https://ww2.ac-poitiers.fr/math/spip.php?article1367 - Auteur : Xavier Garnier

Parcours clé-en-main « De Blockly à Python : réussir la transition »

publié le 27/01/2026

Retour d'expérience en classe de seconde

Descriptif :

Présentation d'un retour d'expérience en classe de seconde sur les parcours de programmation proposés par la plateforme Citizen Code.

Sommaire :

Contexte de l'expérimentation

Cet article s'inscrit dans la continuité de deux publications consacrées à l'usage de Citizen Code Python en classe de mathématiques au lycée.

Ces deux premiers usages ont permis de mettre en évidence plusieurs apports de la ressource comme l'engagement des élèves, l'intérêt des feedbacks immédiats, le droit à l'erreur ou la confiance progressive dans la capacité à programmer. Ils ont également soulevé une question restée en partie ouverte : dans quelle mesure ces activités permettent-elles aux élèves de structurer des apprentissages en algorithmique et de les réinvestir au-delà de la résolution immédiate de missions ?

Si le parcours exploratoire favorise l'entrée dans l'activité et la motivation, il reste parfois difficile, du point de vue de l'enseignant, d'identifier précisément ce que les élèves apprennent, ce qu'ils sont capables d'expliquer, et ce qu'ils pourront mobiliser ultérieurement dans d'autres contextes.

C'est dans ce cadre qu'a été menée l'expérimentation présentée ici, centrée sur l'usage d'un parcours pédagogiques clés en main intitulé « De Blockly à Python : réussir la transition » et proposé à des élèves de Seconde. L'objectif est d'analyser ce que change ce type de parcours, non seulement en termes d'engagement, mais aussi du point de vue des apprentissages perçus par les élèves, de la place de l'explicitation et du rôle de l'enseignant.

- Lien vers l'article "Programmer avec Citizen Code Python - des blocs au texte"
- Lien vers l'article "Faciliter la transition collège-lycée en algorithmique et programmation"

Des parcours pédagogiques clés en main : rapide panorama

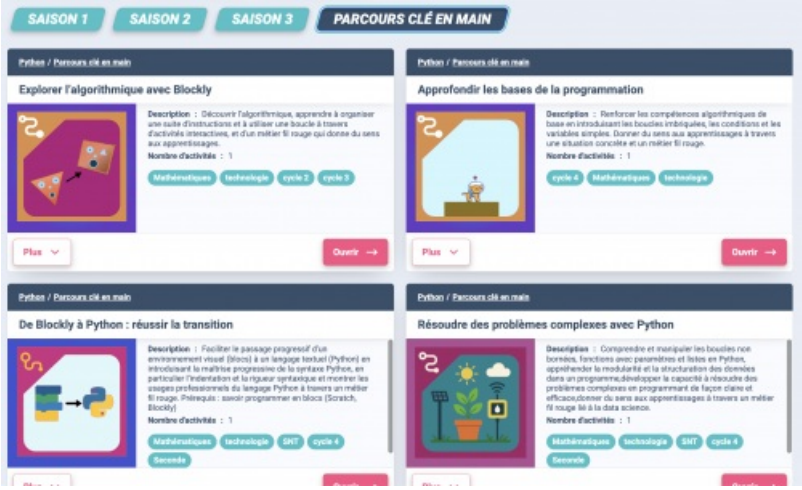
Citizen Code Python propose désormais plusieurs parcours pédagogiques structurés, conçus pour accompagner les élèves dans des étapes identifiées de leur apprentissage de l'algorithmique et de la programmation.

Quatre parcours sont actuellement proposés, chacun répondant à des objectifs et à des niveaux différents :

- un parcours d'initiation (cycle 3), destiné à découvrir les bases de l'algorithmique à travers la programmation par blocs ;
- un parcours de consolidation (cycle 4), visant à renforcer les notions fondamentales (instructions, boucles,

conditions) par des situations variées ;

