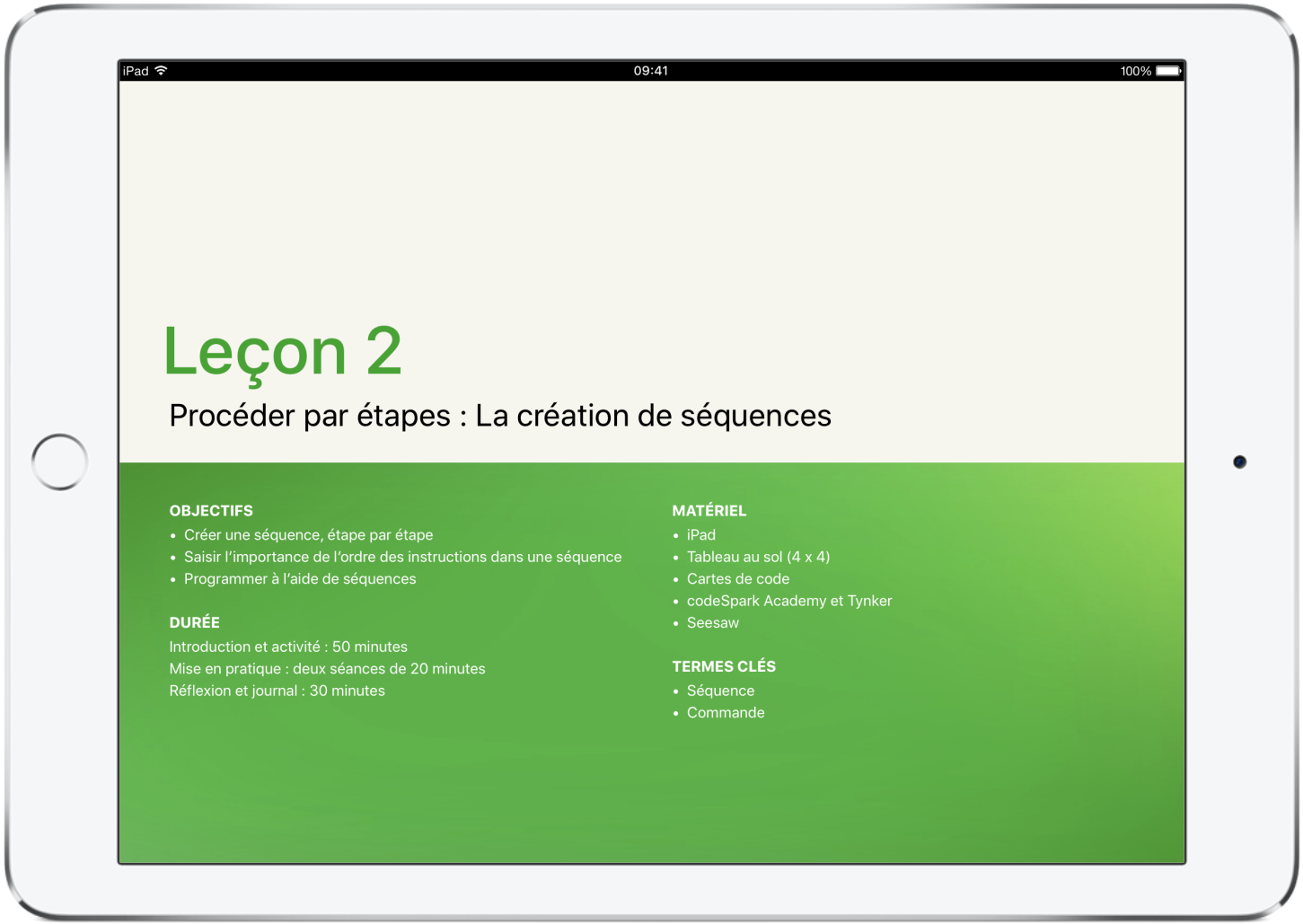
# Guide de ressources Premiers pas en programmation

Mai 2017

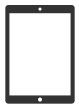
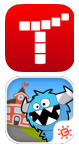


# La programmation pour tous

La technologie possède un langage. La programmation, ou le code, est la façon de le parler. Et nous croyons qu'il s'agit d'une compétence essentielle. En apprenant à programmer, on apprend aussi à résoudre des problèmes et à travailler en équipe de façon créative. Par-dessus tout, on maîtrise les outils nécessaires pour transformer une simple idée en app. Selon nous, tout le monde devrait avoir la possibilité de créer quelque chose qui peut changer le monde. Nous avons donc conçu un nouveau programme qui offre des outils et des ressources permettant à tous d'apprendre et d'enseigner la programmation.

