

# Algorithmique et programmation structurée

3 j (21 heures)

Ref : ALGO

## Public

Apprentis développeurs, analystes évoluant vers la programmation

## Pré-requis

Rigueur logique et bonne connaissance de l'outil informatique

## Moyens pédagogiques

Formation réalisée en présentiel ou à distance selon la formule retenue  
Exposés, cas pratiques, synthèse, assistance post-formation pendant trois mois Vidéoprojecteur, support de cours fourni à chaque stagiaire

## Modalités de suivi et d'évaluation

Feuille de présence émargée par demi-journée par les stagiaires et le formateur  
Exercices de mise en pratique ou quiz de connaissances tout au long de la formation permettant de mesurer la progression des stagiaires  
Questionnaire d'évaluation de la satisfaction en fin de stage  
Auto-évaluation des acquis de la formation par les stagiaires  
Attestation de fin de formation

## Objectifs

- Disposer du vocabulaire propre à la programmation
- Traduire des besoins fonctionnels en algorithmes
- Concevoir une application
- Comprendre comment accéder aux données
- Connaître les grands paradigmes de programmation

## Programme détaillé

### INTRODUCTION

---

- Les différentes phases : sources, compilation, binaire
- Interpréteur et compilateur
- Ne pas tout réécrire (bibliothèques, éditions de liens et exécutables)

## **ALGORITHMME**

---

Les "atomes" pour s'exprimer  
Apprendre à formuler un traitement  
Utilisation d'un pseudo langage  
Exemples d'algorithmme bien conçu, "mal" conçu, et ...faux !  
Représentation avec organigramme

## **PREMIER PROGRAMME**

---

Présentation de l'environnement de développement  
Un premier programme simple en Java  
Compilation et exécution

## **VARIABLES ET TYPES DE DONNEES**

---

Pourquoi typer les variables ?  
Exemples de types (entier, réel, caractères...)  
Exemples de problèmes liés aux types  
Les opérateurs disponibles (+, /, \* / % ...)  
Le confort des tableaux et des structures  
Typage statique et dynamique

## **LA "GRAMMAIRE" D'UN LANGAGE**

---

Les blocs de programme (début ... fin)  
Le test d'une expression (si ... alors ... sinon ...)  
La boucle (tant que ...)

## **STRUCTURER SON PROGRAMME**

---

La nécessité d'utiliser des procédures ou des fonctions  
Différences conceptuelles et syntaxiques entre les deux  
Passer des arguments à une fonction (prototype, code retour)  
Les bibliothèques  
Ne pas réécrire ce qui existe déjà (librairies)  
Écrire ses propres librairies  
Comment les utiliser dans d'autres programmes

## **IMPORTANCE DE LA DOCUMENTATION**

---

Les bonnes habitudes pour anticiper les erreurs (convention de nommage)  
Les commentaires utiles

---