DOI : https://doi.org/10.26034/cm.iptic.2025.11755

INTÉGRATION DE LA PENSÉE INFORMATIQUE DANS LA FORMATION DES PROFESSIONNELS DE L'ÉDUCATION : ANALYSE CROISÉE DE PERFORMANCE ET DU DEGRÉ DE CERTITUDE

Sabrin Housni, Université de Mons, Belgique,
sabrin.housni@umons.ac.be

Gaëtan Temperman, Université de Mons, Belgique,
gaetan.temperman@umons.ac.be

Audrey Kumps, Université de Mons, Belgique,
audrey.kumps@umons.ac.be

Karim Boumazguida, Université de Mons, Belgique,
karim.boumazguida@umons.ac.be

Laetitia Dragone, Université de Mons, Belgique,
laetitia.dragone@umons.ac.be

Sarah Descamps, Université de Mons, Belgique,
sarah.descamps@umons.ac.be

Bruno De Lièvre, Université de Mons, Belgique,
bruno.delievre@umons.ac.be

Résumé

Notre étude de l'effet de l'intégration de la pensée informatique (PI) dans l'enseignement supérieur a été menée auprès de 122 étudiants en Master Sciences de l'Éducation, spécialisés en Technologie de l'Éducation. Le dispositif comprend six séances de deux heures mêlant théorie, pratique et réflexion conduisant à une évaluation de la PI des étudiants par le « Computational Thinking Test » (CTT) et du sentiment de maitrise des étudiants objectivé par le degré de certitude. Les résultats des étudiants montrent une progression significative des deux variables. L'analyse révèle également une corrélation entre les scores au CTT et le sentiment de maîtrise.

Mots-clés : Pensée informatique, Évaluation, Performance, Degré de certitude, Computational Thinking Test.

1. Introduction

La pensée informatique (PI) est une compétence jugée importante pour s'adapter à l'avenir (Hsu et al., 2018) et pour mieux comprendre et interagir avec les algorithmes qui influencent nos vies (Romero et Vallerand, 2016 ; Dillenbourg, 2018). Elle est indispensable partout où il y a transformation de l'information et innovation des systèmes d'information (Ben Henda, 2016). Elle semble également nécessaire pour comprendre le monde complexe qui nous entoure (Dillenbourg, 2018) et pour répondre au besoin sociétal croissant dans le domaine des métiers digitaux (Amarelle, 2018).

Dans cette perspective, l'école joue un rôle clé dans cet apprentissage. Comme le discutent Bugmann et Karsenti (2018), en intégrant les compétences numériques, elle favorise le développement de la PI dès les premiers stades de l'éducation. Il est essentiel que les enseignants maîtrisent les savoirs associés et développent ces compétences pour pouvoir les enseigner efficacement (Tchounikine, 2017).

Ainsi, en Belgique francophone, l'intégration du nouveau référentiel « Formation Manuelle, Technique, Technologique et Numérique » (FMTTN) dans le cadre de la mise en œuvre d'un nouveau tronc commun renforce cette idée. Ce référentiel indique que la pensée informatique et algorithmique (PIA) doit être développée dès l'enseignement primaire (Fédération Wallonie-Bruxelles, 2022).

Dans ce contexte, notre étude s'intéresse au développement de la PI chez les étudiants du Master en Sciences de l'Éducation, spécialisés en Technologie de l'Éducation à l'Université de Mons en Belgique francophone. Nous cherchons à évaluer dans quelle mesure le cours intitulé « Développement pédagogique de la pensée informatique », basé sur le postulat de l'isomorphisme pédagogique (Develay, 1994), contribue à son développement. Cette recherche vise à proposer des perspectives pour l'intégration de la PI dans les programmes académiques en sciences de l'éducation, afin de renforcer la formation des futurs enseignants et de soutenir l'effort global d'intégration de ces compétences dans le système éducatif.

Cet article se structure comme suit : dans un premier temps, une définition de la PI est présentée, suivie d'une revue des recherches empiriques consacrées à son développement et à son évaluation. Ensuite, nous détaillons la méthodologie de notre recherche, suivie de l'analyse des résultats obtenus. Enfin, l'article se conclut par une discussion sur les implications pour la conception de dispositifs éducatifs visant à développer la PI dans les programmes d'enseignement supérieur.

2. Ancrage théorique

2.1 Définition de la pensée informatique

Le concept de PI, également appelée Computational Thinking, fut utilisé pour la première fois par Papert (1981). Bien qu'il n'existe pas de définition unanimement acceptée de ce concept dans la littérature (Drot-Delange et al., 2019 ; Lockwood et Mooney, 2017 ; Moreno-León et al., 2018), notamment en raison de son évolution en fonction des avancées dans le domaine (Shute et al., 2017), de nombreux auteurs s'appuient sur les définitions proposées par Wing (2006) comme point de départ de leurs réflexions. En effet, Wing (2006) a popularisé le concept de PI en préconisant son apprentissage pour les nouveaux étudiants universitaires. L'auteure définit la PI comme l'habileté à résoudre des problèmes de manière algorithmique. Elle précise que la PI devrait être considérée comme une compétence de la vie quotidienne dont tout le monde a besoin, plutôt qu'une simple expertise en programmation utile uniquement aux informaticiens (Hsu et al., 2018). La PI a donc d'abord été définie comme un processus humain de résolution de problèmes qui utilise la décomposition et nécessite une réflexion à plusieurs niveaux d'abstraction (Wing, 2006), soit les « processus de réflexion impliqués dans la formulation des problèmes et de leurs solutions afin que les solutions soient représentées sous une forme qui puisse être efficacement mise en œuvre par un agent de traitement de l'information » (Wing, 2010, p.20). Aho (2012) propose une perspective plus réductrice en décrivant la PI comme un processus mental qui consiste à formuler des problèmes de manière à pouvoir les résoudre étape par étape à travers des algorithmes.La PI se définit également de manière plus globale pour favoriser le développement de solutions numériques plus universelles comme la capacité de « savoir décomposer un problème en sous-problèmes plus simples ; savoir réfléchir aux tâches à accomplir pour résoudre un problème en termes d'étapes et d'actions (ce que l'on appelle un algorithme) ; savoir décrire les problèmes et les solutions à différents niveaux d'abstraction, ce qui permet d'identifier des similitudes entre problèmes et, par suite, de pouvoir réutiliser des éléments de solutions ; ainsi que, dans l'approche que [l'auteur] adopte, savoir écrire et tester ces algorithmes avec [un] langage de programmation [tel que] Scratch » (Tchounikine, 2017, para. 4).

