

République Tunisienne
Ministère de l'Education

A decorative graphic on the left side of the page consists of several overlapping, semi-transparent blue squares of various sizes. Some squares have a gradient from light to dark blue, while others are solid. Small white dots are scattered across the squares, giving a digital or data-like appearance. The squares are arranged in a cluster that tapers towards the top right.

CURRICULUM D'INFORMATIQUE SPECIFIQUE

SECTIONS : SCIENCES EXPERIMENTALES,
SCIENCES TECHNIQUES
ET MATHEMATIQUES

Septembre 2021

NIVEAU : 2^{EME} ANNEE

Aide pédagogique 2021-2022

| Domaine d'apprentissage | Savoirs associés | Pistes pédagogiques et directives |
|--|--|--|
| Pensée Computationnelle et programmation | <ul style="list-style-type: none">● Identifier les étapes de résolution d'un problème.<ul style="list-style-type: none">○ Dégager les éléments essentiels permettant la résolution d'un problème.● Élaborer des solutions sous forme d'algorithmes.<ul style="list-style-type: none">○ Utiliser des structures de données à bon escient.○ Utiliser les structures de contrôle adéquates pour résoudre un problème.○ Traiter essentiellement les notions suivantes :<ul style="list-style-type: none">▪ Les objets (constantes et variables)▪ Les types de données (entier, réel, caractère, booléen, chaîne).▪ Les structures simples. | <ul style="list-style-type: none">● Etablir des liens et trouver des fils conducteurs entre les différents domaines d'apprentissage rompant ainsi avec l'aspect linéaire du programme.● Il est préconisé de présenter les savoirs associés à travers des projets, des mini-projets ou des activités ayant du sens pour l'apprenant.● Favoriser l'investigation, le questionnement, l'apprentissage expérientiel, l'apprentissage par problème. etc.● Il est fortement recommandé d'opter pour une démarche de création au cours de laquelle les apprenants développent leur autonomie, leur créativité et leur imagination, mais aussi le sens du travail collaboratif.● Donner du sens aux activités, les diversifier et opter pour une démarche interdisciplinaire permettant le décroisement entre les divers champs d'apprentissages et l'ouverture de l'informatique sur les autres disciplines.● Favoriser l'exploitation des ressources en ligne et développer la communication.● Inciter à l'innovation et motiver les apprenants pour la créativité.● Inciter les apprenants à faire des échanges autour des solutions proposées et de les partager en ligne.● Il est pertinent de :<ul style="list-style-type: none">○ familiariser l'apprenant à formuler sous forme d'actions des solutions à des problèmes puisés de son vécu. On pourra exprimer ces solutions à |

| | | |
|--|--|---|
| | <ul style="list-style-type: none"> ▪ La structure de contrôle conditionnelle (Si ... Alors ; Si ... Sinon). ▪ La structure de contrôle répétitive (Pour). <p>• Exploiter un environnement de programmation</p> <ul style="list-style-type: none"> ○ Implémenter un algorithme en utilisant un langage de programmation. ○ Tester et valider un programme. | <p>l'aide d'un schéma, d'un organigramme, d'une carte heuristique, d'un pseudocode, etc.</p> <ul style="list-style-type: none"> ○ inviter les apprenants à déterminer les entrées, les sorties et les traitements. ○ inciter les apprenants à vérifier la validité d'une solution donnée par rapport à l'énoncé d'un problème. <p>• Il est recommandé d'inciter les apprenants à :</p> <ul style="list-style-type: none"> ○ étudier quelques séquences algorithmiques (décrire, comparer, déterminer le rôle, etc.). ○ modifier des algorithmes existants pour changer leurs comportements. ○ corriger les erreurs de logique dans une séquence algorithmique afin de parvenir aux résultats souhaités. ○ évaluer différentes solutions algorithmiques d'un même problème donné. <p>• Il est préconisé :</p> <ul style="list-style-type: none"> ○ de familiariser l'apprenant avec un environnement de programmation. ○ d'inciter l'apprenant à : <ul style="list-style-type: none"> • réutiliser des codes sources existants. • modifier un programme existant pour obtenir un résultat différent. • écrire un programme pour résoudre un problème. <p>• Toutes les solutions des problèmes sont implémentées via le langage de programmation Python.</p> |
|--|--|---|

- **Prendre conscience de l'intérêt de la robotique**

- Définir la robotique.
- Identifier des domaines d'application de la robotique.

- **Piloter un objet connecté**

- Connaître les interfaces de l'objet à connecter.
- Savoir connecter un objet à l'ordinateur
- Programmer des objets simples virtuels ou réels pour réaliser différentes tâches d'une façon innovante.

