FONDAMENTAUX : HTML, CSS, JS, PYTHON

Algorithmique et logique de programmation

Comprendre les bases de la pensée algorithmique avant d'apprendre un langage. Variables, conditions, boucles, fonctions : les fondations de tout développeur.

DURÉE	TARIF HT	NIVEAU	LANGUE	GROUPE	FORMAT
2j (14h)	~~490.00 €~~ **299.00 €**	**Débutant**	**Français**	**2-6**	**Formation**

1 PUBLIC VISÉ

- Grands débutants sans aucune expérience en développement ou en programmation
- Personnes en reconversion professionnelle cherchant un premier contact structuré avec l'informatique
- Étudiants souhaitant consolider leur compréhension de la logique avant d'apprendre un langage
- Toute personne souhaitant comprendre comment "pense" un développeur avant de coder

2 PRÉREQUIS

- Aucun prérequis technique
- Savoir utiliser un ordinateur et un navigateur web
- Curiosité et motivation suffisent

3 OBJECTIFS PÉDAGOGIQUES

- Comprendre ce qu'est un algorithme et son rôle dans le développement
- Manipuler les concepts fondamentaux : variables, types de données, conditions et boucles
- Décomposer un problème complexe en étapes logiques et ordonnées
- Écrire des pseudocodes lisibles et structurés
- Comprendre le rôle des fonctions et de la modularité
- Réaliser le passage du pseudocode à l'implémentation dans un langage réel

4 PROGRAMME DÉTAILLÉ

Module 1 – Introduction à la pensée algorithmique (2h)

- Qu'est-ce qu'un algorithme ? Exemples du quotidien
- Différence entre algorithme, programme et langage
- Les étapes de la résolution d'un problème
- Notion d'entrée, traitement, sortie

Module 2 – Variables et données (2h)

- Variables : déclaration, affectation, portée
- Types de données : entiers, flottants, chaînes, booléens
- Opérateurs arithmétiques et logiques
- Exercices pratiques en pseudocode

Module 3 – Structures conditionnelles (2h)

- Si / Sinon / Sinon si
- Conditions imbriquées
- Cas pratiques : validation d'une saisie, choix d'un chemin
- Erreurs de logique fréquentes

Module 4 – Boucles et itérations (3h)

- Boucle Tant que (while)
- Boucle Pour (for)
- Boucles imbriquées
- Cas pratiques : comptage, accumulation, recherche

Module 5 – Fonctions et modularité (2h)

- Définir et appeler une fonction
- Paramètres et valeurs de retour
- Intérêt de la réutilisabilité du code
- Exercices de décomposition fonctionnelle

Module 6 – Tableaux et structures de données simples (1h)

- Notion de tableau / liste
- Parcourir et modifier un tableau
- Cas pratiques : tri simplifié, recherche d'un élément

Module 7 – Du pseudocode au vrai code (2h)

- Transposition vers Python (ou JavaScript selon le parcours)
- Comparaison syntaxe pseudocode / langage réel
- Mini-exercice de traduction et exécution

5 COMPÉTENCES VISÉES

- Raisonner de façon algorithmique face à un problème concret
- Écrire des structures de contrôle (conditions, boucles) en pseudocode
- Décomposer un problème complexe en sous-problèmes résolubles
- Lire et comprendre un algorithme simple écrit par un tiers
- Transposer une logique algorithmique vers un premier langage de programmation

6 MODALITÉS PÉDAGOGIQUES

Formation délivrée en présentiel ou distanciel (visioconférence). Le formateur alterne entre méthode démonstrative (résolution pas à pas au tableau ou à l'écran), méthode interrogative (questionnement socratique pour faire émerger la logique) et méthode active (exercices de pseudocode et mise en situation). L'accent est mis sur la compréhension profonde plutôt que sur la mémorisation syntaxique.

7 MOYENS PÉDAGOGIQUES ET TECHNIQUES

- Support de cours numérique mis à disposition des apprenants
- Fiches d'exercices progressives avec corrigés détaillés
- Outil de pseudocode en ligne (ex : Flowgorithm ou équivalent) pour visualiser l'exécution
- Tableau blanc interactif (présentiel) ou outil collaboratif type Excalidraw (distanciel)
- Pour le distanciel : visioconférence (Microsoft Teams ou équivalent), partage d'écran, chat en direct
- Accès à la plateforme pédagogique **LaPolaris** (supports, ressources, émargement)

8 MODALITÉS D'ÉVALUATION

- En cours de formation : exercices de pseudocode corrigés à chaque module, mises en situation logiques
- En fin de formation : résolution d'un problème algorithmique complet (de l'analyse au pseudocode structuré)
- Questionnaire d'auto-évaluation des acquis en fin de parcours

9 CRITÈRES D'ÉVALUATION

- Capacité à décomposer un problème en étapes logiques claires et ordonnées
- Exactitude et lisibilité du pseudocode produit
- Bonne utilisation des structures de contrôle (conditions, boucles, fonctions)
- Autonomie dans la résolution d'un exercice algorithmique inédit
- Capacité à transposer un algorithme simple vers un langage de programmation réel

10 MODALITÉS DE VALIDATION

Attestation de fin de formation délivrée à l'issue du parcours, conditionnée à une assiduité d'au moins 80 % et à la réalisation de l'exercice final. L'attestation précise les objectifs atteints et les compétences acquises.

11 SUIVI ET ACCOMPAGNEMENT

- Feuilles d'émargement signées par demi-journée (présentiel) ou émargement numérique (distanciel)
- Traçabilité des activités pédagogiques réalisées
- Attestation d'assiduité délivrée en fin de formation
- Suivi individuel de la progression via les exercices pratiques corrigés

12 CONDITIONS D'ACCÈS

Formation accessible sur inscription directe, sans prérequis administratif particulier. Le financement peut être pris en charge par l'employeur dans le cadre d'un plan de développement des compétences, ou en autofinancement avec possibilité de paiement en plusieurs fois.

13 DÉLAIS D'ACCÈS

Inscription possible jusqu'à **5 jours ouvrés** avant le début de la session. Pour toute demande urgente, nous contacter directement.

ACCESSIBILITÉ · HANDICAP

Nos formations sont accessibles aux personnes en situation de handicap. Pour toute situation nécessitant un aménagement (matériel, temporel ou pédagogique), nous vous invitons à nous contacter avant l'inscription afin d'étudier les adaptations possibles. Référent handicap : contact@lapolaris.fr

LaPolaris

TÉL. +33762584798

EMAIL contact@lapolaris.fr

WEB lapolaris.fr