D'un point de vue pragmatique, la PI est en mesure dès lors de rationaliser notre pensée en structurant les séquences d'instructions permettant de résoudre un problème de manière automatique ou d'accomplir une tâche de manière systématique (algorithme).

Enfin, Roman-Gonzalez (2015) indique que la PI implique la capacité à formuler et à résoudre des problèmes en s'appuyant sur les concepts fondamentaux de l'informatique et en utilisant les lexiques et la syntaxe des langages de programmation dont les éléments fondamentaux de l'affectation des variables, les séquence de base de lecture et d'écriture, les alternatives (conditions) et itérations (boucles).

2.2 Enseigner la pensée informatique

La littérature souligne un manque de clarté quant aux méthodes d'enseignement de la PI (Hsu et al., 2018). L'apprentissage de la programmation informatique est un processus itératif alliant activités de programmation, analyse et résolution d'un problème (Zapata-Cáceres et al., 2020). Ainsi, cela amène à résoudre des problèmes en appliquant une logique structurée en vue de construire des solutions : développer une PI va donc au-delà du simple apprentissage de la programmation (Jimenez et al., 2024).

Dans cette optique, une approche centrée sur l'algorithmique et la programmation apparaît pertinente pour développer la PI. Elle favorise le travail sur des réifications tout en offrant un support concret – qu'il s'agisse d'une activité, d'un écran ou d'un objet de programmation tangible – qui soutient la réflexion et l'action (Tchounikine, 2017). Elle implique de mettre l'algorithme et les structures de base sous-jacentes (notions de variables, de séquence de base de lecture et d'écriture, les alternatives et itérations) au cœur de l'enseignement pour permettre aux apprenants de savoir créer un algorithme avec ou sans ordinateur et utiliser un langage de programmation pour l'expérimenter (Tchounikine, 2017). Cette approche est également privilégiée en Belgique francophone, comme le souligne le nouveau référentiel FMTTN : « En abordant des notions de logigramme et de programmation, au travers d'exercices simples de lecture et d'écriture, l'élève développe sa pensée informatique et algorithmique » (REF FMTTN, p.25).

L'apprentissage de l'algorithmique et de la programmation peut entrainer plusieurs difficultés pour les apprenants (Fessard et al., 2019). Pour surmonter ces défis, différents choix pédagogiques sont possibles. Classiquement, la littérature oppose les activités débranchées et les activités branchées de programmation (Frison et al., 2018).

Les activités débranchées consistent notamment à développer des algorithmes sans utiliser d'ordinateurs, permettant aux apprenants de comprendre les concepts fondamentaux liés à cet apprentissage (Drot-Delange, 2013). Celles-ci constituent une première étape avant de passer à la programmation effective (Frison et al., 2019). Les activités branchées consistent, quant à elles, à mettre en œuvre ces algorithmes, soit de concevoir les programmes opérationnels en s'appuyant sur un environnement informatique ou un outil informatique (Romero et al., 2018), tels que des logiciels de programmation ou des objets tangibles programmables (Sigayret et al., 2021).

La littérature consultée ne converge pas nécessairement sur l'efficacité d'un dispositif par rapport à un autre (Romero et al., 2018 ; Fessard et al., 2018 ; Sigayret et al., 2021). Au contraire, ceux-ci ne sont pas nécessairement en opposition et peuvent cohabiter (Tchounikine, 2017). Ils semblent donc complémentaires selon le moment de l'apprentissage ou les objectifs pédagogiques visés. Les activités débranchées réduisent la charge cognitive liée à l'utilisation d'une machine, permettant aux apprenants une meilleure attention et une meilleure interaction avec le groupe (Romero et al., 2018). Néanmoins, les logiciels de programmation offrent une expérience interactive lors de laquelle les apprenants peuvent expérimenter différentes approches de résolution et recevoir un feedback immédiat, favorisant ainsi l'apprentissage par l'essai-erreur (Sigayret et al., 2021). De même, les robots pédagogiques tangibles permettent de combiner la programmation avec une composante physique, ce qui peut renforcer la compréhension pratique des concepts abstraits de programmation (Sullivan et Heffernan, 2016).

Dillenbourg (2018) souligne qu'il existe désormais une variété d'environnements numériques d'apprentissage : des plateformes logicielles et des robots éducatifs pour l'apprentissage de la programmation et de la PI. Cette diversité de dispositifs permet aux enseignants d'adopter des approches variées en fonction des besoins spécifiques des apprenants et des objectifs pédagogiques qu'ils visent.

Ainsi, c'est l'articulation de diverses méthodes qui semble enrichir l'apprentissage de la PI en offrant aux apprenants des opportunités variées de développer et d'appliquer leurs compétences en programmation et en résolution de problèmes. Pour objectiver ces apprentissages, il est nécessaire d'évaluer systématiquement le développement de ces compétences.

2.3 Évaluer la pensée informatique

Bien que la manière de mesurer le niveau de développement de la PI ne fasse pas consensus dans la littérature scientifique (Román-González et al., 2017), celle-ci fait l'objet de nombreuses initiatives que nous nous proposons de catégoriser au regard du modèle des 4P : performances, processus, perceptions et profils (Temperman, 2013).

Les performances des apprenants peuvent être évaluées à partir de questionnaires administrés en pré et post-test portant sur l'évaluation de la PI (Brackmann et al., 2017 ; Romero et al., 2018 ; Marfisi-Schottman et al., 2020). Les processus mis en œuvre dans l'apprentissage peuvent être analysés via un modèle d'observation des étapes de résolution de problèmes (Chevalier et al., 2020) ou encore sur base d'une grille d'observation qui permet d'évaluer la PI à travers la réalisation d'activités par les apprenants (Bellegarde & Boyaval, 2020 ; Chaker et Njingang Mbadjoin, 2020). Le lien entre les performances et les processus peut être questionné via une analyse croisée en considérant la progression des apprenants (via des questionnaires pré et post expérimentations) et les comportements en cours d'apprentissage comme les traces d'activité de programmation (Grover et al., 2014 ; Peter et al., 2019 ; Léonard et al., 2021). Enfin, La perception des compétences des apprenants est également évaluée, permettant de croiser leur ressenti sur leurs capacités avec leur progression effective.

L'éventail de ces approches montre qu'il est important de réfléchir à la complémentarité entre les différentes stratégies d'évaluation de la PI, tout en tenant compte des contraintes éventuelles de l'environnement d'apprentissage et du contexte dans lequel sont récoltées les données.

3. Méthodologie

La méthodologie de cette étude se compose de trois sections principales : le contexte de la recherche et la description de l'échantillon, la description du dispositif pédagogique mis en place, et enfin les questions de recherche accompagnées de la présentation de l'instrument utilisé.