- Il est nécessaire de présenter la robotique et d'expliquer ses fondements en s'appuyant sur des séquences vidéo, des ressources numériques, des études de cas, etc.

- L'apprenant n'est pas appelé à développer une application de commande.

- Utiliser **Micro-Python / Arduino** pour programmer la carte.

- Traiter les activités telles que :

- Faire clignoter une diode Led.
- Feu de carrefour (rouge, vert, orangé).

NIVEAU : 3^{EME} ANNEE

Aide pédagogique 2021-2022

| Domaine d'apprentissage | Savoirs associés | Pistes pédagogiques et directives |
|---|---|--|
| <p>Pensée Computationnelle et programmation</p> | <ul style="list-style-type: none"> ● Utiliser les structures algorithmiques adéquates pour résoudre un problème. <ul style="list-style-type: none"> ○ Les objets (constantes et variables). ○ Les types de données (entier, réel, caractère, booléen, chaîne de caractères). ○ Les structures simples. ○ Les structures de contrôle (Si, Selon, Pour, Tant Que et Répéter). ○ Les tableaux à une dimension. ● Elaborer des solutions algorithmiques modulaires. <ul style="list-style-type: none"> ○ Analyser un problème. ○ Acquérir la capacité de décomposer un problème en modules. ○ Identifier les éléments principaux d'un module (paramètres, résultat, type, portée des objets). | <ul style="list-style-type: none"> ● Exprimer les solutions, selon les besoins, sous forme d'un organigramme, d'une carte mentale, d'un pseudocode, etc. ● Inciter les apprenants à choisir les structures de données et les structures de contrôle adéquates. ● Concevoir des solutions algorithmiques. ● Choisir des exemples concrets pour montrer les avantages de la décomposition modulaire. ● Argumenter et justifier les choix de la modularité. ● Inciter les apprenants à écrire des solutions modulaires. ● Il est conseillé d'habituer les apprenants à commenter les solutions proposées. ● Utiliser des modules prédéfinis et implémenter des modules personnels. ● Décrire correctement les entrées / les sorties et le rôle de chaque module. |
| | <ul style="list-style-type: none"> ● Utiliser un langage de programmation pour implémenter une solution. <ul style="list-style-type: none"> ○ Analyser un programme existant et lui apporter des modifications pour améliorer sa fonctionnalité. ○ Implémenter un algorithme en un programme. | <ul style="list-style-type: none"> ● Les solutions des problèmes sont implémentées via le langage de programmation Python. ● L'apprentissage se fait à travers un projet ou des mini projets faisant appel essentiellement aux traitements suivants : |

| | | |
|---|--|--|
| | <ul style="list-style-type: none"> ○ Écrire un programme pour résoudre un problème. ○ Tester, corriger, améliorer et valider un programme. | <ul style="list-style-type: none"> ○ Calculs arithmétiques tels que : PGCD, PPCM, nombres premiers, décomposition en facteurs premiers. ○ Tri d'un tableau (une méthode de tri). ○ Recherche d'un élément dans un tableau. |
| <p>Internet et Nouvelles technologies</p> | <ul style="list-style-type: none"> ● Prendre conscience de l'intérêt de la robotique <ul style="list-style-type: none"> - Définir la robotique. - Identifier des domaines d'application de la robotique. ● Piloter un objet connecté <ul style="list-style-type: none"> - Connaître les interfaces de l'objet à connecter. - Savoir connecter un objet à l'ordinateur - Programmer des objets simples virtuels ou réels pour réaliser différentes tâches d'une façon innovante. | <ul style="list-style-type: none"> ● Il est nécessaire de présenter la robotique et d'expliquer ses fondements en s'appuyant sur des séquences vidéo, des ressources numériques, des études de cas, etc. ● L'apprenant n'est pas appelé à développer une application de commande. ● Utiliser Micro-Python / Arduino pour programmer la carte. ● Traiter les activités telles que : <ul style="list-style-type: none"> ○ Faire clignoter une diode Led. ○ Feu de carrefour (rouge, vert, orangé). |

NIVEAU : 4^{EME} ANNEE

Les apprentissages à développer en termes de compétences

| Domaine d'apprentissage spécifique | Compétences disciplinaires à développer | Savoirs associés (savoir, savoir-faire et savoir-être) | Compétences de vie visées |
|--|---|---|---|
| Pensée Computationnelle et programmation | <ul style="list-style-type: none">• <i>Elaborer des solutions algorithmiques pour résoudre un problème.</i>• <i>Exploiter un environnement de programmation pour implémenter une solution.</i> | <ul style="list-style-type: none">• <i>Utiliser les nouvelles structures algorithmiques pour résoudre un problème.</i>• <i>Utiliser un langage de programmation pour développer une solution.</i>• <i>Mobiliser les compétences développées dans des mini-projets concrets et variés.</i> | <p><i>La résolution de problèmes</i></p> <p><i>L'auto-évaluation</i></p> <p><i>La communication et la collaboration</i></p> <p><i>L'innovation et la créativité</i></p> |