# La programmation pour tous

La programmation pour tous comprend une multitude de ressources qui guident les élèves, de leurs premiers pas en programmation à la création de leur toute première app. Le tableau ci-dessous donne un aperçu des ressources éducatives qui vous sont offertes gratuitement.

| Cours | Appareil | Public cible | App | Prérequis | Aperçu | Matériel pédagogique | Ressources d'aide | Nombre d'heures de cours |
|---|---|---|---|---|---|---|---|---|
| |  | De la maternelle à la 2e année |  | Aucun | Commencer à penser comme les programmeurs grâce à des activités pratiques sur des concepts clés dans des apps de programmation visuelle. | • Leçons de l'app codeSpark Academy<br>• Cours Cadet de l'espace de l'app Tynker | • Premiers pas en programmation 1 – Guide de l'enseignant | 20 heures, y compris les activités du guide de l'enseignant et les leçons de l'app |
| | | De la 3e à la 5e année | | Aucun | Apprivoiser des concepts de programmation fondamentaux et s'entraîner à penser comme les programmeurs. | • Cours Sorts de dragon de l'app Tynker | • Premiers pas en programmation 2 – Guide de l'enseignant | 20 heures, y compris les activités du guide de l'enseignant et les leçons de l'app |
| | | Dès l'école intermédiaire ou secondaire | | Aucun | Apprendre les bases de la programmation en utilisant le code Swift. | • App Swift Playgrounds<br>• Leçons Apprendre à coder 1 et 2<br>• Cours iTunes U* | • Apprendre à coder 1 et 2 – Guide de l'enseignant<br>• Badges Apple Teacher Learning Center – Swift Playgrounds | 45 heures, y compris les activités du guide de l'enseignant et les leçons Apprendre à coder 1 et 2 |
| | | Dès l'école secondaire | | Aucun | Acquérir une expérience pratique des outils, techniques et concepts nécessaires pour concevoir de A à Z une app iOS simple. | Livre Introduction au développement d'apps avec Swift et fichiers de projets | • Introduction au développement d'apps avec Swift – Guide de l'enseignant | 90 heures |

* En anglais seulement.

