GRANDES IDÉES

La **décomposition** aide à résoudre des problèmes difficiles en les simplifiant.	Les **algorithmes** sont essentiels pour résoudre des problèmes au moyen de l'informatique.	La programmation est un outil qui permet de mettre en pratique la **pensée informatique**.	**Résoudre des problèmes** est un processus créatif.

Normes d'apprentissage

Compétences disciplinaires	Contenu
L'élève sera capable de : **Raisonner et modéliser** • Développer une **souplesse de raisonnement** pour analyser et créer des algorithmes • Explorer, **analyser** et appliquer des idées mathématiques et des concepts informatiques au moyen du **raisonnement**, de la **technologie** et d'**autres outils** • **Modéliser** au moyen des mathématiques dans des **situations contextualisées** • Faire preuve de **pensée créatrice** et manifester de la **curiosité et de l'intérêt** dans l'exploration de problèmes **Comprendre et résoudre** • Développer, démontrer et appliquer sa compréhension des concepts par des expériences, l'**investigation** et la résolution de problèmes • Explorer et représenter des concepts et des relations informatiques par la **visualisation** • Appliquer des **approches flexibles et stratégiques** pour **résoudre des problèmes** • Résoudre des problèmes avec **persévérance et bonne volonté** • Réaliser des expériences de résolution de problèmes **qui font référence** aux lieux, aux histoires, aux pratiques culturelles et aux perspectives des peuples autochtones de la région, de la communauté locale et d'autres cultures	*L'élève connaîtra :* • Différentes manières de représenter des **types de données simples** • **Concepts fondamentaux de la programmation** • **Portée** variable • Différentes manières de formuler et d'évaluer des **énoncés logiques** • Utilisation du **flux de commande** pour organiser l'exécution d'un programme • **Élaboration d'algorithmes** pour résoudre des problèmes de plusieurs façons • Différentes techniques pour effectuer des **opérations** et des **recherches** dans des tableaux et des listes • Décomposition de problèmes par la **modularité** • Utilisations de l'informatique en **analyse financière** • Différentes manières de modéliser des **problèmes mathématiques**

Normes d'apprentissage (suite)

Compétences disciplinaires	Contenu
Communiquer et représenter • **Expliquer et justifier** des concepts et des **décisions** mathématiques de **plusieurs façons** • **Représenter** des concepts informatiques sous forme concrète, graphique et symbolique ainsi qu'en pseudocode • Utiliser le vocabulaire et le langage de l'informatique et des mathématiques pour participer à des **discussions** en classe • Prendre des risques en proposant des idées dans le cadre du **discours** en classe **Faire des liens et réfléchir** • **Réfléchir** sur l'approche mathématique et informatique • **Faire des liens entre différents concepts mathématiques et informatiques**, et entre ces concepts et d'autres domaines et intérêts personnels • Voir les **erreurs** comme des **occasions d'apprentissage** • **Incorporer** les visions du monde, les perspectives, les **connaissances** et les **pratiques** des peuples autochtones pour faire des liens avec des concepts informatiques	

Grandes idées – Approfondissements

- **décomposition :**
  - diviser un problème complexe en parties plus faciles à concevoir, à comprendre et à programmer

  Questions pour appuyer la réflexion de l'élève :
  - Comment décompose-t-on un problème en plusieurs parties plus simples?
  - Comment savoir si un problème doit être décomposé encore plus?
  - Existe-t-il un meilleur moyen de décomposer un problème en petites parties et de réutiliser le code?
- **algorithmes :**
  - ensemble de règles ou d'instructions qui définissent précisément une séquence d'opérations

  Questions pour appuyer la réflexion de l'élève :
  - Comment le fait de mettre en situation une solution peut-il aider à développer un algorithme?
  - Comment formule-t-on un algorithme?
  - Qu'est-ce qui fait qu'un algorithme est meilleur qu'un autre?
  - Comment savoir si un algorithme est approprié?
  - Est-ce que tous les problèmes peuvent être résolus par une série d'étapes prédéfinies?
- **pensée informatique :**
  - processus de réflexion qui se fonde sur la reconnaissance des régularités et sur la décomposition dans le but de produire un algorithme exécutable par un ordinateur

  Questions pour appuyer la réflexion de l'élève :
  - Comment choisit-on le langage informatique à employer pour résoudre un problème donné?
  - Pourquoi la lisibilité du code est-elle importante?
  - Quels facteurs influent sur la lisibilité du code?
  - Quelle quantité de documentation du code source est nécessaire?
  - Le problème présente-t-il des régularités qui pourraient être généralisées?
  - Comment reconnaît-on des régularités qui peuvent être traduites en règles?
- **résoudre des problèmes :**

  Questions pour appuyer la réflexion de l'élève :
  - De combien de façons peut-on résoudre tel ou tel problème?
  - Comment procéder pour résoudre un problème de plusieurs façons différentes?
  - Si aucune solution n'est suggérée, comment commencer à résoudre un problème?

