Introduction

- Plan du cours
- Slides

Plan du cours

- Les fondements de la programmation

1. Qu'est-ce qu'un programme ? Qu'est-ce qu'un langage ? Les différents paradigmes. Quel langage pour quelle application ?
2. Les compilateurs. Les exécutables.
3. Les responsabilités d'un programmeur.
4. Qu'est-ce qu'un algorithme ?
5. Les besoins auxquels répond un algorithme.
6. Le concept de pseudo-langage.

Travaux pratiques Présentation de différents langages (Java, C#, Visual Basic, C, C++). Ecriture d'un premier algorithme en pseudo-langage. » Genèse d'un premier programme

- Ecriture d'un programme simple : syntaxe et instructions.

1. Compilation et exécution du programme.
2. Qu'est-ce qu'une librairie ? Son rôle, son usage.
3. Travaux pratiques
4. Découverte de l'environnement de développement et d'exécution. Ecriture, compilation et exécution d'un premier programme.

- Règles de programmation

1. Convention de nommage.
2. Convention syntaxique.
3. Utilisation des commentaires. Pourquoi commenter les développements ?
4. Améliorer la lisibilité des programmes : indentation du code, découpage du code...

- Les variables

1. Qu'est-ce qu'une variable ?
2. Pourquoi typer une variable ?
3. Les types primitifs : entiers, chaînes de caractères, nombres réels, autres.
4. Déclaration, définition et initialisation d'une variable.
5. Les constantes.
6. Saisie, affichage, affectation, conversion de type.
7. Organiser ses données sous forme de tableaux.
8. Les types évolués : enregistrement, matrice, arbre.

Travaux pratiques Ecriture de plusieurs programmes simples manipulant les variables.

- Opérateurs et expressions

1. Les différents opérateurs (multiplicatif, additif, comparaison, égalité, logique, affectation).
2. Combinaison d'opérateurs.
3. Expression booléenne.
4. Travaux pratiques
5. Manipulation des opérateurs et des expressions booléennes.

- Les structures de contrôle

1. Les sélections alternatives (si, si-alors-sinon, sélection cas).
2. Les blocs d'instructions (notion de Début... Fin).
3. Les boucles itératives (tant que-répéter, répéter-jusqu'à, pour-de- à).
4. Imbrication des instructions.
5. Les commentaires.

Travaux pratiques Utilisation des structures de contrôle pour implémenter un algorithme.

- Les procédures et les fonctions

1. Définitions : procédure, fonction.
2. Pourquoi sont-elles incontournables en programmation (réutilisabilité, lisibilité...) ?
3. Le passage de paramètres.
4. Le code retour d'une fonction.
5. Sensibilisation aux limites du passage de la valeur d'une variable.
6. Notion de passage par adresse.
7. Appel de fonctions.

- Introduction à la programmation objet

1. Les concepts associés à la programmation objet : classe, attribut, méthode, argument.
2. La modélisation objet à partir des exigences fonctionnelles.
3. Introduction aux bonnes pratiques d'organisation de conception et d'organisation d'un programme.

Travaux pratiques Illustration des concepts objets.

- L'accès aux bases de données

1. Organisation et stockage des données.
2. Les traitements de base (connexion, requêtes, récupération des données).
3. Application cliente et serveur de données.
4. Affichage et manipulation des données dans l'application cliente.

Travaux pratiques Création d'un formulaire de recherche d'informations dans une base de données.

- Maintenance, débogage et test des programmes

1. Savoir lire et interpréter les différents messages d'erreurs.

2. Utiliser un débogueur : exécuter un programme pas à pas, points d'arrêts, inspecter les variables pendant l'exécution.
3. Prévoir les tests unitaires.

Travaux pratiques Utilisation d'un débogueur pour contrôler l'exécution des programmes.

Slides

Ce cour est dispensé sous forme de slides:

inj_2017.pdf