3.1 Contexte de la recherche et description de l'échantillon

Les 122 participants à cette étude sont des étudiants inscrits en début de cycle du Master en sciences de l'éducation, avec une spécialisation en technologie de l'éducation, à l'Université de Mons (90 étudiantes et 32 étudiants). Le cours intitulé « Développement pédagogique de la pensée informatique », introduit durant l'année académique 2021-2022, vise deux objectifs complémentaires : d'une part, faire découvrir aux étudiants les concepts de la PI (tels que les séquences, les itérations, les alternatives, etc.), et d'autre part, leur permettre d'identifier, d'utiliser et d'évaluer différents outils pédagogiques pour développer cette pensée. L'objectif est donc de familiariser les étudiants avec diverses activités pédagogiques, leur permettant de mobiliser les concepts et outils pour les engager dans une réflexion pédagogique et critique sur l'intégration de ces concepts et outils dans leur propre pratique éducative.

A l'instar des travaux de Tchounikine (2017), il est important de souligner que former des professionnels de l'éducation à l'enseignement de la PI n'implique pas nécessairement de les former à l'informatique. Il s'agit plutôt de leur fournir des éléments permettant de comprendre ce qu'est l'enseignement de cette forme de pensée et comment l'aborder pédagogiquement. Néanmoins, tout enseignement nécessite une maîtrise de l'objet d'enseignement (Tchounikine, 2017), ce qui renforce la pertinence du cours proposé et justifie son intégration dans le cursus des futurs enseignants et formateurs d'enseignants.

3.2 Présentation du dispositif technopédagogique

Dans ce dispositif de formation, nous nous appuyons sur le postulat de l'isomorphisme pédagogique tel que défini par Develay (1994, p.23) : « On utilise souvent le terme d'isomorphisme ou d'homomorphisme pour désigner le postulat selon lequel c'est en faisant vivre et analyser aux formés des situations proches, au niveau des attitudes, des méthodes voire des contenus, à celles que ceux-ci auront à faire vivre à leurs élèves que le formateur aide durablement ses formés à intégrer l'ensemble des procédures cognitives et affectives mises en jeu ». Cela soutient l'idée qu'il s'agit « d'appliquer aux formés les méthodes et techniques que l'on souhaite leur voir appliquer auprès des élèves » (Develay, 1994, p.23).

Pour mettre en pratique ce postulat et atteindre les deux objectifs visés – la familiarisation et l'engagement critique – le cours s'articule autour de séances synchrones et asynchrones. Chaque session offre, dans un premier temps, l'occasion de découvrir de manière active différents outils branchés et débranchés, puis, dans un second temps, de prendre du recul sur les démarches cognitives mises en œuvre.

Le dispositif pédagogique, réparti sur trois mois, inclut six sessions synchrones de deux heures chacune, dont les contenus sont détaillés dans le Tableau 1. Ces sessions sont complétées par des tâches asynchrones d'une durée équivalente. Le dispositif combine des apports théoriques, des ateliers pratiques et des échanges réflexifs. Pour les ateliers, des sous-groupes de 3 à 4 étudiants sont constitués afin de faciliter la manipulation directe des outils. La présence de 2 à 3 encadrants pédagogiques est nécessaire pour accompagner l'ensemble du groupe.

Ce dispositif peut être qualifié d'hybride (Charlier et al., 2006), car il alterne trois séances à distance via Teams, une application de communication collaborative et trois séances en présentiel, permettant ainsi l'utilisation de matériel tangible. Chaque séance synchrone est suivie de tâches d'application et de découverte des logiciels, à réaliser de manière asynchrone et autonome. Pour guider leur réflexion pédagogique, les étudiants disposent d'un syllabus détaillé leur permettant d'approfondir les concepts abordés.

Tableau 1. contenus et modalités des 6 séances du dispositif pédagogique

Séance(s)	Modalités	Contenu
1	Distanciel	• Présentation des concepts théoriques fondamentaux de la PI ; • Discussion sur les origines et le développement historique du concept ; • Analyse réflexive du contexte actuel et des initiatives en Belgique francophone ; • Découverte du logiciel Algoblocs (Piva, C. ; Villeubanne, France).
2	Distanciel	• Introduction aux environnements de programmation par blocs. Par exemple : Scratch (Resnick, M. ; Cambridge, Etat-Unis), Minecraft Education (Mojang Studios, avec le soutien de Microsoft ; Washington, Etat-Unis) ; • Découverte des principes informatiques et pédagogiques sous-jacents ; • Analyse réflexive autour des études empiriques en sciences de l'éducation associées à la programmation par blocs.
3	Distanciel	• Introduction aux principes de robotiques pédagogiques avec l'environnement MakeCode (Microsoft ; Washington, États-Unis) et la carte Micro-bit (British Broadcasting Corporation ; Londres, Royaume-Uni) ; • Découverte des principes informatiques et pédagogiques sous-jacents ;
4	Présentiel	• Présentation de Lego Spike (LEGO Education ; Billund, Danemark) • et de leur fonctionnement ; • Exploration des principes informatiques appliqués à la robotique ; • Analyse des principes informatiques, pédagogiques et des études empiriques en robotique éducative.
5	Présentiel	• Introduction aux activités débranchées et à leur importance ; • Stratégies pour passer des activités débranchées aux activités branchées ; • Analyse des principes informatiques, pédagogiques et des études empiriques sous-jacents.
6	Présentiel	• Introduction aux outils d'évaluation de la PI ; • Méthodologies d'évaluation basées sur des données de performance, de processus et de perception.

3.3 Collecte des données

Pour évaluer le cours, nous nous appuyons sur les travaux réalisés par les étudiants dans le cadre des tâches asynchrones intermédiaires, ainsi qu'un examen final. Cette évaluation vise à vérifier l'atteinte des objectifs du cours, en particulier la compréhension des concepts associés à la PI, leur intégration dans les pratiques pédagogiques, ainsi que la capacité des étudiants à aborder ces concepts avec des apprenants.