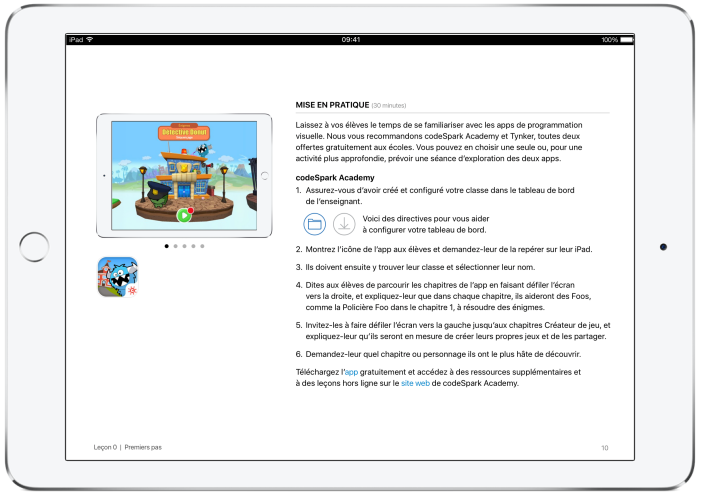
# Aperçu

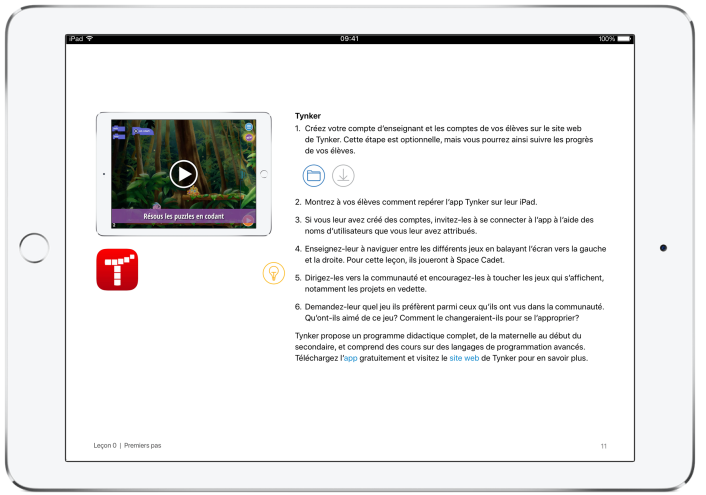
Les premières années d'école constituent un bon moment pour s'initier à la programmation. Ses concepts amènent les élèves à envisager le quotidien et le monde numérique sous un autre angle, et leur permettent d'acquérir les bases de la pensée informatique. Les apps de programmation spécialement conçues pour les jeunes élèves, comme codeSpark Academy et Tynker, proposent des puzzles visuels pour encourager la persévérance, la créativité et le développement d'habiletés en résolution de problèmes. Destinée aux élèves de cinq à sept ans, codeSpark Academy comporte une interface sans texte, ce qui permet aux enfants ne savant pas encore lire, aux apprenants de l'anglais et aux élèves ayant des difficultés en lecture de jouer aussi. Et avec Tynker, les élèves âgés de cinq à onze ans explorent d'abord des blocs de code visuels, puis progressent vers la programmation par texte : ils résolvent des puzzles, conçoivent des projets et commandent des drones et des robots.

## En classe

Les apps Tynker et codeSpark Academy, de même que les leçons des guides de l'enseignant Premiers pas en programmation, sont conçus pour vous aider à enseigner la programmation dès les premières années du primaire. Les leçons présentent les concepts clés de la programmation, tout en illustrant en quoi le mode de pensée qui les sous-tend s'applique à d'autres sphères de l'apprentissage et du quotidien.

Ces guides vous donnent tout le soutien nécessaire pour aider vos élèves à résoudre les puzzles, quelle que soit votre expérience en programmation. Activités connexes, questions de réflexion, idées de journal de bord, grille d'évaluation et plus encore vous aideront à favoriser chez vos élèves une compréhension approfondie de la matière. Vous pouvez enseigner les leçons dans leur intégralité ou par sections. En annexe, des tables de corrélation indiquent la conformité préliminaire des leçons avec les normes intérimaires d'enseignement informatique établies par la Computer Science Teachers Association (CSTA) pour le niveau 1.



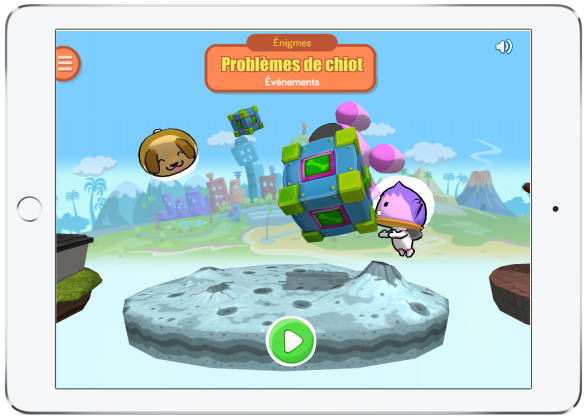
# Principales caractéristiques

## codeSpark Academy

**Apprentissage.** Dans ce jeu sans texte, les élèves résolvent des puzzles pour s'initier à des concepts de base de l'informatique, comme le séquencement, les boucles, les instructions conditionnelles et plus encore.

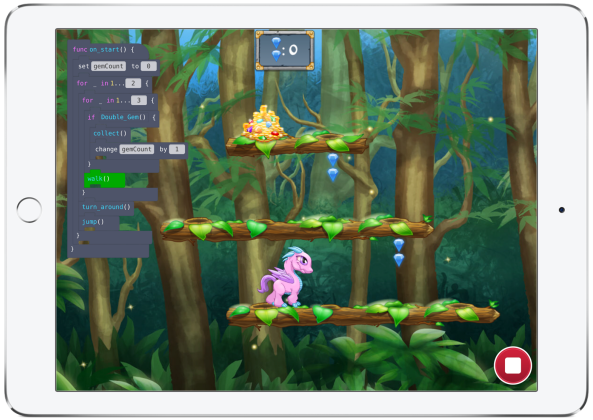
**Création.** Les élèves mettent leurs connaissances en pratique en créant leurs propres projets de programmation dans le Créateur de jeu.

