République Tunisienne
Ministère de l'Education

Aides pédagogiques
D'INFORMATIQUE

SECTION : SCIENCES DE L'INFORMATIQUE

Septembre 2024

NIVEAU : 2EME ANNEE
Matière : Informatique

Aide pédagogique 2024-2025

Domaines d'apprentissage	Compétences et savoirs associés détaillés	Pistes pédagogiques et directives
Pensée computationnelle et programmation	• Reconnaître les phases de résolution d'un problème. - Lire et comprendre l'énoncé d'un problème afin de dégager les tâches à réaliser. - Dégager les éléments essentiels pour la résolution (les entrées, les sorties et les traitements). - Élaborer une solution sous forme d'un algorithme. - Écrire et exécuter le programme solution sur ordinateur. - Apporter des modifications à la solution (actions correctives, actions évolutives). • Décomposer un problème en modules. - Identifier des sous-problèmes pertinents (modules). - Identifier les éléments principaux d'un module (Type, paramètres, résultat, etc.). - Acquérir la capacité de décomposer un problème en sous-problèmes : décomposition logique. • Exploiter des concepts algorithmiques pour résoudre des problèmes. - Utiliser des structures de données à bon escient : • Dégager les objets nécessaires (variables/constantes) pour résoudre un problème. • Distinguer les usages et les particularités de chaque type de données, afin	• Il est possible de faire appel à des séquences vidéo, des sites internet, divers documents ou d'une situation réelle, … pour dégager les phases de résolution d'un problème. • Il est recommandé d'utiliser des exemples concrets pour montrer les avantages de la décomposition (meilleure lisibilité, diminution de risque d'erreurs, réutilisation de modules dans un ou plusieurs algorithmes, simplicité de l'entretien, favorisation de travail en équipe). • Chaque niveau de décomposition est suivi par l'élicitation (valorisation, argumentation, justification) de sous problèmes. • L'initiation à l'algorithmique peut se faire à partir d'un algorithme existant (structure d'un algorithme et tournage à

	d'utiliser le plus adapté pour déclarer un objet nécessaire dans la résolution d'un problème donné (Entier, Réel, Booléen, Caractère, Chaîne de caractères et Tableau à une dimension). • Utiliser les structures simples pour lire des données, pour afficher des informations et pour attribuer une valeur à une variable. • Utiliser les structures de contrôle adéquates pour résoudre un problème. - Utiliser les structures conditionnelles pour effectuer des choix en fonction des circonstances. - Utiliser les structures répétitives pour répéter un ensemble d'instructions autant de fois que nécessaire. • Utiliser un langage de programmation pour implémenter une solution. - Traduire un algorithme en un programme exécutable. - Écrire un programme pour résoudre un problème. • Tester une solution implémentée afin de répondre à un besoin spécifique. - Exécuter une solution implémentée. - Modifier un code de programmation existant pour changer le comportement d'un programme.	la main). • Il est nécessaire d'habituer les apprenants à exploiter à bon escient les structures de données (Objets et types) et les structures de contrôle lors de la résolution d'un problème (nombre de variables, nombre d'instructions, structure de contrôle adéquate, etc.) • il est nécessaire de se limiter aux traitements adaptés au niveau des apprenants. • On pourra utiliser des outils d'exécution d'algorithmes tels que "Algobox", "Larp", etc. • Inciter les apprenants à comparer différents algorithmes pouvant résoudre le même problème. • Toutes les solutions des problèmes sont implémentées via le langage de programmation **Python**. • On pourra utiliser des outils tels que Trinket.io et Pencilcode.net • L'initiation à l'utilisation du langage peut se faire à partir d'un programme existant (structure d'un programme, exécution et exploration du code). • Il est possible de traduire un algorithme existant en un programme.