Compétences disciplinaires – Approfondissements

- **souplesse de raisonnement :**
  - comprendre que différents algorithmes peuvent résoudre un même problème
- **analyser :**
  - examiner la structure des concepts mathématiques et informatiques et les liens entre eux (p. ex. démontrer la relation entre une probabilité théorique et une probabilité expérimentale au moyen d'une simulation)
- **raisonnement :**
  - raisonnement inductif et déductif
  - prédictions, généralisations et conclusions tirées d'expériences (p. ex. programmation)
- **technologie :**
  - technologie graphique, géométrie dynamique, calculatrices, matériel de manipulation virtuelle, applications conceptuelles
  - usages très variés, notamment :
    - exploration et démonstration de relations mathématiques
    - organisation et présentation de données
    - formulation et mise à l'épreuve de conjectures inductives
    - modélisation mathématique
- **autres outils :**
  - environnements de développement intégrés (EDI)
  - bibliothèques externes
  - utilitaires visuels de comparaison pour visualiser les différences de code (p. ex. Meld)
- **Modéliser :**
  - à l'aide de concepts et d'outils mathématiques, résoudre des problèmes et prendre des décisions (p. ex. dans des scénarios de la vie quotidienne ou abstraits)
  - choisir les concepts et les outils mathématiques nécessaires pour déchiffrer un scénario complexe et essentiellement non mathématique
- **situations contextualisées :**
  - par exemple, des scénarios de la vie quotidienne et des défis ouverts qui établissent des liens entre les mathématiques et la vie quotidienne
- **pensée créatrice :**
  - être ouvert à l'essai de stratégies différentes
  - on fait référence ici à une réflexion mathématique créatrice et innovatrice plutôt qu'à une représentation créative des mathématiques, p. ex. par les arts ou la musique
- **curiosité et de l'intérêt :**
  - poser des questions pour approfondir sa compréhension ou pour ouvrir de nouvelles voies d'investigation
- **investigation :**
  - investigation structurée, orientée et libre
  - observer et s'interroger
  - relever les éléments nécessaires pour comprendre un problème et le résoudre

Compétences disciplinaires – Approfondissements

- **visualisation :**
  - visualiser graphiquement des structures de données
  - utiliser des organigrammes
  - se servir d'utilitaires ou de sites Web de visualisation de code (p. ex. http://pythontutor.com/)
- **approches flexibles et stratégiques :**
  - utiliser différents algorithmes pour résoudre un même problème
  - concevoir des algorithmes capables de résoudre une catégorie de problèmes plutôt qu'un seul problème
  - choisir les régularités de programmation appropriées pour résoudre un problème
  - choisir une stratégie efficace pour résoudre un problème (p. ex. essai-erreur, modélisation, résolution d'un problème plus simple, utilisation d'un graphique ou d'un diagramme, jeu de rôle)
- **résoudre des problèmes :**
  - interpréter une situation pour cerner un problème
  - appliquer les mathématiques à la résolution de problème
  - analyser et évaluer la solution par rapport au contexte initial
  - répéter ce cycle jusqu'à ce qu'une solution vraisemblable ait été trouvée
- **persévérance et bonne volonté :**
  - ne pas abandonner devant les difficultés
  - résoudre les problèmes avec dynamisme et détermination
- **qui font référence :**
  - aux activités quotidiennes, aux pratiques locales et traditionnelles, aux médias populaires, aux événements d'actualité et à l'intégration interdisciplinaire
  - en posant et en résolvant des problèmes, ou en posant des questions sur les lieux, les histoires et les pratiques culturelles
  - à la cryptographie (p. ex. les « Code Talkers » Navajos de la Deuxième Guerre mondiale)
- **Expliquer et justifier :**
  - utiliser des arguments mathématiques pour convaincre
  - prévoir des conséquences
- **décisions :**
  - demander aux élèves de choisir parmi deux scénarios, puis de justifier leur choix
- **plusieurs façons :**
  - orale, écrite, graphique; utilisation de la technologie
  - communiquer efficacement d'une manière adaptée à la nature du message et de l'auditoire
- **Représenter :**
  - à l'aide de modèles, de tables, d'organigrammes, de mots, de nombres, de symboles
  - en établissant des liens de sens entre plusieurs représentations différentes
  - au moyen de matériels concrets et d'une technologie interactive dynamique