**Autonomie.** Aucune expérience en programmation n'est requise pour enseigner, apprendre ou jouer. Les ressources et le tableau de bord de l'enseignant, qui comprend les rapports des progrès des élèves, sont offerts gratuitement aux éducateurs, dans 10 langues différentes.




## Tynker

**Environnement de programmation.** À leur propre rythme, les élèves résolvent des puzzles de programmation structurés pour assimiler les concepts et les mettre en pratique de façon créative.



**Évaluation automatique.** Le tableau de bord de l'enseignant analyse les puzzles, les quiz et les exercices de programmation réalisés par les élèves afin d'évaluer leur maîtrise des compétences.

**Nouvelle fonctionnalité Swift.** Les élèves peuvent passer des blocs de code visuels aux blocs de code Swift dans les puzzles; ils se familiarisent avec Swift et se préparent aux niveaux de programmation plus avancés.

## Guides de l'enseignant

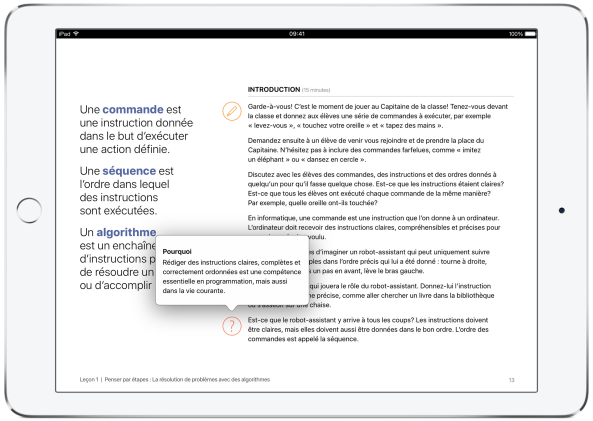
**Fichiers téléchargeables.** Des modèles de documents pour les activités des élèves et des présentations Keynote favorisent l'apprentissage en classe.

**Corrigés.** Les solutions aux puzzles de Tynker et codeSpark Academy vous permettent d'aider les élèves qui sont bloqués.

**Exemples de travaux d'élèves.** Voyez les résultats attendus des activités.

**Réflexions.** Les questions et sujets de réflexion fournis vous aident à renforcer le lien entre l'application du concept en programmation et dans la vie de tous les jours.

**Astuces et exemples.** Des idées pour approfondir et simplifier les leçons sont incluses.



# Structure des cours

## Premiers pas en programmation 1

Grâce à des activités pratiques et interactives où ils appliquent des concepts de programmation à des situations du quotidien, les élèves commencent à penser comme des programmeurs. Ils découvrent les commandes, les séquences, les boucles, les événements et les algorithmes. Ensemble, ils s'exercent à prédire le résultat du code et à déboguer leur propre code et celui des autres. Ils utilisent les compétences acquises dans les apps de programmation visuelle, résolvent des puzzles et conçoivent des programmes.

**Leçon 0 – Premiers pas.** Vous évaluez les connaissances des élèves en programmation, leur présentez les principales apps dont ils se serviront au fil des leçons et organisez le tableau d'apprentissage de la classe.

**Leçon 1 – Mettre de l'ordre : Introduction au séquencement.** Les élèves découvrent les séquences à l'aide d'exemples du quotidien, créent une séquence en s'inspirant d'une histoire connue et utilisent des séquences simples pour résoudre des puzzles dans des apps de programmation visuelle.

**Leçon 2 – Procéder par étapes : La création de séquences.** En comprenant l'importance de l'ordre des étapes dans une séquence, les élèves apprennent que les mêmes actions peuvent être placées dans un autre ordre pour former une séquence différente, puis ils créent une drôle de danse. Ils découvrent de nouvelles commandes et résolvent des problèmes plus complexes dans les apps de programmation.