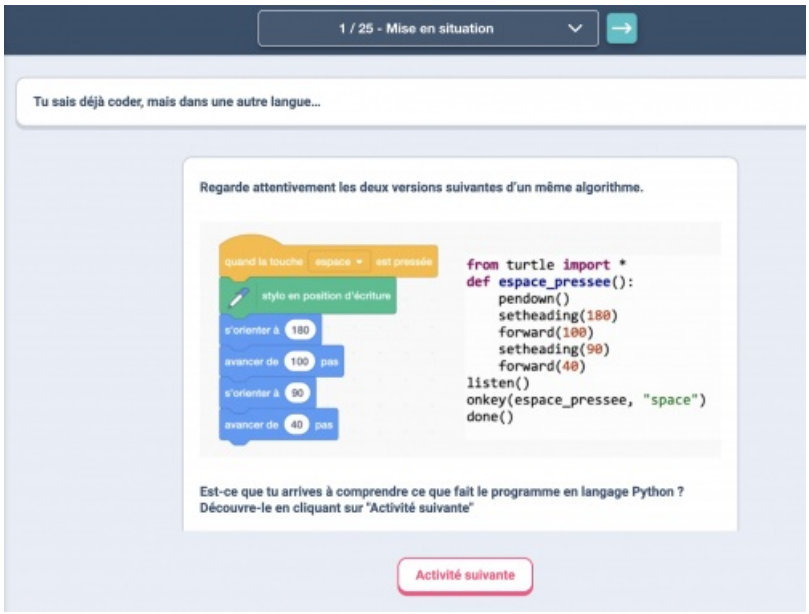
- un parcours de transition Blockly → Python (liaison 3e-2de), spécifiquement pensé pour accompagner le passage d'un langage visuel à un langage textuel ;
- un parcours orienté vers des usages plus autonomes (cycle terminal).



Le parcours « De Blockly à Python : réussir la transition » choisi pour cette expérimentation répond à un enjeu bien identifié au lycée : permettre aux élèves de comprendre que le passage au langage Python ne constitue pas une rupture avec ce qui a été travaillé au collège, mais une continuité, l'algorithmique précédant toujours le langage de programmation lui-même.

Ce parcours se distingue notamment par :

- l'alternance d'activités de programmation et de questions de type QCM permettant de faire le point sur les notions en jeu ;
- des moments de synthèse explicitant les liens entre blocs et instructions Python ;
- des temps de verbalisation et de mise à distance, destinés à aider les élèves à identifier ce qu'ils ont appris ;
- l'intégration d'une courte vidéo métier présentant le travail d'une développeuse logicielle, visant à donner du sens aux apprentissages et à ouvrir des perspectives.



● L'expérimentation en classe

L'expérimentation a été menée en classe de Seconde générale, dans un cadre ordinaire de cours de mathématiques. En amont, tous les élèves avaient été inscrits à Citizen Code Python et le parcours « De Blockly à Python : réussir la transition » leur avait été attribué via l'interface professeur.

La consigne était identique pour tous :

> *À l'issue du devoir sur table, dès que le travail est terminé, accéder au parcours depuis l'onglet « mes assignations », le réaliser intégralement, puis répondre au questionnaire de fin de parcours.*

Les élèves ne terminant pas le devoir au même rythme, quelques élèves ont dû terminer le parcours à la maison. *(NB : ce n'est pédagogiquement pas idéal. Cependant, ce choix a aussi permis à tous les élèves d'être en situation de travail et de terminer le devoir sur table à leur rythme.)*

Méthodologie de recueil des données

Afin d'analyser l'impact du parcours du point de vue des élèves, un questionnaire a été proposé à l'issue de l'expérimentation via un formulaire en ligne distinct de la plateforme. À date, 15 réponses ont été saisies.

Le questionnaire visait à recueillir :
• le ressenti des élèves sur le parcours (intérêt, difficulté, utilité) ;
• leur perception des apprentissages réalisés ;
• leur point de vue sur certains choix pédagogiques du parcours (questions de type QCM, moments de résumé, aides disponibles, rôle du professeur).

Le questionnaire :

bilan_du_parcours_citizen_code_python (PDF de 84 ko)

Ce que disent les élèves : premiers résultats

- **Durée**

La majorité des élèves indique avoir passé entre 30 et 60 minutes sur le parcours, avec quelques réponses situées entre 15 et 30 minutes.