Avant la première séance et après la dernière séance, nous administrons un test de mesure de la PI via un Google Forms. L'adaptation française du *Computational Thinking Test* ou *CTT*, validé par Román-González et al. (2017), réalisée dans le cadre de l'étude CoCreaTIC (Romero et al., 2018) est utilisée. Notre choix s'est porté sur cet outil, car il a fait l'objet de validation (Zapata-Caceres et al., 2020 ; El-Hamamsy et al., 2022 ;

Lockwood et Mooney, 2018) et d'adaptations linguistiques en le traduisant (Çetin et al., 2020 ; Zhang et Wong, 2023 ; Romero et al., 2018). De plus, il permet l'administration rapide d'un pré-test et d'un post-test, offrant ainsi une évaluation de l'évolution des apprenants à deux moments distincts à partir d'un calcul de gains relatifs. Selon D'Hainaut (1975), le gain relatif permet d'évaluer l'efficacité d'une action pédagogique où chacun des participants possède un niveau initial qui lui est propre et dont la mesure s'obtient en calculant le rapport de ce que l'apprenant a gagné à ce qu'il aurait pu gagner au maximum. Selon ce même auteur, en sciences humaines, il y a un apprentissage effectif, et donc un effet positif de l'apprentissage, lorsque le seuil des 30 % est franchi.

D'un point de vue édumétrique, le CTT se compose de 28 items répartis de manière égale entre les 7 concepts du questionnaire (4 items par concept) : directions et séquences de base, boucle « for », boucle « while », condition « if » simple, condition « if/else » complexe, condition « while », fonction tels que définis par Román-González (2015). Ainsi, « Tous les items qui composent le test impliquent, dans une mesure plus ou moins grande, les processus cognitifs des quatre piliers de la PI : décomposition, reconnaissance des formes, abstraction et algorithmes » (Brackmann et al., 2017, p.66).

En outre, un item, évaluant le sentiment de maîtrise des étudiants, les invite à évaluer leur propre niveau de réussite au test : « Sur une échelle de 0 à 10, estimez votre niveau de réussite à ce test ». On peut associer ce sentiment de maîtrise au principe de degré de certitude qui constitue un marqueur de l'autorégulation (Focant, 2007). Cette perception évalue la capacité d'un individu à juger de manière précise la justesse de ses propres réponses (Focant, 2007 ; Dumont, 2021). En proposant cet item, on obtient une valeur numérique qui indique la probabilité qu'une réponse soit considérée comme correcte par le répondant (Leclercq et Plunus, 1996, cités par Dumont, 2021, p. 84).

Un individu est considéré comme un bon autorégulateur s'il manifeste une grande confiance dans ses réponses lorsqu'elles sont correctes et reconnaît leurs erreurs lorsque celles-ci sont avérées (Focant, 2007). Pour éviter les ambiguïtés dans l'évaluation de la certitude, Leclercq (1986, cité par Dumont, 2021, p. 88) préconise de ne pas utiliser de termes imprécis tels que « peu sûr », « moyennement sûr » ou « très sûr », car ces expressions peuvent être interprétées de manière variable selon les individus. Dans le domaine des sciences de l'éducation, cet indicateur est souvent employé pour garantir la validité des interprétations des résultats et sert également de facteur discriminant dans les évaluations objectives (Bernier, 1971 ; Magis et al. 2011).

3.4 Questions de recherche et traitement des données

A partir de ces différentes mesures, nous pouvons formuler les trois questions de recherche suivantes :

> **Question n°1** : *Le dispositif pédagogique permet-il une progression de la performance au Computational Thinking Test ?*
>
> **Question n°2** : *Le dispositif pédagogique modifie-t-il le degré de certitude des étudiants au Computational Thinking Test ?*
>
> **Question n°3** : *Quels sont les liens entre les scores obtenus aux Computational Thinking Tests et le degré de certitude ?*

Sur le plan statistique, l'ensemble des données a été traité avec le logiciel SPSS version 29.

Les questions 1 et 2 sont traitées à travers des analyses descriptives et inférentielles, en tenant compte de la normalité des données. Ces analyses visent à évaluer la progression des scores au CTT (Q1) ainsi que l'évolution de la perception du degré de certitude (Q2).

Pour approfondir notre analyse, nous avons réalisé une analyse de régression linéaire multiple (Q3) en intégrant, comme variables d'entrées indépendantes, les différents scores obtenus pour les 7 concepts du CTT et comme variable de sortie à modéliser/estimer le degré de certitude de la réussite au CTT. L'ambition est d'identifier parmi les scores aux différents concepts au CTT, ceux qui peuvent caractériser le degré de certitude.

Au niveau statistique, nous avons choisi d'appliquer la méthode de sélection par retrait progressif (méthode descendante) afin que le logiciel d'analyse statistique mette en évidence le modèle qui offre le degré de prédiction le plus élevé, en éliminant successivement les variables les moins pertinentes et non

significatives (par ex. sélections parmi les modèles candidats). Pour utiliser de manière appropriée cette méthode, il faut à la fois veiller à obtenir un nombre restreint de prédicteurs pertinents et significatifs tout en s'intéressant au degré de prédiction fourni par la valeur du R2 ajusté.

Enfin, nous avons considéré l'indice de tolérance qui nous donne la possibilité d'identifier un éventuel problème de colinéarité dans le modèle proposé. En nous référant à Stafford & Bodson (2006), il est généralement admis qu'une valeur de tolérance inférieure à .20 risque d'entraîner des problèmes d'estimation des coefficients associés aux variables affectées de colinéarité. De manière diachronique, nous avons réalisé cette analyse d'abord pour les données liées au pré-test, ensuite pour celles liées au post-test.

4. Résultats et discussion

Notre analyse des résultats s'articule autour de nos 3 questions de recherche.

4.1 Réponse à la question 1 : Le dispositif pédagogique permet-il une progression de la performance au Computational Thinking Test ?

Le tableau 2 montre les résultats descriptifs des 122 participants, mesurés avant (pré-test) et après (post-test) notre dispositif pédagogique. Il inclut également la moyenne des gains relatifs pour chaque concept évalué par le *CTT* (Román-González, 2015).

Nos résultats suggèrent que les apprenants ont montré des tendances positives dans l'amélioration des performances évaluées entre le pré-test et le post-test. L'analyse du tableau 3 indique une progression générale des apprenants, soulignant l'efficacité de l'intervention pédagogique pour renforcer les compétences en programmation. Quel que soit le concept considéré, le niveau au post-test est systématiquement supérieur au niveau initial. Les apprenants semblent avoir le plus progressé pour le concept conditions « if » simples avec une moyenne allant de 2,81 (70,30%) au pré-test à 3,11 (77,80%) au post-test. La moyenne des gains relatifs pour ce concept s'élève à 27,60%. Nous notons également une progression notable dans la maîtrise des boucles « for » dont le score moyen passe de 3,46 (86,50%) au pré-test à 3,70 (92,50%) au post-test, avec une moyenne des gains relatifs de 26,90%, indiquant que le dispositif pédagogique a bien renforcé ce concept. Par ailleurs, le score moyen des structures conditionnelles « if/else » complexes augmente de 2,80 (70,00%) à 3,17 (79,30%), soit une moyenne des gains relatifs de 24,30%. À l'inverse, c'est au niveau du concept conditions « while » que les apprenants semblent avoir le moins progressé avec une moyenne des gains relatifs s'élevant à 14,10%. Cette faible progression peut s'expliquer par la maitrise difficile d'une structure de contrôle qui fait appel à la logique booléenne et qui permet d'exécuter une série d'instructions de manière répétée tant qu'une condition est vraie. L'articulation de ces compétences complémentaires demande probablement un enseignement plus spécifique.