**Leçons 3 – Faire des choix : Le séquencement modulable.** Les élèves apprennent que l'ordre de certaines étapes d'une séquence peut être modifié, puis ils créent leurs propres séquences modulables. Ils découvrent qu'ils peuvent résoudre un puzzle de plusieurs façons et partagent leurs solutions avec leurs camarades.

**Leçon 4 – Répéter à l'infini : Les boucles.** Les élèves trouvent des boucles dans des situations du quotidien, puis les appliquent de façon créative au cours d'une activité de percussions corporelles. Ils voient comment les boucles sont représentées en programmation et s'en servent pour simplifier l'écriture de leur code.

**Leçon 5 – Corriger les erreurs : Le débogage.** En saisissant l'importance de la persévérance et du débogage dans divers contextes, les élèves évaluent et utilisent leurs nouvelles compétences de programmation pour résoudre un problème et déboguer les solutions de leurs camarades. Ils s'exercent à prédire le résultat du code qu'ils écrivent et à y repérer les bogues quand il ne fonctionne pas comme prévu.

**Leçon 6 – Déclencher le processus : Les événements et les actions.** Les élèves explorent comment les événements rendent leur expérience de jeu et leur code plus dynamiques et réactifs, et apprennent à s'en servir en programmation. Ils réfléchissent aux différentes façons de déclencher des événements au quotidien, puis créent une télécommande de robot pour s'exercer à appeler des événements.

**Leçon 7 – Respecter les conditions : Les instructions *if*.** Les élèves sont initiés aux instructions conditionnelles et réfléchissent à la place des instructions *if* dans leur quotidien. Ils repèrent celles-ci dans des règles de jeux de société connus. Ils programment en utilisant des instructions *if* et créent du code qui réagit davantage aux conditions imposées par l'environnement.

**Leçon 8 – Résoudre les problèmes : Les algorithmes.** Les élèves consolident leurs acquis et conçoivent des algorithmes composés de séries d'étapes leur permettant de résoudre des problèmes. Ils commencent par comprendre le processus d'une recette simple, puis créent et programment leur propre labyrinthe interactif.

# Structure des cours (suite)

## Premiers pas en programmation 2

Au fil des leçons, les élèves apprivoisent des concepts de programmation fondamentaux et s'entraînent à penser comme des programmeurs. En plus de découvrir les algorithmes, les fonctions, les boucles, les instructions conditionnelles et les variables, ils se familiarisent avec les bases de la conception d'interface. Seuls ou en équipe, ils renforcent leurs compétences de programmation en résolvant de véritables problèmes, en testant le code de leurs camarades et en concevant des programmes pour différents robots. Ils développent aussi leurs habiletés dans Tynker en trouvant des solutions à des problèmes variés et en appliquant les concepts qu'ils ont appris au cours des activités.

**Leçon 0 – Premiers pas.** Vous évaluez les connaissances des élèves en programmation, leur présentez une app de programmation visuelle comme Tynker et configurez leurs portfolios numériques au moyen d'un app comme Seesaw.

**Leçon 1 – Penser par étapes : La résolution de problèmes avec des algorithmes.** Les élèves apprennent qu'un algorithme est une suite de commandes permettant de résoudre un problème ou d'effectuer une tâche. Dans l'app Tynker, ils choisissent un dragon, puis créent des algorithmes pour résoudre des puzzles et ainsi développer leurs capacités de séquencement. Ils conçoivent et testent des algorithmes au cours d'activités en classe.

**Leçon 2 – Penser aux solutions : Le débogage.** Les élèves apprennent à détecter et à corriger des erreurs dans leurs algorithmes et leurs programmes. Dans Tynker, ils modifient les algorithmes qui comportent des bogues pour créer un programme permettant de résoudre les puzzles.

**Leçon 3 – Penser en rond : Les boucles.** Les élèves apprennent à considérer les boucles comme patrons, puis conçoivent et testent un algorithme pour créer un serpent bouclé. Dans Tynker, ils utilisent les boucles *for* pour résoudre des puzzles en repérant des patrons.

**Leçon 4 – Penser en segments : La composition et la décomposition.** Pour concevoir un algorithme qui permettra de jouer la mélodie du gobelet, les élèves décomposent les mouvements en plusieurs segments. Dans Tynker, ils résolvent des problèmes en les décomposant en sous-problèmes.