Compétences disciplinaires – Approfondissements

- **discussions :**
  - dialogues entre pairs, discussions en petits groupes, rencontres enseignants-élèves
- **discours :**
  - utile pour approfondir la compréhension des concepts
  - peut aider les élèves à clarifier leur réflexion, même s'ils doutent quelque peu de leurs idées, ou si leurs prémisses sont erronées
- **Réfléchir :**
  - présenter le résultat de son raisonnement mathématique et informatique et celui d'autres personnes, y compris évaluer les stratégies et les solutions, développer les idées et formuler de nouveaux problèmes et de nouvelles questions
- **Faire des liens entre différents concepts mathématiques et informatiques :**
  - s'ouvrir au fait que l'informatique peut aider à se connaître et à comprendre le monde autour de soi (p. ex. activités quotidiennes, pratiques locales et traditionnelles, médias populaires, événements d'actualité, justice sociale et intégration interdisciplinaire)
- **erreurs :**
  - de syntaxe, de sémantique, d'exécution et de logique
- **occasions d'apprentissage :**
  - en :
    - analysant ses erreurs pour cerner les éléments mal compris
    - apportant des correctifs à la tentative suivante (p. ex. débogage)
    - relevant non seulement les erreurs mais aussi les parties d'une solution qui sont correctes
- **Incorporer :**
  - en :
    - collaborant avec les Aînés et les détenteurs du savoir parmi les peuples autochtones de la région
    - explorant les principes d'apprentissage des peuples autochtones (http://www.fnesc.ca/wp/wp-content/uploads/2015/09/PUB-LFP-POSTER-Principles-of-Learning-First-Peoples-poster-11x17.pdf : l'apprentissage est holistique, introspectif, réflexif, expérientiel et relationnel [axé sur la connexité, les relations réciproques et l'appartenance]; l'apprentissage demande temps et patience)
    - faisant des liens explicites avec l'apprentissage des mathématiques
    - explorant les pratiques culturelles et les connaissances des peuples autochtones de la région, et en faisant des liens avec les mathématiques
- **connaissances :**
  - connaissances locales et pratiques culturelles qu'il est convenable de partager et qui ne relèvent pas d'une appropriation
- **pratiques :**
  - pratiques culturelles selon Bishop : compter, mesurer, localiser, concevoir, jouer, expliquer (http://www.csus.edu/indiv/o/oreyd/ACP.htm_files/abishop.htm)
  - ressources sur l'éducation autochtone (www.aboriginaleducation.ca)
  - *Teaching Mathematics in a First Nations Context*, FNESC (http://www.fnesc.ca/resources/math-first-peoples/)

Contenu – Approfondissements

- **types de données simples :**
  - systèmes numériques (p. ex. binaire, hexadécimal)
  - chaînes, entiers relatifs, caractères, virgule flottante
- **Concepts fondamentaux de la programmation :**
  - variables, constantes, opérations mathématiques, entrée/sortie, génération de nombres aléatoires
- **Portée :**
  - locale ou globale
- **énoncés logiques :**
  - opérateurs logiques (ET, OU, NON)
  - opérateurs relationnels (<, >, <=, >=, ==, !=, <>)
  - équivalences logiques (p. ex. lois de De Morgan), simplification d'énoncés logiques, tables de vérité
- **flux de commande :**
  - structures de décision (p. ex. si-alors-sinon [if-then-else])
  - boucles (p. ex. pour [for], tant que [while] et boucle emboîtée)
- **Élaboration d'algorithmes :**
  - affinement progressif, pseudocode ou organigrammes, traduire du pseudocode en code (et vice-versa)
- **opérations :**
  - ajouter, supprimer, insérer, effacer
- **recherches :**
  - algorithmes de recherche (p. ex. recherches linéaires et binaires)
- **modularité :**
  - utilisation de méthodes et de fonctions pour simplifier, recyclage de code et utilisation de paramètres de fonctions
  - valeurs de retour
- **analyse financière :**
  - valeur temporelle de l'argent, appréciation/dépréciation, amortissement hypothécaire
  - modifier les variables d'un scénario financier pour faire une analyse par simulation (p. ex. comparer différents versements mensuels, différentes durées et différents taux d'intérêt)
- **problèmes mathématiques :**
  - estimer une probabilité théorique par la simulation
  - représenter des suites et des séries arithmétiques finies
  - résoudre un système d'équations linéaires, croissance/décroissance exponentielle
  - résoudre une équation polynomiale
  - calculer les valeurs statistiques de grands ensembles de données, comme la fréquence, la tendance centrale et l'écart-type
  - calculer le plus grand commun diviseur et le plus petit commun multiple