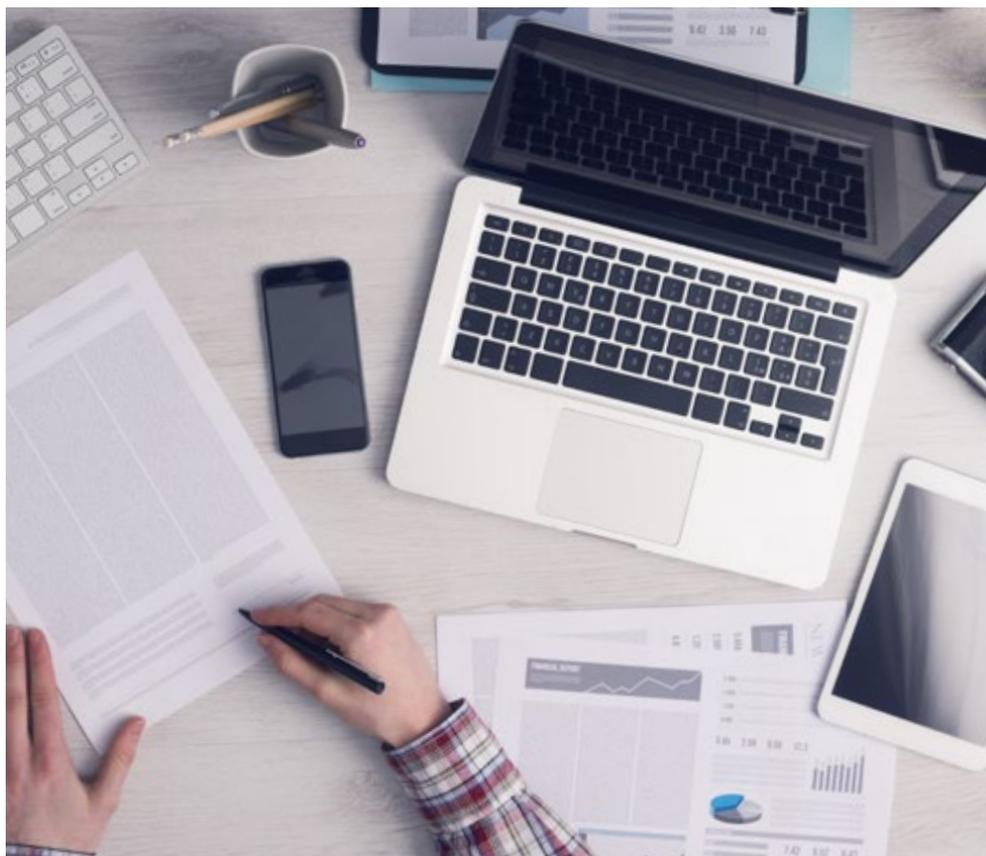
Notre programme offre une méthode révolutionnaire de développement des compétences et des connaissances. Notre objectif est de renforcer les compétences dans un contexte changeant, compétitif et hautement exigeant.

“

*Avec TECH, vous pouvez expérimenter une manière d'apprendre qui ébranle les fondations des universités traditionnelles du monde entier”*



*Vous bénéficierez d'un système d'apprentissage basé sur la répétition, avec un enseignement naturel et progressif sur l'ensemble du cursus.*



*L'étudiant apprendra, par des activités collaboratives et des cas réels, à résoudre des situations complexes dans des environnements commerciaux réels.*

## Une méthode d'apprentissage innovante et différente

Cette formation TECH est un programme d'enseignement intensif, créé de toutes pièces, qui propose les défis et les décisions les plus exigeants dans ce domaine, tant au niveau national qu'international. Grâce à cette méthodologie, l'épanouissement personnel et professionnel est stimulé, faisant ainsi un pas décisif vers la réussite. La méthode des cas, technique qui constitue la base de ce contenu, permet de suivre la réalité économique, sociale et professionnelle la plus actuelle.

“ Notre programme vous prépare à relever de nouveaux défis dans des environnements incertains et à réussir votre carrière ”

La méthode des cas est le système d'apprentissage le plus largement utilisé dans les meilleures écoles d'informatique du monde depuis qu'elles existent. Développée en 1912 pour que les étudiants en Droit n'apprennent pas seulement le droit sur la base d'un contenu théorique, la méthode des cas consiste à leur présenter des situations réelles complexes afin qu'ils prennent des décisions éclairées et des jugements de valeur sur la manière de les résoudre. En 1924, elle a été établie comme méthode d'enseignement standard à Harvard.