**Leçon 5 – Penser aux ensembles : L'abstraction.** Les élèves découvrent les notions de similarité et de généralisation tandis qu'ils classent des éléments dans des ensembles, puis ils expliquent leur raisonnement. Dans Tynker, ils font appel à l'abstraction pour repérer les similitudes entre les problèmes et résolvent des puzzles de plus en plus complexes grâce à leurs nouveaux outils de programmation.

**Leçon 6 – Penser aux patrons : Les fonctions.** Les élèves créent une série de mouvements pour un robot-assistant et la décomposent en fonctions; ils échangent ensuite leurs algorithmes pour comparer les résultats prévus et réels. Dans Tynker, ils utilisent des noms de fonctions et appellent celles-ci en réutilisant des séries d'instructions pour optimiser leur code.

**Leçon 7 – Penser en détail : Les instructions conditionnelles.** Les élèves définissent les conditions de leur voyage de rêve virtuel en utilisant un ensemble de critères sous forme d'instructions *if*. Dans Tynker, ils utilisent les instructions *if* pour gérer les décisions à prendre et les solutions de rechange dans les puzzles.

**Leçon 8 – Penser en cycles : Les boucles *while* et emboîtées.** Responsables d'une boutique de beignes virtuelle, les élèves utilisent les boucles *while* et emboîtées pour créer des algorithmes et ainsi faire en sorte que le robot-beigne glace des beignes pour chaque client. Dans Tynker, les élèves utilisent les boucles pour raccourcir leur code.

**Leçon 9 – Penser aux changements : Les variables, les entrées et les sorties.** Dans l'activité Jam poétique, les élèves utilisent des variables pour concevoir l'algorithme d'une chanson ou d'un rap, qu'ils interprètent en intégrant des données fournies par la classe (des entrées). Dans Tynker, ils utilisent des variables pour s'attaquer à des puzzles plus complexes, mettant à profit les compétences de programmation qu'ils ont acquises.

**Leçon 10 – Penser en pratique : La conception d'une interface utilisateur.** Les élèves analysent les caractéristiques d'un bon concept, puis créent un pictogramme pour leur école. Dans l'activité de Tynker, ils utilisent toutes leurs nouvelles habiletés et terminent ainsi les leçons de Premiers pas en programmation 2.

# Renseignements supplémentaires

**Ressources pour les cours Premiers pas en programmation**

- Tynker
- codeSpark Academy
- Premiers pas en programmation 1
- Premiers pas en programmation 2

**Ressources pour l'app Swift Playgrounds**

- Learn to Code 1&2 – Cours iTunes U*
- Apprendre à coder 1 et 2 – Guide de l'enseignant
- App Swift Playgrounds

**Guides Introduction au développement d'apps avec Swift**

- Introduction au développement d'apps avec Swift
- Introduction au développement d'apps avec Swift – Guide de l'enseignant

**Ressources complémentaires**

- Apprenez-en plus sur le programme La programmation pour tous.
- Apprenez-en plus sur Swift.
- Apprenez-en plus sur Xcode.
- Prenez part à la conversation sur le forum des développeurs d'Apple* en cliquant sur l'icône « For Educators » .
- Apprenez-en plus sur codeSpark Academy.
- Apprenez-en plus sur Tynker.

* En anglais seulement.

# Conformité avec le curriculum : Premiers pas en programmation 1

Les leçons du cours Premiers pas en programmation 1 sont conformes au concept Algorithmes et programmation des normes intérimaires d'enseignement informatique établies en 2016 par la CSTA pour le niveau 1, de la maternelle à la 2e année. Au fil des leçons et des activités, les élèves explorent les concepts fondamentaux qui sous-tendent la pensée informatique et font leurs premiers pas en programmation.

Seuls ou en équipe, les élèves participent à des activités et utilisent les apps de programmation codeSpark Academy et Tynker.