Le parcours reste donc compatible avec un temps de travail raisonnable à l'échelle d'une séance. Comme on s'y attend, les élèves qui ont réalisé une bonne partie du parcours seuls à la maison sont ceux qui ont mis le plus de temps, sans l'aide du professeur.

- **Continuité collège–lycée**

À la question « Le parcours s'appuie sur ce que j'ai fait au collège », 2/3 des élèves se déclarent plutôt ou totalement d'accord. Quelques réponses plus critiques apparaissent néanmoins. Elles peuvent s'interpréter de plusieurs manières : profils déjà à l'aise en Python, acquis du collège peu fiables, ou encore élèves n'ayant pas identifié explicitement les liens entre blocs et texte.

- **Place des QCM et des moments de synthèse**

Deux items ressortent de manière particulièrement nette :
- « Les questions de type QCM sont utiles pour faire le point et apprendre des choses » ;
- « Les moments de résumé sont importants pour apprendre des choses ».

Sur ces deux affirmations, une majorité d'élèves exprime un accord marqué. Ce résultat indique que les élèves ne perçoivent pas ces moments comme des interruptions de l'activité de programmation, mais comme des temps utiles pour comprendre ce qu'ils sont en train de faire. Autrement dit, les élèves identifient un intérêt à ne pas se limiter à l'enchaînement de missions, mais à disposer de temps d'explicitation et de synthèse pour structurer leurs apprentissages.

- **Programmation : plaisir et utilité perçue**

Les activités de programmation sont majoritairement jugées amusantes, mais surtout utiles pour apprendre à

programmer en Python. Ce double regard est intéressant : l'engagement ludique est présent, mais il ne masque pas l'objectif d'apprentissage.

- **Sentiment de progression et rapport à la programmation**

À l'issue du parcours, une majorité d'élèves exprime un intérêt accru pour la programmation. Les élèves se sont exprimés librement sur la question "qu'avez-vous appris" et, globalement, ils ont bien identifié l'objectif d'apprentissage autour de la programmation en Python.

Qu'avez-vous appris ?
J'ai appris les bases du langage python.
J'ai mieux compris comment fonctionnais les lignes de codes
Les bases du language phyton
j'ai appris un peu plus le langage python
Rien de nouveau
A mieux savoir utiliser python
certaine formule python
cela m'a permis de commencer à avoir un peu plus une logique mathématique.
Comment coder sur Python
a maitriser quelques aspects de python.
des nouvelles commandes
Le python c'est chouette mais avec modération
J'ai appris comment fonctionne un algorithme et a faire des boucles
Comment utiliser le programme Python
J'ai appris à mieux programmer et à être plus à l'aise en programmation.

Cependant, seulement 1/3 des élèves se sentent plus forts en programmation Python après avoir réalisé le parcours. Cela peut témoigner d'une nécessité de consolidation bien perçue par les élèves, un besoin de s'entraîner bien légitime. Il ne semble pas que ce soit le but de ce parcours cependant que de viser une maîtrise du langage. C'est plutôt une porte d'entrée pour montrer que programmer en Python n'est pas sorcier, prendre un bon départ et mettre en avant les concepts algorithmiques mis en jeu.

- **Rôle du professeur : un point de tension révélateur**

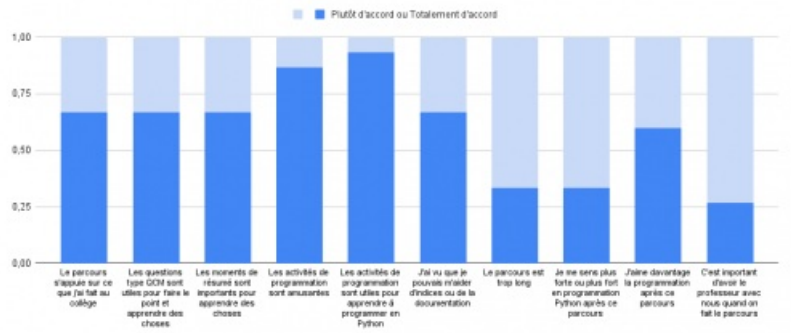
L'item « C'est important d'avoir le professeur avec nous quand on fait le parcours » fait apparaître une tendance de réponse à la négative. C'est sûrement ce qui est visé par la dénomination d'un parcours "clé-en-main". Cependant, côté professeur, certaines aides apportées en classe on permis un échange utile aux apprentissages des élèves demandeurs.