NIVEAU : 4^{EME} ANNEE

Aide pédagogique 2021-2022

| Domaine d'apprentissage | Savoirs associés | Pistes pédagogiques et directives |
|---|---|--|
| <p>Pensée Computationnelle et programmation</p> | <p>Utiliser les structures algorithmiques pour résoudre un problème</p> <ul style="list-style-type: none"> • <i>Utiliser les structures algorithmiques adéquates pour résoudre un problème.</i> <ul style="list-style-type: none"> ○ Les types de données standards. <ul style="list-style-type: none"> ▪ Entier, réel, booléen, caractère, chaîne. ○ Les structures de contrôle conditionnelles. <ul style="list-style-type: none"> ▪ si ... alors ▪ si ... sinon ▪ selon | <ul style="list-style-type: none"> • 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 préconisé de présenter le contenu à enseigner via des activités et/ou des mini-projets afin de stimuler l'activité, la collaboration et la créativité chez l'apprenant. • Il est judicieux de proposer un contenu motivant ayant un sens pour l'apprenant (situation problème, jeux, simulation, ...) et favorisant ainsi l'aspect interdisciplinaire. • Favoriser l'exploitation des ressources en ligne. • Inviter les apprenants à participer à des communautés de développement et de partage de solutions pour une autoformation, pour trouver des réponses à des questionnements ou pour l'enrichir avec leurs productions. • Exprimer les solutions sous forme d'un pseudocode. • Habituer les apprenants à dégager, à partir d'un énoncé, les mots clés permettant de dégager les tâches à réaliser et de déterminer les entrées, les sorties et les pistes des traitements nécessaires pour établir une solution à un problème donné. • Dégager les éléments essentiels pour la résolution d'un problème (structures algorithmiques, types de données, traitements, etc.). • Analyser une solution existante et identifier les rôles de différentes structures utilisées. • Inciter les apprenants à expliquer et à argumenter une séquence d'instructions afin de comprendre le traitement. |

| | | |
|--|---|---|
| | <ul style="list-style-type: none"> ▪ des structures imbriquées ○ <i>Les structures de contrôle répétitives.</i> ▪ pour ... faire ▪ répéter ... jusqu'à ▪ tant que ... faire ▪ des boucles imbriquées ○ <i>Les tableaux à une dimension</i> • Elaborer des solutions algorithmiques modulaires. ○ <i>Analyser et modéliser un problème.</i> ○ <i>Acquérir la capacité de décomposer un problème en sous-problèmes.</i> ○ <i>Identifier les éléments principaux d'un module (l'entête, paramètres,</i> | <ul style="list-style-type: none"> • Inviter les apprenants à identifier, pour un problème donné, une solution parmi plusieurs programmes proposés. • Distinguer les usages et les particularités de chaque type de données. • 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 algorithmiques. • Inviter les apprenants à concevoir des solutions algorithmiques. • Inviter les apprenants à corriger une séquence d'instructions ou une solution erronée. • Avantager les échanges et les discussions autour des solutions proposées. • Apporter les modifications nécessaires à un programme existant pour obtenir un résultat différent. • Inciter les apprenants à identifier et à choisir les structures de données et les structures de contrôle adéquates. • Écrire une solution utilisant les structures conditionnelles. • Écrire une solution utilisant les structures itératives. • Argumenter et justifier les choix des différentes structures. • Il est conseillé d'habituer les apprenants à commenter les solutions proposées. • L'apprentissage se fait à travers la résolution de problèmes élémentaires, par exemple : <ul style="list-style-type: none"> ○ traitement simples sur les chaînes. ○ traitement simples sur les objets numérique. ○ somme d'une suite arithmétique. ○ recherche d'un élément. ○ recherche de nombres premiers. • Analyser et modéliser un problème, le découper en sous-problèmes plus faciles à résoudre. |
|--|---|---|

