Apprendre le langage python en autonomie en seconde

Thomas Renaud, Yann Mamode

Année universitaire 2018-2019

Diplôme Universitaire

Métiers de l'enseignement, de l'éducation et de la formation

Mention Second degré

Parcours : Mathématiques

Apprendre le langage python en autonomie en seconde

Présenté par Thomas Renaud et Yann Mamode

Écrit scientifique réflexif encadré par Benjamin Wack

Remerciements

Nous tenons à remercier :

- Nos tuteur et tutrice académiques Nicolas Bourdarias, Cécile Reignier et Lise Pillons pour leur patience, leur bienveillance et leur écoute de tous les jours.
- Nos tuteur et tutrice ESPE, Valérie Martelet et Hervé Gaussier pour leurs conseils avisés,
- Notre tuteur ESR, Benjamin Wack pour la pertinence de ses remarques et sa disponibilité.
- Nos collègues stagiaires pour les échanges toujours riches lors de nos rencontres à l'ESPE
- L'ensemble des formateurs de l'ESPE, Geneviève Martiel, Daniela Guiol, Stéphane Labbé, Michèle Gandit et en particulier Anne Misony.

Tables des matières

1 Introduction

Créée en mars 2013, l'École 42 offre un lieu d'apprentissage à la programmation à des promotions d'étudiants de 18 à 30 ans. En rupture avec le système scolaire classique, l'École 42 ne propose pas de cours et n'offre pas de diplôme à la fin du cursus, mais propose un dispositif d'apprentissage conçu autour de la résolution de problème dans une approche "learning by doing" où les élèves sont placés en autonomie et en responsabilité face à leurs apprentissages ([Lui, 2015]).
Devant enseigner la programmation du langage Python au lycée, nous nous sommes demandés si une approche similaire au système d'apprentissage de l'Ecole 42 pouvait être transposée dans un cadre scolaire plus classique, en particulier dans nos classes de seconde. Nous formulions l'hypothèse que nous pouvions mettre nos élèves en autonomie dans un dispositif adapté à l'apprentissage du langage Python.

Dans le cadre de cet ESR, nous étudierons la possibilité et la pertinence de mettre en place un dispositif d'apprentissage du langage Python dans lequel les élèves auront un niveau d'autonomie important.

Dans une première partie, nous définirons les concepts d'algorithmique et de programmation pour clarifier ce que l'on appelle la pensée algorithmique et la pensée informatique. Ensuite, nous ferons une brève rétrospective de l'enseignement de la programmation dans le secondaire. Puis, nous préciserons ce qui nous entendons par autonomie dans un cadre scolaire ainsi que les obstacles à prendre en compte dans la mise en place d'un dispositif d'apprentissage en autonomie. Enfin nous ferons un bref tour d'horizon des ressources favorisant l'apprentissage de la programmation et en particulier du langage Python.
Dans la deuxième partie, nous présenterons la mise en oeuvre du dispositif que nous avons choisi d'expérimenter au sein de nos classes de seconde. Enfin, dans la dernière partie, nous réaliserons un bilan de cette expérimentation à la fois du point de vue didactique et pédagogique. Nous étudierons les points forts de ce dispositif ainsi que ses limites. Puis nous proposerons les prolongements possibles à cette expérimentation.

2 Etat de l'art

Dans cette première partie, nous préciserons les significations des termes algorithme et programmation, nous conduisant à introduire la notion de pensée informatique. Ensuite, nous résumerons brièvement l'évolution de l'enseignement de l'algorithmique et de la programmation. Puis nous présenterons la notion de processus et de dispositif favorisant l'autonomie des élèves. Enfin nous présenterons différentes ressources et dispositifs pour l'enseignement de la programmation.

2.1 Les concepts d'algorithme et de programmation

Selon Gilles Dowek, "*le concept d'algorithme est un concept très ancien car les comptables égyptiens (2500 avant notre ère) utilisaient déjà des procédures faisant intervenir les quatre opérations arithmétiques pour calculer les prêts ou des héritages*" [Dowek, 2011]. Nous pourrions dire que l'algorithme est un objet mathématique ancien.

De plus, le terme trouve son origine dans le nom d'un illustre mathématicien "Al-Khwarismi", précurseur de l'algèbre au IXe siècle. Cependant depuis l'essor de l'informatique, comme le souligne Simon Modeste : "*souvent considéré comme un objet de l'informatique, l'algorithme est parfois associé à la programmation*" [Modeste, 2012].