		• Il est utile d'inciter les apprenants à analyser un programme exécutable afin de comprendre les traitements. • Se servir de dispositifs ou de robots pour appliquer des notions de programmation, en mettant à profit différents outils et langages de programmation. • Ecrire un programme en Micro-Python ou Arduino pour programmer une carte Esp32 afin de réaliser différentes tâches. • Il est essentiel d'habituer les apprenants à commenter les solutions.
Systèmes, technologies et Internet	• Exploiter des techniques de développement pour créer des documents web. - Créer un document web ▪ S'approprier le vocabulaire et la syntaxe du langage HTML5 en créant des pages web significatives. ✓ Identifier la structure de base d'une page web : < **!doctype>, <html>, <head> et <body>** . ✓ Manipuler les éléments du langage Html5. ○ Organiser un texte de manière sémantique : <**p**>, <**br**>, <**h**$_n$>, <**mark**>, <**ol**>, <**ul**>, <**li**> <**table**>, <**tr**>, <**td**>, <**th**>, <**tfoot**>, <**thead**>, <**tbody**>. ○ Intégrer des éléments multimédias : <**img**>, <**audio**>, <**video**>. ○ Intégrer des liens : <**a**> .	• Exploiter un éditeur Web WYSIWYG qui intègre le HTML5. • Découvrir la structure de base d'un document HTML5 en explorant des pages web existantes. • Habituer les apprenants à indenter et formater correctement le code et à le commenter.

	○ Créer/Modifier un formulaire en utilisant les éléments : <**form**>, <**input**> de type (**text, number, radio, checkbox, reset, submit**), <**select**>, <**option**>, <**fieldset**>, <**legend**>. ▪ Exploiter les techniques appropriées pour appliquer des mises en forme à une page web. ✓ Reconnaître les propriétés de mises en forme (texte, bordure et arrière plan). ✓ Appliquer des styles : ○ en ligne (inline) sur des éléments d'une page web. ○ internes (head) sur des éléments d'une page web. • Reconnaître la syntaxe d'une règle CSS3 (**sélecteur, propriété** et **valeur)**. • Déclarer une règle CSS3 sur des éléments HTML. - Valider le contenu HTML5. ▪ Utiliser des outils de validation du contenu des pages web. Corriger les erreurs et les avertissements détectés par les validateurs.	• Pour chaque élément html, traiter les attributs qui lui correspondent (voir annexe). • Prévoir des activités basées sur la variation de la valeur de l'attribut ***style*** pour mettre en forme des éléments d'une page web créée. • Inciter les apprenants à découvrir la syntaxe d'une règle CSS3 (**déclaration, sélecteur**, **propriété** et **valeur**) en explorant des pages web existantes.

Recommandations générales
- Avantager les échanges et les discussions autour des solutions proposées. - Etablir des liens et trouver des fils conducteurs entre les différents domaines d'apprentissage rompant ainsi avec l'aspect linéaire de sa mise en œuvre. - Il est préconisé de présenter le contenu à enseigner via des projets, des mini-projets ou des activités, ayant un sens pour l'apprenant (jeux, simulation, ...) et stimulant chez lui l'activité, la collaboration et la créativité ; tout en favorisant l'aspect interdisciplinaire. - Il est recommandé de consulter des communautés de développement et de partager des solutions (algorithmes ou programmes) dans des espaces de partage créés pour l'échange et l'apprentissage. - Favoriser l'exploitation des ressources en ligne. - Il est important que l'apprenant conserve une trace écrite du travail réalisé en classe. Il appartient à l'enseignant de choisir le support le plus adapté à ses élèves. - Il est recommandé d'aborder les problèmes et de systématiser leurs résolutions en se basant sur les quatre composantes de la pensée computationnelle : décomposition, reconnaissance de modèles ou de formes, abstraction et algorithme.

NIVEAU : 3EME ANNEE
Matière : Algorithmique & programmation
Aide pédagogique 2024-2025