Enfin, nous observons de manière globale que la moyenne des gains relatifs, exprimée en pourcentage, s'élève à 34,50%, passant de 76,70% au pré-test à 83,90% au post-test. Selon D'Hainaut (1975), cette progression indique un effet positif de l'apprentissage. Le niveau de compétence élevé observé chez les participants dès le pré-test suggère une maîtrise préalable des concepts évalués par le CTT, probablement acquise au cours de leur parcours antérieur. Nous ajoutons que le taux d'hétérogénéité diminue après notre dispositif pédagogique, passant de 20,41% au pré-test à 16,07% au post-test. Ceci indique qu'après le dispositif pédagogique, le niveau d'hétérogénéité est plutôt faible, se rapprochant du seuil de 15% (D'Hainaut, 1975 ; Ouellet, 1985 cité par Gerard, 2003), indiquant une disparité plus faible entre les aptitudes des sujets relatives aux concepts mesurés par le CTT à l'issue de l'apprentissage.

Tableau 2. analyses descriptives et inférentielles : Progressions au CTT entre le pré-test et le pst-test (N = 122)

	Pré-test		Post-test		Moyenne des gains relatifs (%)	p-value
	χ (%)	σ	χ (%)	σ		
Directions et séquences de base	3,68 (92,00%)	0,59	3,81 (95,00%)	0,43	14,50	**0,014**
Boucles « for »	3,46 (86,50%)	0,67	3,70 (92,50%)	0,54	26,90	**<0,001**
Boucles « while »	3,30 (82,50%)	0,75	3,47 (86,80%)	0,66	22,00	**0,043**
Conditions « if » simples	2,81 (70,30%)	0,92	3,11 (77,80%)	0,99	27,60	**0,002**
Conditions « if/else » complexes	2,80 (70,00%)	1,23	3,17 (79,30%)	1,04	24,30	**0,002**
Conditions « while »	2,52 (63,00%)	1,21	2,78 (69,50%)	1,13	14,10	**0,03**
Fonctions	3,09 (77,30%)	1,04	3,45 (86,3%)	0,93	23,60	**<0,001**
Score total	21,48 (76,70%)	4,38	23,49 (83,90%)	3,78	**34,50**	**<0,001**
Taux d'hétérogénéité	20,41%		16,07%			

Les hypothèses nulles associées à cette première question de recherche sont similaires pour chaque concept évalué par le CTT, ainsi que pour le score global : « *il n'y a pas de différences significatives entre les scores au pré-test et les scores au post-test* ». En raison de la non-normalité des données[1], pour chacun des 7 concepts mesurés par le CTT ainsi qu'au score global, nous avons privilégié un test non paramétrique afin de vérifier l'H0.

Les résultats du test de Wilcoxon (Tableau 2) révèle des différences significatives[2] pour chaque concept considéré dans le CTT : séquences de base ($p = 0,014$), boucles « for » ($p < 0,001$), boucles « while » ($p = 0,043$), conditions « if » simples ($p = 0,002$), conditions « if/else » complexes ($p = 0,002$), conditionnelles ($p = 0,03$), fonctions ($p < 0,001$). Quel que soit le concept, les étudiants ont significativement progressé entre le pré-test et le post-test. Les étudiants ont globalement un score au CTT significativement plus élevé après avoir participé au dispositif technopédagogique ($p < 0,001$).

4.1 Réponse à la question 2 : Le dispositif pédagogique modifie-t-il le degré de certitude des étudiants au Computational Thinking Test ?

Le degré de certitude au CTT a augmenté entre le pré-test et le post-test. Les apprenants (N = 122) passent d'un score moyen de 6,61/10 (soit 66,10% ; -) à 7,70/10 (soit 77,00% ; -). Cette perception est par ailleurs plus homogène au terme de l'apprentissage (coefficient de variation [CV] = 26,17% au prétest ; contre CV = 17,66 % au post-test). Ce résultat tend à montrer une cohérence entre la progression effective et la progression perçue.

L'H0 associée à cette deuxième question de recherche est : « il n'y a pas de différence significative entre le degré de certitude au pré-test et le degré de certitude au post-test ». En raison de la non-normalité des données[1], un test non paramétrique est privilégié afin de vérifier l'H0. Les résultats du test de Wilcoxon indiquent que les étudiants ont un degré de certitude significativement[2] plus élevé après avoir participé au dispositif technopédagogique ($p < 0,001$). Ils expriment une perception plus positive concernant leur capacité de réussite au test.

[1] La non-normalité a été confirmée pour l'ensemble des variables par le test de Kolomogorov-Smirnov ($p < 0.001$)

[2] Intervalle de confiance = 95%

4.3 Réponse à la question 3 : Quels sont les liens entre les scores obtenus aux Computational Thinking Tests et le degré de certitude ?

Nous mettons en évidence, l'existence d'un lien positif significatif entre la performance mesurée par le score total au CTT des apprenants et leur degré de certitude quant à leur réussite à la fin du test, aussi bien au pré-test ($r = 0,602$; $p < 0,001$) qu'au post-test ($r = 0,723$; $p < 0,001$). Ainsi, l'augmentation de la corrélation au post-test indique que cette relation se renforce entre le début et la fin de la formation. Ces observations suggèrent que la formation a un impact positif sur le plan métacognitif, en alignant davantage la perception des étudiants avec l'évaluation objective de leur réussite au test.

4.3.1 Liens entre les scores au pré-test et le degré de certitude au pré-test

Le tableau 3 présente le modèle issu de l'analyse de régression multiple. La variable prédite est le degré de certitude. Dans ce modèle, nous constatons que la puissance estimée à partir du R^2 est de 0,429.