- **Éléments d'interprétation didactique**

Les réponses des élèves montrent clairement que les questions de type QCM et les moments de résumé sont perçus comme utiles pour apprendre. Ces temps intermédiaires jouent un rôle spécifique : ils obligent à ralentir l'action, à revenir sur ce qui a été fait, à identifier des invariants et à mettre des mots sur des procédures jusque-là implicites.

Ce point fait écho au triptyque *manipuler – verbaliser – abstraire* du plan Villani-Torrossian.
Les missions de programmation permettent de manipuler et d'expérimenter. Les QCM, les temps réflexifs et les synthèses contribuent à la verbalisation, puis à une première forme d'abstraction, en aidant les élèves à distinguer la tâche réalisée de la notion algorithmique sous-jacente.

Dans cette perspective, le parcours ne vise pas seulement à faire réussir une suite d'activités, mais à créer des conditions favorables à l'institutionnalisation des savoirs : identifier ce qui doit être retenu, généralisé et réutilisé. C'est un point essentiel si l'on cherche à favoriser le transfert, c'est-à-dire la capacité des élèves à mobiliser ces connaissances dans d'autres situations que celles proposées par l'outil lui-même.



Points de vigilance et conditions de réussite

Tout d'abord, l'autonomie offerte par le parcours ne doit pas être confondue avec une mise à distance de l'enseignant. Les réponses des élèves montrent que, pour une partie d'entre eux, la présence du professeur reste importante, notamment pour lever des blocages, expliciter un feedback ou aider à comprendre une notion algorithmique sous-jacente. Un parcours structuré facilite l'autonomie, mais il ne remplace ni l'étayage, ni les temps d'institutionnalisation portés par l'enseignant.

Ensuite, l'organisation hybride (classe / maison) met en lumière des écarts d'expérience entre les élèves. Ceux qui réalisent majoritairement le parcours en classe bénéficient davantage d'interactions et de régulations immédiates, tandis que ceux qui travaillent surtout à la maison peuvent avancer plus librement, mais au risque de rester seuls face à certaines incompréhensions. Ce constat n'invalide pas le dispositif, mais invite à penser les parcours comme des supports à articuler finement avec le temps de classe, selon les objectifs visés.

Enfin, les retours qualitatifs montrent que certains élèves souhaitent aller plus loin (parcours plus complexes, défis supplémentaires), tandis que d'autres pointent un besoin d'explications renforcées sur certaines étapes. Cela rappelle qu'un parcours clé en main ne peut être qu'un point d'appui, à adapter, compléter ou prolonger en fonction des profils d'élèves et des intentions pédagogiques.

Conclusion

Comparé à un usage centré uniquement sur la résolution de missions ou l'écriture de programmes, le parcours semble favoriser un ancrage plus explicite des apprentissages algorithmiques, en particulier grâce aux temps de verbalisation et d'institutionnalisation intégrés. Les élèves reconnaissent l'utilité de ces moments pour comprendre, faire le point et progresser.

Pour autant, les résultats rappellent que ces parcours ne constituent pas une solution clé en main au sens strict. Leur efficacité dépend fortement des conditions de mise en œuvre, de la place laissée à l'accompagnement de l'enseignant et de l'articulation avec le travail en classe. Utilisés comme supports d'une réflexion pédagogique plus large, ils offrent néanmoins des perspectives intéressantes pour structurer l'apprentissage de l'algorithmique et de la programmation au lycée, dans une logique de continuité avec le collège.

Compétences numériques mobilisées (référentiel CRCN)

Dans cette expérimentation, plusieurs compétences du Cadre de Référence des Compétences Numériques (CRCN) sont mobilisées par les élèves :
• Programmer : écrire, lire et modifier des instructions en Python à partir de situations algorithmiques variées.
• Environnement numérique
• Évoluer dans un environnement numérique : se repérer dans une plateforme d'apprentissage, accéder aux ressources, utiliser les aides et la documentation.
• Résoudre des problèmes techniques simples : interpréter un message d'erreur, ajuster un code ou une syntaxe.