Dans une situation donnée, que doit faire un professionnel? C'est la question à laquelle nous sommes confrontés dans la méthode des cas, une méthode d'apprentissage orientée vers l'action. Tout au long du programme, les étudiants seront confrontés à de multiples cas réels. Ils devront intégrer toutes leurs connaissances, faire des recherches, argumenter et défendre leurs idées et leurs décisions.

## Relearning Methodology

TECH combine efficacement la méthodologie des Études de Cas avec un système d'apprentissage 100% en ligne basé sur la répétition, qui associe différents éléments didactiques dans chaque leçon.

Nous enrichissons l'Étude de Cas avec la meilleure méthode d'enseignement 100% en ligne: le Relearning.

*En 2019, nous avons obtenu les meilleurs résultats d'apprentissage de toutes les universités en ligne du monde.*

À TECH, vous apprendrez avec une méthodologie de pointe conçue pour former les managers du futur. Cette méthode, à la pointe de la pédagogie mondiale, est appelée Relearning.

Notre université est la seule université autorisée à utiliser cette méthode qui a fait ses preuves. En 2019, nous avons réussi à améliorer les niveaux de satisfaction globale de nos étudiants (qualité de l'enseignement, qualité des supports, structure des cours, objectifs...) par rapport aux indicateurs de la meilleure université en ligne.





Dans notre programme, l'apprentissage n'est pas un processus linéaire, mais se déroule en spirale (apprendre, désapprendre, oublier et réapprendre). Par conséquent, chacun de ces éléments est combiné de manière concentrique. Cette méthodologie a permis de former plus de 650.000 diplômés universitaires avec un succès sans précédent dans des domaines aussi divers que la biochimie, la génétique, la chirurgie, le droit international, les compétences en gestion, les sciences du sport, la philosophie, le droit, l'ingénierie, le journalisme, l'histoire, les marchés financiers et les instruments. Tout cela dans un environnement très exigeant, avec un corps étudiant universitaire au profil socio-économique élevé et dont l'âge moyen est de 43,5 ans.

*Le Relearning vous permettra d'apprendre avec moins d'efforts et plus de performance, en vous impliquant davantage dans votre formation, en développant un esprit critique, en défendant des arguments et en contrastant les opinions: une équation directe vers le succès.*

À partir des dernières preuves scientifiques dans le domaine des neurosciences, non seulement nous savons comment organiser les informations, les idées, les images et les souvenirs, mais nous savons aussi que le lieu et le contexte dans lesquels nous avons appris quelque chose sont fondamentaux pour notre capacité à nous en souvenir et à le stocker dans l'hippocampe, pour le conserver dans notre mémoire à long terme.

De cette manière, et dans ce que l'on appelle Neurocognitive context-dependent e-learning, les différents éléments de notre programme sont reliés au contexte dans lequel le participant développe sa pratique professionnelle.

Ce programme offre le support matériel pédagogique, soigneusement préparé pour les professionnels:



#### Support d'étude

Tous les contenus didactiques sont créés par les spécialistes qui enseigneront le cours, spécifiquement pour le cours, afin que le développement didactique soit vraiment spécifique et concret.

Ces contenus sont ensuite appliqués au format audiovisuel, pour créer la méthode de travail TECH en ligne. Tout cela, avec les dernières techniques qui offrent des pièces de haute qualité dans chacun des matériaux qui sont mis à la disposition de l'étudiant.



#### Cours magistraux

Il existe des preuves scientifiques de l'utilité de l'observation par un tiers expert.

La méthode "Learning from an Expert" renforce les connaissances et la mémoire, et donne confiance dans les futures décisions difficiles.