En ce qui concerne ces variables explicatives, le modèle indique que cinq prédicteurs sont significatifs et un prédicteur à la limite de la significativité. Ils permettent d'expliquer 42,9% du degré de certitude au prétest ($R^2 = 0,429$; $R = 0,655$; $p < 0,001$). Nous pouvons mettre en évidence qu'un score élevé au niveau des séquences de base (**β** = 0,253 ; p=0,005), des boucles « for » (**β** = 0,151 ; p= 0,078), des conditions « if/else » complexes (**β** = 0,214 ; p= 0,014), des boucles « while » (**β** =0,168 ; p= 0,044) et des fonctions (**β** = 0,226 ; p=0,010) ainsi qu'un score faible au niveau des conditions « if » simples (**β** = -0,170 ; p= 0,046), permet de prédire le degré de certitude au pré-test.

Tableau 3. analyses de régression multiple : Modèle explicatif du niveau de réussite perçu au prétest

Prédicteurs		*Bêta*	*Taux de signification*	*Tolérance*
1	Directions et séquences de base	0,253	0,005	0,628
2	Boucles « for »	0,151	0,078	0,692
3	Conditions « if » simples	-0,17	0,046	0,695
4	Conditions « if/else » complexes	0,214	0,014	0,679
5	Conditions « while »	0,168	0,044	0,729
6	Fonctions	0,226	0,010	0,67

Parmi les différentes variables, l'examen des coefficients Bêta montre que ce sont les scores aux séquences de base, aux conditions « if/else » complexes et aux fonctions qui ont le plus de poids dans l'explication du degré de certitude. Le coefficient Bêta négatif observé pour les conditions « if » simples soulève une question. Pour expliquer ce résultat, nous pouvons avancer l'idée que les étudiants, au début de leur apprentissage, ne disposent pas d'une perception précise de leur propre niveau de réussite.

4.3.2 Liens entre les scores au post-test et le degré de certitude au post-test

Le tableau 4 présente la même analyse au terme de l'apprentissage. La lecture de celui-ci montre que quatre variables significatives permettent d'expliquer 55.1% du degré de réussite au posttest ($R^2 = 0,551$; $R=0,742$; $p <0,001$). Les scores élevés aux items boucle « for » ($\beta = 0,142$; $p= 0,001$), conditions « if » simples ($\beta = 0,196$; $p= 0,005$), conditions « if/else » complexes ($\beta = 0,256$; $p= 0,001$) et boucles « while » ($\beta = 0,331$; $p= 0,001$) expliquent la perception de réussite au posttest. D'un point de vue didactique, on observe que les items les plus complexes c'est-à-dire conditions « if/else » complexes et boucles « while » ont le plus de poids dans l'explication. Sur le plan métacognitif, ce résultat laisse à penser que les étudiants ont développé un meilleur jugement du niveau de difficulté du test et sont en mesure de mieux estimer, au terme de la formation, leur performance réelle.

Tableau 4. analyses de régression multiple : Modèle explicatif du niveau de réussite perçu au post-test

Prédicteurs		*Bêta*	*Taux de signification*	*Tolérance*
1	Boucles « while »	0,142	0,001	0,817
2	Conditions « if » simples	0,196	0,005	0,803
3	Conditions « if/else » complexes	0,256	0,001	0,78
4	Conditions « while »	0,331	0,001	0,656

A ce stade, nous considérons que certaines variables d'entrée deviennent prépondérantes dans la prédiction du degré de certitude.

4.3.3 Liens entre les scores au pré-test, le degré de certitude au pré-test, les scores au post-test et le degré de certitude au post-test

Pour aller plus loin dans notre réflexion, nous nous questionnons sur les variables ayant le plus d'influence dans la prédiction du degré de certitude au post-test. Notamment, au regard des performances initiales élevées, nous nous demandons si le degré de certitude est lié au parcours antérieur des étudiants, déterminé au regard du score au CTT et du degré de certitude au pré-test, ou au dispositif, déterminé par le score au post-test.

Le tableau 5 indique que cinq variables significatives permettent d'expliquer 70,8% du degré de certitude au post-test (R^2= 0,708 ; R=0,842 ; p <0,001). On peut observer que la variable qui a le plus de poids dans l'explication est le niveau de certitude par rapport à leurs compétences en PI au départ (β = 0,482 ; p = 0,001). L'examen du tableau montre que cette perception initiale interagit positivement avec la maîtrise de 4 dimensions du CTT dont une relative aux boucles (while) et trois relatives aux conditions (« if » simples, « if/else » complexes, while) pour expliquer leur compétence perçue au terme de l'apprentissage. Sur la base du tableau 5, on peut mettre en avant qu'il s'agit de quatre concepts plus complexes à mettre en œuvre d'un point de vue cognitif et où la maîtrise est la plus hétérogène chez les apprenants.

Tableau 5. analyses de régression multiple : Modèle explicatif du niveau de réussite perçu au post-test (2)

Prédicteurs		*Bêta*	*Taux de signification*	*Tolérance*
1	Boucles « while »	0,184	0,001	0,799
2	Conditions « if » simples	0,218	<0,001	0,799
3	Conditions « if/else » complexes	0,155	0,009	0,751
4	Conditions « while »	0,15	0,023	0,592
5	Degré de certitude au pré-test	0,482	<0,001	0,707

5. Perspective

Ces premiers résultats apportent des réponses aux défis posés par notre dispositif technopédagogique qui ouvrent la voie à futures recherches.

Au niveau de la conception du dispositif, nous avons intégré dans notre scénario pédagogique des activités de découverte et d'apprentissage des concepts clés favorisant le développement de la PI (Tchounikine, 2017). Malgré l'absence de groupe témoin, les résultats de cette expérimentation révèlent une progression significative pour tous les concepts mesurés par le CTT. Bien qu'un niveau global initial assez élevé soit constaté, nous observons une moyenne des gains relatifs de 34,5%, indiquant un effet positif de l'apprentissage (D'Hainaut, 1975).

L'examen des niveaux de maîtrise initiaux montre une bonne maîtrise dans la majorité des dimensions du CTT, avec des scores plus faibles pour les dimensions relatives aux conditions « if » simples, aux conditions « if/else » complexes et aux conditions « while ». Si nous observons une progression positive entre le niveau initial et le niveau final pour ces 3 dimensions, les niveaux de maitrise indiquent cependant que l'acquisition de ces concepts se révèle plus complexe pour les étudiants. Il est donc nécessaire de porter une attention particulière à ces concepts dans la régulation de notre scénario pédagogique, en intégrant des activités spécifiques visant à leur développement.

L'analyse de la perception de la réussite montre également une progression significative entre le niveau initial et le niveau final. Ces résultats, a priori cohérents avec la progression des performances, indiquent que les étudiants semblent plus sûrs d'eux dans les réponses qu'ils fournissent. Sur la base de notre analyse corrélationnelle, nous observons une plus grande cohérence entre ce qu'ils expriment via leur degré de certitude et ce qu'ils réalisent effectivement. Les étudiants ont ainsi une meilleure conscience du niveau de difficulté du test et sont en mesure de mieux estimer, au terme de la formation, leur performance réelle.

