Progression Term / 2023 - 2024

Développer les compétences suivantes :

- Analyser et modéliser un problème en termes de flux et de traitement d'informations ;
- Décomposer un problème en sous-problèmes, reconnaître et réutiliser des situations ;
- Concevoir des solutions algorithmiques ;
- Traduire un algorithme dans un langage de programmation (interfaces, interactions, comprendre et réutiliser des codes sources existants, développer des processus de mise au point et de validation de programmes) ;
- Mobiliser les concepts et les technologies utiles pour assurer les fonctions d'acquisition, de mémorisation, de traitement et de diffusion des informations ;
- Développer des capacités d'abstraction et de généralisation.

Développer des compétences transversales :

- Faire preuve d'autonomie, d'initiative et de créativité ;
- Présenter un problème ou sa solution, développer une argumentation ;
- Coopérer au sein d'une équipe dans le cadre d'un projet ;
- Rechercher de l'information, partager des ressources ;
- Faire un usage responsable et critique de l'informatique.

0 Langage et programmation - Rappels Python

1. Affectation de variables
2. P-uplets (ou tuple)
3. Tableau indexé (ou list), tableau donné en compréhension, tableaux de tableaux
4. Dictionnaires par clés et valeurs (méthodes keys(), values () et items ())
5. Fonctions et appels de fonction
6. Séquences, affectation conditionnelles
7. Boucles bornées, boucles non bornées

1 Langage et programmation - Modularité

1. Utiliser de bibliothèques (travailler avec un fichier texte/csv, mesurer un temps d' exécution, dessiner des graphes, ...) ou des API
2. Exploiter la documentation
3. Créer des modules simples et les documenter.

2 Bases de données - Modèle relationnel

1. Identifier les concepts définissant le modèle relationnel (relation, attribut, domaine, clef primaire, clef étrangère, schéma relationnel)
2. Exprimer les contraintes d'intégrité (domaine, relation et référence)
3. Savoir distinguer la structure d'une base de données de son contenu
4. Repérer des anomalies dans le schéma (redondances de données, anomalies d'insertion, de suppression, de mise à jour)

3 Langages et programmation - Mise au point des programmes et gestion des bugs

1. Mettre au point des programmes (utiliser des jeux de tests – assert, Doctest, ...)
2. Reconnaître et savoir répondre aux problèmes liés au typage
3. Reconnaître et savoir répondre aux effets de bord non désirés, débordements dans les tableaux
4. Reconnaître et savoir répondre aux instruction conditionnelle non exhaustive
5. Reconnaître et savoir répondre aux choix des inégalités, comparaisons et calculs entre flottants
6. Reconnaître et savoir répondre aux mauvais nommage des variables

4 Bases de données – SGBD et langage SQL

1. Identifier les services rendus par un SGBD (persistance des données, gestion des accès concurrents, efficacité de traitement des requêtes, sécurisation des accès)
2. Identifier les composants d'une requête
3. Construire des requêtes d'interrogation à l'aide de : SELECT, FROM, WHERE, JOIN, DISTINCT, ORDER BY
4. Construire des requêtes d'insertion et de mise à jour à l'aide de : UPDATE, INSERT, DELETE

5 Structures de données – Vocabulaire de la programmation objet

1. Spécifier une structure de données par son interface
2. Écrire la définition d'une classe
3. Accéder aux attributs et méthodes d'une classe

6 Langage et programmation – Récursivité

1. Comprendre que tout programme est aussi une donnée
2. Écrire un programme récursif
3. Analyser le fonctionnement d'un programme récursif
4. Savoir répondre aux effets de bord non désirés
5. Comprendre et utiliser des décorateurs python

7 Structures de données - Arbres

1. Identifier des situations nécessitant une structure de données hiérarchique, arborescente
2. Définir un arbre en tant que structure hiérarchique (nœuds, racines, feuilles, sous-arbres)
3. Évaluer la taille, la hauteur, ... d'un arbre.