| | | |
|--|--|--|
| | <p><i>résultat, type du module, portée des objets, appel).</i></p> | <ul style="list-style-type: none"> • Choisir des exemples concrets pour montrer les avantages de la décomposition modulaire. • Élaborer des solutions algorithmiques modulaires. • Inciter les apprenants à écrire des solutions modulaires. • Argumenter et justifier les choix de la modularité. • Utiliser diverses techniques de décomposition (Ascendante, descendante, etc.). • Utiliser des modules prédéfinis et implémenter des modules personnels. • Ecrire des solutions modulaires à un problème donné et réutiliser des modules déjà programmés. • Décrire correctement les entrées / les sorties et le rôle de chaque module. • Choisir les données appropriées pour tester chaque module. • Inscrire le développement des programmes dans un travail collaboratif. • L'apprentissage se fait à travers la résolution de problèmes modulaires. • Comme exemples de projets ou de mini projets, on peut citer : <ul style="list-style-type: none"> ○ somme des carrés des premiers entiers naturels ○ tri d'un tableau (au moins deux différents principes de tri). ○ calcul de combinaison. ○ triangle de Pascal ○ somme des premiers entiers naturels ○ recherche d'un élément dans un tableau. ○ calcul arithmétique (PGCD, PPCM, nombres premiers, décomposition en facteurs premiers, etc.) • Collaborer efficacement au sein d'une équipe dans le cadre d'un projet. |
|--|--|--|

Exploiter un langage de programmation pour implémenter une solution

- Implémenter un algorithme en un programme.
- Écrire un programme pour résoudre un problème.
 - **Les types de données standards.**
 - **Les structures de contrôle.**
 - **Les tableaux à une dimension.**
- Tester, corriger, améliorer et valider un programme sur machine.
- Utiliser les fonctions prédéfinies de base

- Les solutions des problèmes sont implémentées cette année via le langage de programmation **Python**.
- L'apprentissage est axé principalement sur la pratique.
- Se familiariser avec l'environnement de programmation.
- Écrire un programme en respectant les contraintes de la syntaxe.
- Repérer l'erreur de syntaxe dans un programme existant.
- Tester, corriger et modifier un programme.
- Utiliser ses capacités de raisonnement, ses connaissances sur le langage de programmation, ses savoirs faire et à des outils variés pour améliorer sa solution.
- Savoir s'autoévaluer et être capable de décrire ses intérêts, ses compétences et ses acquis.
- Amener l'apprenant à effectuer une recherche et à présenter les structures de données et de contrôle en python.
- Il est nécessaire d'habituer les apprenants à exploiter à bon escient les structures de données (Objets et types).
- Habituer les apprenants à appliquer les bonnes pratiques de programmation (nomenclature des objets, indentation de la solution, commentaire, etc.).
- Lors de l'implémentation, l'apprenant devrait se conformer aux recommandations de l'enseignant quant à l'usage des méthodes, étapes, formules...
- Comme exemples de projets ou de mini projets, on peut citer :
 - **nombre paire ou impaire.**
 - **résolution d'une équation du second degré.**
 - **nombre d'occurrences du maximum dans un tableau.**
 - **nombres amis.**
 - **nombres d'Amstrong.**

| | | |
|--|--|---|
| | <ul style="list-style-type: none"> ▪ sqrt, abs, around, trunc, randint, floor, chr, ord, int ▪ upper, lower, find, replace, count, len, val, str, isalpha, isdigit, eval, slice, find ○ <i>Concevoir une interface graphique pour développer des applications simples.</i> ▪ Utiliser les objets graphiques (widgets) les plus usuels (boîte de dialogue, zone texte, label, bouton, bouton radio, liste, liste déroulante, case à cocher). | <ul style="list-style-type: none"> • L'interface graphique est utilisée avec le langage de programmation Python pour développer des applications simples. • Installer le logiciel Qt designer et les bibliothèques nécessaires. • Concevoir et utiliser une interface utilisateur graphique (GUI) • La découverte d'une interface graphique peut se faire à partir d'une application existante. • Initier les apprenants au principe de la programmation événementielle. • La conception d'une interface graphique peut se faire en utilisant la technique « Glisser-Déposer » et la programmation des objets se fait à l'aide du langage python. • Amener les apprenants à importer, installer et utiliser des bibliothèques. • Motiver les apprenants par la création des jeux. • Comme exemples de projets ou de mini projets, on peut citer : <ul style="list-style-type: none"> ○ convertisseur de devises. ○ convertisseur de bases. ○ message en morse. ○ réalisation d'une calculatrice arithmétique. • L'usage de l'interface graphique devrait être fait dans le but d'améliorer l'ergonomie de quelques programmes, mais en aucun cas un objet principal d'apprentissage. • L'ergonomie concerne les composants graphiques simples d'une fenêtre permettant la saisie des données et l'affichage des résultats (bouton, label ...). |
|--|--|---|