Au niveau du contenu de la formation, nous avons favorisé la découverte complémentaire de trois types de dispositifs tels que préconisés par Sigayret et al. (2021). Ces dispositifs incluent : (1) les activités débranchées, (2) les logiciels de programmation, et (3) les objets tangibles programmables tels que les robots pédagogiques. D'un point de vue organisationnel, nous avons combiné deux modalités : des séances en distanciel et des séances en présentiel. Ce choix semble faciliter une meilleure assimilation des concepts par les étudiants. Cette hypothèse devra être vérifiée en proposant, par exemple, des dispositifs avec des modalités pédagogiques différentes afin de comparer les effets des diverses approches pédagogiques utilisées dans ce dispositif. Au niveau de l'évaluation, nous nous sommes basés sur le CTT, un outil validé qui offre l'avantage d'une passation rapide et permettant d'objectiver le score en PI à des moments clés du dispositif. Nous pensons que l'intégration d'un item relatif à la perception de la compétence (degré de certitude) offre une piste didactique intéressante pour mieux comprendre la progression des étudiants sur le plan métacognitif. Nos résultats corroborent les observations de Fessard et al. (2019) qui mettent en évidence les difficultés de compréhension de la structuration d'un algorithme chez les apprenants. Nous observons également cette tendance avec des apprenants adultes dans le contexte d'une formation en technologie de l'éducation.

6. Conclusion

Cette étude a permis d'évaluer l'effet d'un dispositif technopédagogique sur le développement de la PI chez les étudiants du Master en Sciences de l'Éducation, spécialité Technologie de l'Éducation. Les résultats montrent une progression significative des performances au CTT, ainsi qu'une amélioration significative de leur perception de réussite. Ces résultats nous donnent la possibilité de répondre positivement à nos questions de recherche, objectivant que le dispositif pédagogique a non seulement renforcé les performances des étudiants pour l'ensemble des concepts mesurés par le CTT, mais a aussi influencé leur sentiment de maitrise objectivé par le degré de certitude.

Les activités proposées, tant en présentiel qu'à distance, ont favorisé un apprentissage des concepts de PI. L'amélioration observée dans les scores du CTT semble confirmer l'efficacité de notre approche hybride, combinant théorie, pratique et analyse réflexive qui passe par l'intégration d'activités variées et structurées pour optimiser l'apprentissage des concepts du Computational Thinking test.

Bien que contextualisée, cette recherche exploratoire présente certaines limites. Elle ne mesure pas spécifiquement l'effet de chaque modalité d'apprentissage proposée dans le scénario pédagogique et le plan expérimental ne comporte pas de groupe témoin. Néanmoins, nous estimons qu'elle peut être utile pour les formateurs d'enseignants en termes d'articulation des activités et de méthodologie pour évaluer les progrès des apprenants engagés dans leur formation. L'exploitation des traces d'apprentissage en cours de processus nous semble être une piste pertinente pour mieux comprendre d'un point de vue qualitatif le développement de la PI des apprenants.

7. Bibliographie

Aho, A. V. (2012). Computation and Computational Thinking. *The Computer Journal, 55*(7), 832-835. https://doi.org/10.1093/comjnl/bxs074

Amarelle, C. (2018). Message de la conseillère d'Etat. Dans G. Parriaux, J.-P. Pellet, G.-L. Baron, E. Bruillard & V. Komis (Eds.), De *0 à 1 ou l'heure de l'informatique à l'école* (p. 17-19). Peterlang.

Bellegarde, K., et Boyaval, J. (2020 5-7 février). *Initier des jeunes élèves à la robotique/informatique : gestes professionnels et agir enseignant* [Communication]. DIDAPRO 8 - L'informatique, objets d'enseignements – enjeux épistémologiques, didactiques et de formation, Villeneuve d'Ascq, France. https://www.didapro.org/8/wp-content/uploads/sites/4/2020/02/Bellegarde_23.pdf

Ben Henda, M. (2016, 26-28 octobre). *L'enseignement du code informatique à l'école : Prémices d'un humanisme numérique congénital* [communication]. 5e rencontre annuelle d'ORBICOM, Paris, France. https://hal.science/hal-01516577/document

Bernier, J. J. (1971). *Le degré de certitude et le degré d'information comme facteurs de discrimination dans un test objectif* [thèse de doctorat, Université de Laval]. Érudit. https://www.erudit.org/fr/theses/laval/1971/

Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., et Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (WiPSCE '17) *(p. 65–72).* https://doi.org/10.1145/ 3137065.3137069

Bugmann, J., et Karsenti, T. (2018). Apprendre à programmer un robot humanoïde : impacts sur des élèves de l'adaptation scolaire. *Formation et profession : revue scientifique internationale en éducation, 26*(1), 26-42. https://doi.org/10.18162/fp.2018.460

Çetin, İ., Otu, T., et Oktaç, A. (2020). Adaption of the computational thinking test into Turkish. *Turkish Journal of Computer and Mathematics Education, 11*(2), 343-360. https://dergipark.org.tr/en/download/article-file/1079231

Charlier, B., Deschryver, N., et Peraya, D. (2006). Apprendre en présence et à distance : une définition des dispositifs hybrides. *Distances et Savoirs, 4*(4), 469-496. https://tecfa.unige.ch/tecfa/teaching/bachelor_74111/Cours_2010_2011/semestre1/cours11/charlier_deschryver_peraya_2006.pdf

Chaker, R., et Mbadjoin, T. N. (2020). Robots scolaires et configuration spatiale de la classe : Effets sur les interactions sociales dans le cadre de séquences pédagogiques au CE2. *Didapro 8 – DidaS-TIC : L'informatique, objets d'enseignements – enjeux épistémologiques, didactiques et de formation* (p. 66-78), Lille, France. https://hal.science/hal-03585614/document

Chevalier, M., Giang, C., Piatti, A., et Mondada, F. (2020). Fostering computational thinking through educational robotics: A model for creative computational problem solving. *International Journal of STEM Education*, *7*(39). https://doi.org/10.1186/s40594-020-00238-z

D'Hainaut, L. (1975). *Des fins aux objectifs de l'éducation*. Labor Nathan.

Develay, M. (1994). Peut-on former les enseignants ? *Recherche & Formation, 16*, 148-151.

Dillenbourg, P. (2018). Pensée computationnelle : Pour un néopapertisme durable car sceptique. Dans G. Parriaux, J.-P. Pellet, G.-L. Baron, E. Bruillard et V. Komis (dir..), *De 0 à 1 ou l'heure de l'informatique à l'école* (pp. 17-19). Peter Lang.