8 Algorithmique - Arbre binaire de recherche

1. Distinguer les arbres binaires
2. Parcourir un arbre de différentes façons (ordres infixe, préfixe ou suffixe ; ordre en largeur d'abord)
3. Rechercher une clé dans un arbre de recherche, insérer une clé

9 Structures de données – Structures linéaires, dictionnaires, index et clé.

1. Spécifier une structure de données par son interface
2. Distinguer interface et implémentation
3. Écrire plusieurs implémentations d'une même structure de données
4. Distinguer les Listes par le jeu des méthodes qui les caractérisent
5. Distinguer les Piles par le jeu des méthodes qui les caractérisent (mode LIFO)
6. Distinguer les Files par le jeu des méthodes qui les caractérisent (mode FIFO)
7. Choisir une structure de données adaptée à la situation à modéliser
8. Distinguer la recherche d'une valeur dans une liste et dans un dictionnaire

10 Algorithmique - Méthode « diviser pour régner »

1. Comprendre le principe de la méthode « diviser pour régner »
2. Écrire un algorithme utilisant la méthode « diviser pour régner » appliqué au tri fusion
3. Aborder les notions de coût

11 Structures de données – Graphes

1. Identifier des situations nécessitant une structure de données relationnelles ou arbre
2. Définir un graphe en tant que structure relationnelle (sommets, arcs, arêtes)
3. Distinguer graphes orientés ou non orientés
4. Écrire les implémentations correspondantes d'un graphe (classes Python, matrice d'adjacence, liste de successeurs/de prédécesseurs)
5. Passer d'une implémentation à une autre
6. Modéliser des situations sous forme de graphes (réseau routier, réseau électrique, Internet, réseaux sociaux)

12 Algorithmique – Graphes

1. Parcourir un graphe en profondeur d'abord, en largeur d'abord (exemple de parcours d' un labyrinthe)
2. Repérer la présence d'un cycle dans un graphe
3. Chercher un chemin dans un graphe
4. Modéliser à l'aide de classes Python
5. Faire un lien avec les protocoles de routage

13 Architectures matérielles OS et réseaux - Protocoles de routage

1. Distinguer tables de routage statiques et dynamiques
2. Les tables de routage étant données, identifier la route empruntée par un paquet
3. Comprendre le protocole RIP et calculer le nombre de sauts
4. Comprendre le protocole OSPF et calculer le coût de chaque route

14 Algorithmique – Programmation dynamique

1. Comprendre le principe de la programmation dynamique
2. Écrire un algorithme utilisant la programmation dynamique
3. Appliquer la programmation dynamique au problème de l' alignement de séquences
4. Appliquer la programmation dynamique au problème du rendu de monnaie

15 Architectures matérielles OS et réseaux - Sécurisation des communications

1. Décrire les principes de chiffrement symétrique (clef partagée)
2. Décrire les principes de chiffrement asymétrique (avec clef privée/clef publique)
3. Décrire l'échange d'une clef symétrique en utilisant un protocole asymétrique pour sécuriser une communication HTTPS

16 Algorithmique – Recherche textuelle

1. Étudier l'algorithme de Boyer-Moore pour la recherche d'un motif dans un texte
2. Comprendre le mécanisme de prétraitement du motif

17 Architectures matérielles OS et réseaux - Gestion des processus et des ressources par un OS

1. Décrire la création d'un processus
2. Décrire l' ordonnancement de plusieurs processus par le système
3. Mettre en évidence le risque de l'interblocage (deadlock)

18 Langage et programmation – Paradigmes de programmation (programmation impérative, fonctionnelle, objet, ...)

1. Distinguer sur des exemples les paradigmes impératif, fonctionnel et objet
2. Choisir le paradigme de programmation selon le champ d'application d'un programme
3. Choisir le paradigme de programmation selon le champ d'application d'un programme

19 Architectures matérielles OS et réseaux - Système sur puce

1. Identifier les principaux composants sur un schéma de circuit
2. Identifier les avantages de leur intégration en termes de vitesse et de consommation
3. Identifier les inconvénients

20 Langages et programmation – Calculabilité, décidabilité

1. Comprendre que la calculabilité ne dépend pas du langage de programmation utilisé
2. Montrer, sans formalisme théorique, que le problème de l'arrêt est indécidable