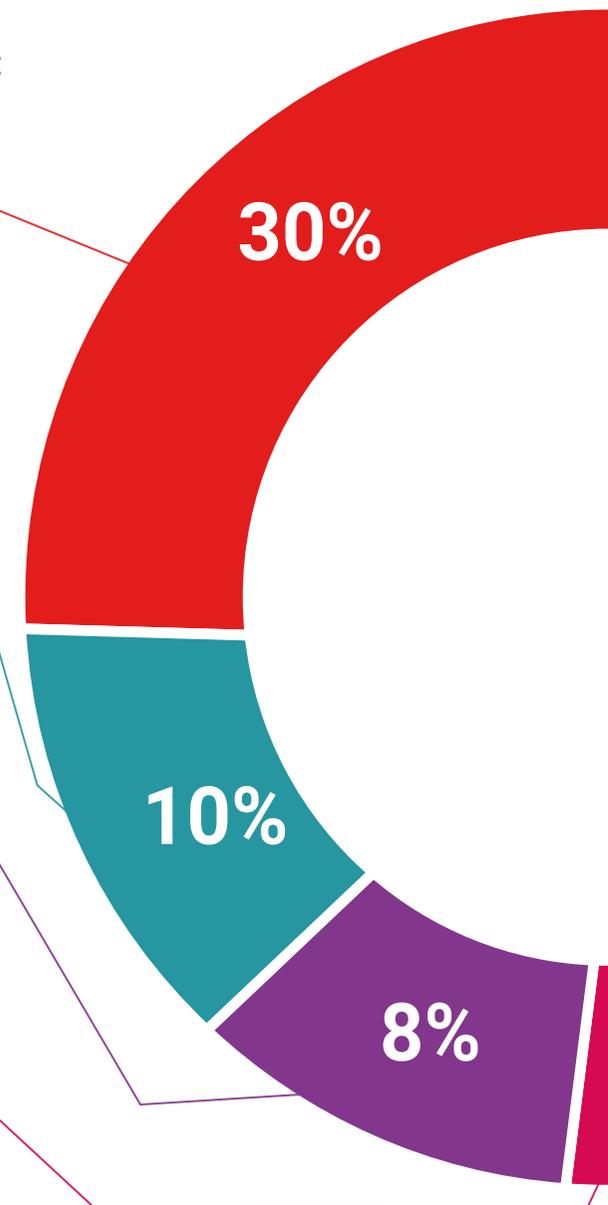
#### Pratiques en compétences et aptitudes

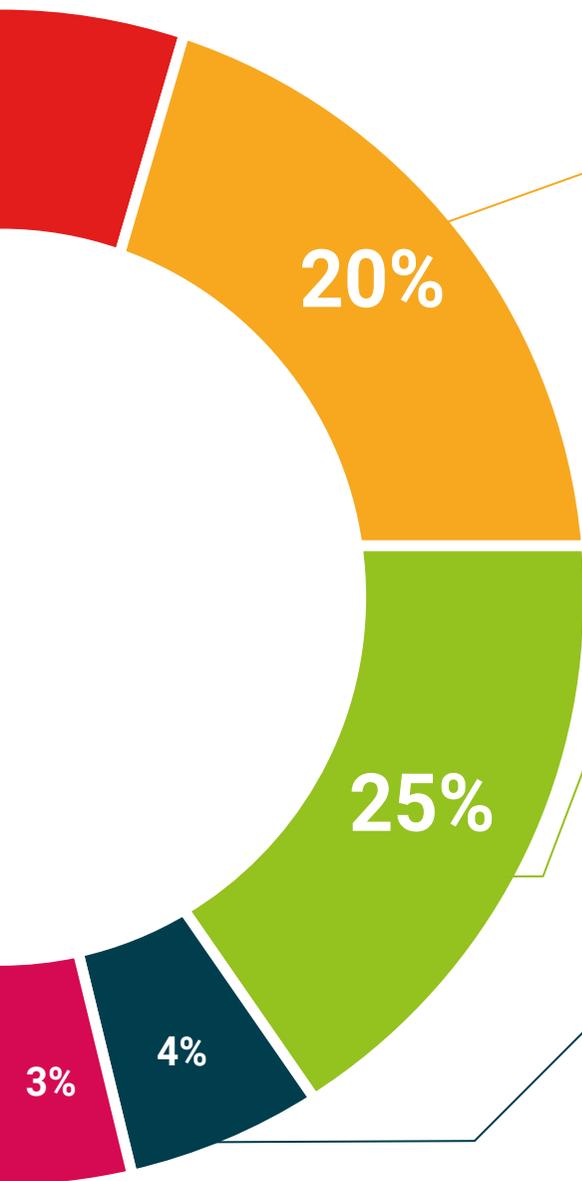
Les étudiants réaliseront des activités visant à développer des compétences et des aptitudes spécifiques dans chaque domaine. Des activités pratiques et dynamiques pour acquérir et développer les compétences et aptitudes qu'un spécialiste doit développer dans le cadre de la mondialisation dans laquelle nous vivons.



#### Lectures complémentaires

Articles récents, documents de consensus et directives internationales, entre autres. Dans la bibliothèque virtuelle de TECH, l'étudiant aura accès à tout ce dont il a besoin pour compléter sa formation.





#### Case studies

Ils réaliseront une sélection des meilleures études de cas choisies spécifiquement pour ce diplôme. Des cas présentés, analysés et tutorés par les meilleurs spécialistes de la scène internationale.