Domaine d'apprentissage	Compétences et savoirs associés détaillés	Pistes pédagogiques et directives
Pensée Computationnelle et programmation	• Exploiter des concepts algorithmiques avancés pour résoudre des problèmes. - Lire et comprendre l'énoncé d'un problème afin de dégager les tâches à réaliser. - Dégager les éléments essentiels pour la résolution d'un problème (structures et types de données, traitements). - Distinguer les usages et les particularités de chaque type de données, afin d'utiliser le plus adapté pour déclarer un objet nécessaire dans la résolution d'un problème donné. - Utiliser des structures de données avancées pour résoudre un problème (Tableau à deux dimensions, Enregistrement).	- Il est possible de faire appel à des séquences vidéo, des sites internet, divers documents ou d'une situation réelle, pour dégager l'utilité de l'utilisation des structures de données avancées. - Il est nécessaire d'habituer les apprenants à : o exploiter à bon escient les structures de données avancées (Objets et types). o appliquer les bonnes pratiques de programmation (nomenclature des objets, commentaire, etc.). - Traiter la méthode de tri par sélection et la méthode de tri à bulles - Traiter la recherche d'un élément (séquentielle et dichotomique).

	- Apporter les modifications nécessaires à une solution pour répondre à un besoin.
	- Exploiter des concepts algorithmiques avancés pour résoudre des problèmes :
	  ○ d'arithmétiques
	  ○ d'optimisation et d'approximation
	• Utiliser un environnement de programmation pour implémenter une solution.
	- Implémenter un algorithme en un programme exécutable.
	- Écrire un programme pour résoudre un problème.
	• Concevoir une interface graphique pour développer des applications simples comportant les objets suivants : fenêtre, zone de texte (Text Edit, Line Edit), bouton (Push Button), bouton radio (Radio Button), liste déroulante (Combo Box), case à cocher (Check Box), étiquette (label).
		- Traiter essentiellement :
		  ○ Des calculs arithmétiques (PGCD, PPCM, nombres premiers, décomposition en facteurs premiers, etc.)
		  ○ Des problèmes d'optimisations
		  ○ Des méthodes de calcul d'une valeur approchée de constantes connues (π, e, …)
		- Le langage adopté est Python.
		- Tester le programme solution sur ordinateur.
		- Apporter des modifications à une solution (actions correctives, actions évolutives).
		- Il est utile d'inciter les apprenants à analyser un programme exécutable afin de comprendre ses traitements.
		- La découverte d'une interface graphique peut se faire à partir d'une application existante.
		- La conception d'une interface graphique se fait en utilisant la technique « Glisser–Déposer » (Drag & drop).
		- Utiliser Qtdesigner comme outil de création d'interfaces graphiques.
		- Tenir compte des contrôles de saisie des champs d'une interface graphique en affichant des messages d'erreurs.
		- L'apprentissage est axé principalement sur la pratique.

Recommandations générales

- Avantager les échanges et les discussions autour des solutions proposées.
- Établir des liens et trouver des fils conducteurs entre les différents domaines d'apprentissage rompant ainsi avec l'aspect linéaire.
- Il est préconisé de présenter le contenu à enseigner via des projets, des mini-projets ou des activités utiles ayant un sens pour l'apprenant (jeux, simulation, ...) afin de stimuler l'activité, la collaboration et la créativité chez l'apprenant et favorisant l'aspect interdisciplinaire.
- Il est possible de faire appel à des séquences vidéo, des sites internet, divers documents ou d'une situation réelle, pour dégager l'utilité de l'utilisation des structures de données avancées.
- Il est recommandé de consulter des communautés de développement et de partager des solutions (algorithmes ou programmes) dans des espaces de partage créés pour l'échange et l'apprentissage.
- Favoriser l'exploitation des ressources en ligne.
- Il est important que l'apprenant conserve une trace écrite du travail réalisé en classe. Il appartient à l'enseignant de choisir le support le plus adapté à ses élèves.
- Il est recommandé d'aborder les problèmes et de systématiser leurs résolutions en se basant sur les quatre composantes de la pensée computationnelle : décomposition, reconnaissance de modèles ou de formes, abstraction et algorithme.
- Le langage adopté est Python.