Conformité du cours Premiers pas en programmation 1 – Normes d'enseignement de la CSTA (de la maternelle à la fin du secondaire) – Niveau 1, de la maternelle à la 2e année

| | Norme de la CSTA | **1A-A-7-1** Souligner le travail des autres | **1A-A-5-2** Créer des programmes | **1A-A-5-3** Élaborer des documents | **1A-A-4-4** Représenter des données | **1A-A-3-5** Décomposer | **1A-A-3-6** Catégoriser des éléments | **1A-A-3-7** Concevoir des algorithmes | **1A-A-6-8** Analyser et déboguer |
|---|---|---|---|---|---|---|---|---|---|
| Leçons de Premiers pas en programmation 1 | **Mettre de l'ordre : Introduction au séquencement** | | ● | ● | ● | ● | | ● | ● |
| | **Procéder par étapes : La création de séquences** | | ● | ● | ● | | | ● | ● |
| | **Faire des choix : Le séquencement modulable** | | ● | ● | ● | ● | | ● | ● |
| | **Répéter à l'infini : Les boucles** | | ● | ● | ● | | | ● | ● |
| | **Corriger les erreurs : Le débogage** | | ● | ● | ● | ● | | ● | ● |
| | **Déclencher le processus : Les événements et les actions** | | ● | ● | ● | ● | | ● | ● |
| | **Respecter les conditions : Les instructions *if*** | | ● | ● | | ● | | ● | ● |
| | **Résoudre les problèmes : Les algorithmes** | | ● | ● | ● | ● | | ● | ● |

Légende : ● Conforme à la norme

# Conformité avec le curriculum : Premiers pas en programmation 2

Les leçons du cours Premiers pas en programmation 2 sont conformes au concept Algorithmes et programmation des normes intérimaires d'enseignement informatique établies en 2016 par la CSTA pour le niveau 1, de la 3e à la 5e année. Au fil des leçons et des activités, les élèves approfondissent leur compréhension des concepts fondamentaux qui sous-tendent la pensée informatique et développent leur compétences en programmation.

Seuls ou en équipe, les élèves participent à des activités et utilisent l'app de programmation Tynker.

Conformité du cours Premiers pas en programmation 2 – Normes d'enseignement de la CSTA (de la maternelle à la fin du secondaire) – Niveau 1, de la 3e à la 5e année

| | Norme de la CSTA | **1B-A-2-1** Adopter des stratégies de collaboration | **1B-A-7-2** Citer les sources | **1B-A-5-3** Planifier | **1B-A-5-4** Créer des programmes | **1B-A-5-5** Utiliser des opérations mathématiques | **1B-A-3-6** Décomposer | **1B-A-3-7** Concevoir des algorithmes | **1B-A-6-8** Analyser et déboguer |
|---|---|---|---|---|---|---|---|---|---|
| Leçons de Premiers pas en programmation 2 | **Penser par étapes : La résolution de problèmes avec des algorithmes** | ● | | ● | ● | | ● | ● | ● |
| | **Penser aux solutions : Le débogage** | ● | | ● | ● | | ● | ● | ● |
| | **Penser en rond : Les boucles** | ● | | ● | ● | | ● | ● | ● |
| | **Penser en segments : La composition et la décomposition** | ● | | ● | ● | | ● | ● | ● |
| | **Penser aux ensembles : L'abstraction** | ● | | | ● | | | ● | ● |
| | **Penser aux patrons : Les fonctions** | ● | ● | ● | ● | | ● | ● | ● |
| | **Penser en détail : Les instructions conditionnelles** | ● | ● | ● | ● | | ● | ● | ● |
| | **Penser en cycles : Les boucles *while* et emboîtées** | ● | | ● | ● | | ● | ● | ● |
| | **Penser aux changements : Les variables, les entrées et les sorties** | ● | | ● | ● | ● | ● | ● | ● |
| | **Penser en pratique : La conception d'une interface utilisateur** | ● | ● | ● | ● | | ● | | ● |

Légende : ● Conforme à la norme

Les fonctionnalités peuvent changer sans préavis. Certaines fonctionnalités peuvent ne pas être offertes dans certaines régions ou certaines langues.

© 2017 Apple Inc. Tous droits réservés. Apple, le logo Apple, iTunes U, Keynote et Xcode sont des marques de commerce d'Apple Inc., déposées aux États-Unis et dans d'autres pays. Swift et Swift Playgrounds sont des marques de commerce d'Apple Inc. Les autres produits et dénominations sociales mentionnés ici peuvent être des marques de commerce de leurs sociétés respectives. Les caractéristiques des produits peuvent changer sans préavis. Le présent document n'est fourni qu'à titre d'information; Apple se dégage de toute responsabilité quant à son utilisation. Mai 2017