#### Résumés interactifs

L'équipe TECH présente les contenus de manière attrayante et dynamique dans des pilules multimédia comprenant des audios, des vidéos, des images, des diagrammes et des cartes conceptuelles afin de renforcer les connaissances. Ce système éducatif unique pour la présentation de contenu multimédia a été récompensé par Microsoft en tant que "European Success Story".



#### Testing & Retesting

Les connaissances de l'étudiant sont périodiquement évaluées et réévaluées tout au long du programme, par le biais d'activités et d'exercices d'évaluation et d'auto-évaluation, afin que l'étudiant puisse vérifier comment il atteint ses objectifs.



# 06 Diplôme

Le Mastère Spécialisé en Développement de Logiciels vous garantit, en plus de la formation la plus rigoureuse et la plus actuelle, l'accès à un diplôme universitaire de Mastère Spécialisé délivré par TECH Université Technologique.



“

*Finalisez cette formation avec succès  
et recevez votre Mastère Spécialisé  
avoir à vous soucier des déplacements  
ou des démarches administratives”*

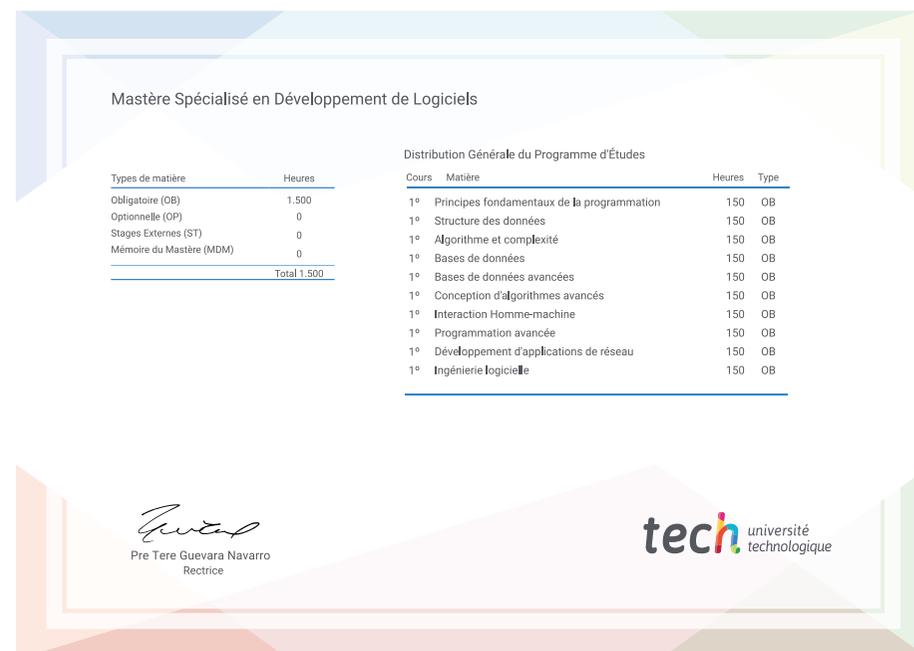
Le **Mastère Spécialisé en Développement de Logiciels** contient le programme le plus complet et le plus actuel du marché.

Après avoir réussi l'évaluation, l'étudiant recevra par courrier postal\* avec accusé de réception son correspondant diplôme de **Mastère Spécialisé** délivré par **TECH Université Technologique**.

Le diplôme délivré par **TECH Université Technologique** indiquera la note obtenue lors du Mastère Spécialisé, et répond aux exigences communément demandées par les bourses d'emploi, les concours et les commissions d'évaluation des carrières professionnelles.

Diplôme: **Mastère Spécialisé en Développement de Logiciels**

N.° d'Heures Officielles: **1.500 h.**



\*Si l' tudiant souhaite que son dipl me version papier poss de l'Apostille de La Haye, TECH EDUCATION fera les d marches n cessaires pour son obtention moyennant un co t suppl mentaire.

future

santé confiance personnes

éducation information tuteurs

garantie accréditation enseignement

institutions technologie apprentissage

communauté engagement

service personnalisé innovation

connaissance présent qualité

en ligne formation

développement institutions

classe virtuelle langues

**tech** université  
technologique

## Mastère Spécialisé Développement de Logiciels

- » Modalité: en ligne
- » Durée: 12 mois
- » Qualification: TECH Université Technologique
- » Intensité: 16h/semaine
- » Horaire: à votre rythme
- » Examens: en ligne

# Mastère Spécialisé Développement de Logiciels