NIVEAU : 4EME ANNEE
Matière : Algorithmique & programmation
Aide pédagogique 2024-2025

Domaines d'apprentissage	Compétences et savoirs associés détaillés	Pistes pédagogiques et directives
Pensée computationnelle et programmation	• Exploiter des concepts algorithmiques avancés pour résoudre des problèmes faisant appel à : ○ des structures de données ▪ Fichier texte ▪ Fichier typé ○ des méthodes de tri ▪ Tri par insertion ▪ Tri shell ○ la récursivité	- Rappeler les notions tableau à deux dimensions et enregistrement à travers la résolution de problème exploitant ces structures de données. - Traiter les notions fichier texte et fichier typé à travers la résolution de problème exploitant ces structures de données. - Utiliser la mémoire centrale pour effectuer les traitements modifiant le contenu initial d'un fichier tels que le tri, l'insertion d'un élément, la suppression d'un élément, le décalage, … - Rappeler les deux méthodes de tri : Le tri par sélection et le tri à bulles. - On pourra traiter d'autres algorithmes de tri (tri par création, tri par comptage, ...). - Montrer, quand c'est possible, le passage d'une formulation itérative à une formulation récursive. - Ne traiter que le cas de récursivité simple (ni croisée, ni indirecte) sur des problèmes naturellement récursifs (factorielle, palindrome, PGCD, …)

	○ des traitements récurrents et arithmétiques	- On traitera divers problèmes en axant sur la relation de récurrence d'ordre un et plus (suites, triangle de pascal, le nombre d'or, …) - Traiter essentiellement : ○ la suite de Fibonacci ○ calcul du factoriel, PGCD, PPCM, nombre premier, décomposition en facteurs premiers. ○ les conversions entre bases de numération ○ les calculs de C (n,p) et de A(n,p)
	○ l'optimisation et l'approximation	- Traiter essentiellement : ○ des problèmes d'optimisations ○ la recherche du zéro d'une fonction (f(x)=0) ○ la recherche du point fixe d'une fonction (f(x)=x) ○ des méthodes de calcul d'une valeur approchée de constantes connues (π, e, …) ○ calcul d'aires (rectangles, trapèzes)
	• Utiliser un environnement de programmation pour implémenter une solution. ○ Implémenter un algorithme en un programme exécutable. ○ Écrire un programme pour résoudre un problème.	- Le langage adopté est Python. - Il est utile d'inciter les apprenants à analyser un programme exécutable afin de comprendre les traitements. - Il est essentiel d'habituer les apprenants à commenter les solutions. - Apporter des modifications à une solution (actions correctives, actions évolutives).

	○ Concevoir une interface graphique pour développer des applications simples comportant les objets suivants : ▪ List Widget ▪ Table Widget	- Tester le programme solution sur ordinateur. - Rappeler les composants d'une interface graphique à partir d'une application existante (fenêtre, Label, liste déroulante (Combo Box), Zone de texte (Text Edit, Line Edit), bouton (Push Button), bouton radio (Radio Button), Case à cocher (Check Box)). - La conception d'une interface graphique se fait en utilisant la technique « Glisser–Déposer » (Drag & drop). - Utiliser Qtdesigner comme outil de création d'interfaces graphiques. - Tenir compte des contrôles de saisie des champs d'une interface graphique en affichant des messages d'erreurs. - Il est recommandé d'utiliser des fichiers pour transférer et récupérer des informations. - L'apprentissage est axé principalement sur la pratique.

Recommandations générales
- Avantager les échanges et les discussions autour des solutions proposées. - Il est possible de faire appel à des séquences vidéo, des sites internet, divers documents ou d'une situation réelle, pour dégager l'utilité de l'utilisation des structures de données avancées. - Il est judicieux d'utiliser la pédagogie active et de traiter divers problèmes de la vie courante (mathématiques, physiques, économies …) - Il est recommandé d'aborder les problèmes et de systématiser leurs résolutions en se basant sur les quatre composantes de la pensée computationnelle : décomposition, reconnaissance de modèles ou de formes, abstraction et algorithme. - Il est recommandé de consulter des communautés de développement et de partager des solutions (algorithmes ou programmes) dans des espaces de partage créés pour l'échange et l'apprentissage. - Favoriser l'exploitation des ressources en ligne. - Il est important que l'apprenant conserve une trace écrite du travail réalisé en classe. Il appartient à l'enseignant de choisir le support le plus adapté à ses élèves. - Le langage adopté est Python.