Drot-Delange, B. (2013). Enseigner l'informatique débranchée : analyse didactique d'activités. *Actualité de la Recherche en Éducation et en Formation* (p. 1-13), Montpellier, France. https://archivesic.ccsd.cnrs.fr/sic_00955208v1/document

Drot-Delange, B., Pellet, J.-P., Delmas-Rigoutsos, Y., et Bruillard, E. (2019). Pensée informatique : Points de vue contrastés. *Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation, 26*(1), 39-61. https://doi.org/10.23709/sticef.26.1.1

Dumont, M. (2021). *Pour un enseignement efficient de l'orthographe en formation initiale des enseignants. Analyse des effets de l'intégration de démarches métacognitives sur l'apprentissage et les perceptions des étudiants* [thèse de doctorat]. Université de Mons. HAL. https://hal.science/tel-03227928

El-Hamamsy, L., Zapata-Cáceres, M., Barroso, E. M., Mondada, F., Zufferey, J. D., et Bruno, B. (2022). The Competent Computational Thinking Test: Development and Validation of an Unplugged Computational Thinking Test for Upper Primary School. *Journal Of Educational Computing Research*, *60*(7), 1818-1866. https://doi.org/ 10.1177/07356331221081753

Fédération Wallonie-Bruxelles. (2022). *Formation manuelle, technique, technologique et numérique*. Fédération Wallonie-Bruxelles. http://www.enseignement.be/index. php?page=23827&do_id=17242&do_check=CNEJLFQGEC

Fessard, G., Wang, P., et Renna, I. (2019, 04-07 juin). Objet tangible ou simulation numérique : Deux simulations équivalentes pour l'apprentissage de la programmation ? Dans J. Broisin, E. Sanchez, A. Yessad et F. Chenevotot (dir.), *Actes de la 9ème Conférence sur les Environnements Informatiques pour l'Apprentissage Humain : Données numériques et prises en compte de l'apprenant dans les environnements informatiques pour l'apprentissage humain* (p.. 223-234), Paris, France. https://eiah2019.sciencesconf.org/data/pages/ActesEIAH2019_V4.0.pdf

Focant, J. (2007). Chapitre 2. La mesure des processus d'autorégulation : quelles méthodes ? quels enjeux ? Dans N. Nader-Grosbois (dir.), *Régulation, autorégulation, dysrégulation : Pistes pour l'intervention et la recherche* (p. 31-42). Mardaga.

Frison, P., Daoud, M., et Adam, M. (2018). Transition didactique de l'activité débranchée à la programmation avec AlgoTouch. Dans G. Parriaux, J.-P. Pellet, G.-L. Baron, E. Bruillard et V. Komis (dir.), *De 0 à 1 ou l'heure de l'informatique à l'école* (p. 273-289). Peter Lang.

Lockwood, J., et Mooney, A. (2017). *Computational thinking in education: Where does it fit? A systematic literary review* [working paper]. arXiv. https://arxiv.org/abs/1703.07659

Moreno-León, J., Román-González, M., et Robles, G. (2018). *On computational thinking as a universal skill: A review of the latest research on this ability.* IEEE Global Engineering Education Conference (CN), 1684-1689. https://doi.org/10.1109/ EDUCON.2018.8363437

Papert, S. (1981). Computers and computer cultures. *Creative Computing, 7*(3), 82-92. http://dailypapert.com/wp-content/uploads/2014/04/Computers-and-Computer-Cultures-March-1981.pdf

Peter, Y., Léonard, M., et Secq, Y. (2019, 4-7 juin). *Reconnaissance de Motifs et Répétitions : Introduction à la Pensée Informatique* [communication]. Conférence Environnements informatiques pour l'apprentissage humain, Paris, France. https://hal.archives-ouvertes.fr/hal-02151035

Román-González, M. (2015, 6-8 juillet). *Computational thinking test: Design guidelines and content validation* [communication]. Proceedings of the 7th International Conference on Education and New Learning Technologies (EDULEARN15) Barcelone, Espagne. https://www.researchgate.net/publication/290391277_COMPUTATIONAL_THINKING_TEST_DESIGN_GUIDELINES_AND_CONTENT_VALIDATION

Román-González, M., Pérez-González, J. C., et Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in human behavior*, *72*, 678-691. https://doi.org/10.1016/j.chb.2016.08.047

Romero, M., et Vallerand, V. (2016). *Guide d'activités technocréatives pour les enfants du 21e siècle* (Vol. 1). https://lel.crires.ulaval.ca/works/guidev1.pdf

Romero, M., Lille, B., Viéville, T., Duflot-Kremer, M., de Smet, C., & Belhassein, D. (2018, 27-29 août). Analyse comparative d'une activité d'apprentissage de la programmation en mode branché et débranché [communication]. *Educode - Conférence internationale sur l'enseignement du numérique et par le numérique*, Bruxelles, Belgique. http://hal.inria.fr/hal-01861732

Shute, V. J., Sun, C., et Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22,* 142-158. https://doi.org/10.1016/j.edurev.2017. 09.003

Sigayret, K., Tricot, A., et Blanc, N. (2021, 7-9 juin). *Pensée informatique et activités de programmation : quels outils pour enseigner et évaluer ?* [communication]. Conférence Environnements informatiques pour l'apprentissage humain, Fribourg, Suisse. https://hal.science/hal-03241689

Stafford, J., et Bodson, P. (2006). *L'analyse multivariée avec SPSS*. Presses de l'Université du Québec.

Sullivan, F. R., et Heffernan, J. (2016). Robotic Construction Kits as Computational Manipulatives for Learning in the STEM Disciplines. *Journal of Research on Technology in Education, 48*(2), 105-128. https://doi.org/10.1080/15391523.2016.1146563

Tchounikine, P. (2017). *Initier les élèves à la pensée informatique et à la programmation avec Scratch* (Version 2). http://lig-membres.imag.fr/tchounikine/PenseeInformatique Ecole.html

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2010). Computational thinking: What and why? *The Link Magazine*. http://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Zapata-Caceres, M., Martin-Barroso, E., et Roman-Gonzalez, M. (2020, 27-30 avril). *Computational Thinking Test for Beginners: Design and Content Validation.* EDUCON, Porto. https://ieeexplore.ieee.org/document/9125368

Zhang, S., et Wong, G. K. W. (2023). Development and validation of a computational thinking test for lower primary school students. *Educational Technology Research And Development*, *71*(4), 1595-1630. https://doi.org/10.1007/s11423-023-10231-2