Selon Mariam Haspekian et Claver Nijimbéré [Haspekian et Nijimbéré, 2016], l'analyse des programmes du lycée " *révèle une polysémie du terme « algorithmique », naviguant entre mathématiques et informatique, cette dernière étant elle-même réduite à sa partie « technologique*".

A ce niveau, nous pouvons expliciter les natures différentes des termes "algorithmique" (ou algorithmie terme souvent utilisé comme synonyme) et "algorithme". En effet dans les programmes du lycée ces termes sont utilisés sans être explicités.

Pour le second terme, selon Simon Modeste [Modeste, 2012] "*un algorithme est une procédure de résolution de problème, s'appliquant à une famille d'instances du problème et produisant, en un nombre fini d'étapes constructives, effectives, non-ambigües et organisées, la réponse au problème pour toute instance de cette famille.*"

Dans cette définition, nous précisons la nature d'objet d'un algorithme, dissociée de toute implémentation informatique et dont la finalité est de permettre de résoudre une famille de problème. La science qui étudie les propriétés des algorithmes en tant qu'objet ou famille d'objets est l'algorithmique.

Le champ d'application d'un algorithme peut-être étendue au delà des mathématiques. Gilles Dowek évoque de manière pragmatique "*la recette de la tarte aux pommes, qui permet de résoudre un problème : faire une tarte aux pommes*" [Dowek, 2011], en soulignant la recette est un algorithme en tant que pratique. Mais que à partir de l'instant où cette recette est écrite, l'algorithme devient un programme.

Nous percevons la nécessité distinguer les concepts d'algorithme et de programme et de préciser leur relation dans le cadre de l'enseignement de l'algorithmique et de la programmation. Selon Mariam Haspekian et Claver Nijimbéré [Haspekian et Nijimbéré, 2016] , reprenant la thèse de Briant [Briant, 2013], il existe une "*distance qui sépare l'activité initiale de résolution d'un problème mathématique, de sa programmation dans une machine*", une double transposition est nécessaire pour traduire un algorithme en programme informatique.

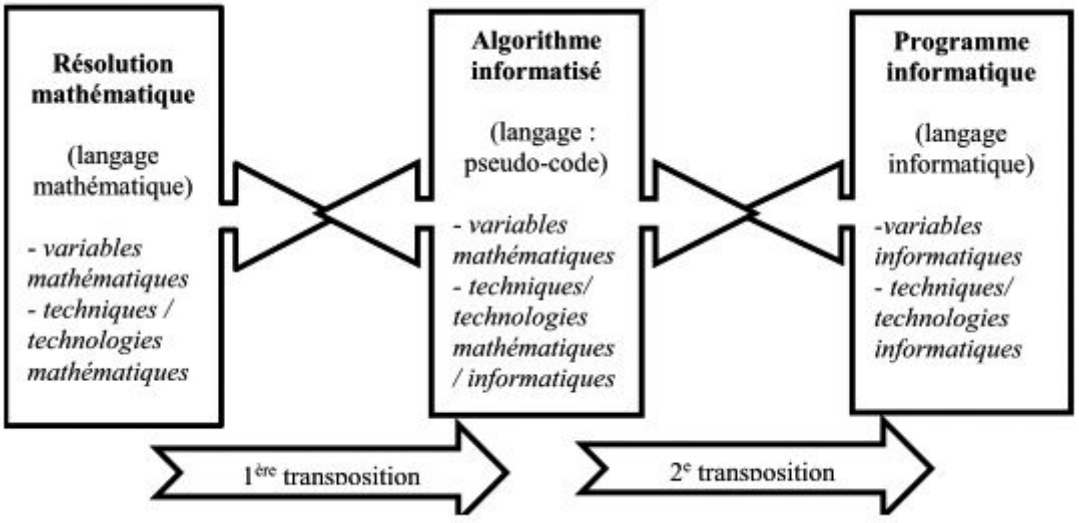


Fig. 1.– *Briant (2013), p. 85 : « Double transposition de la résolution d'un problème mathématique en vue de sa programmation »*

Chacune de ces transpositions nécessite donc de construire un langage approprié, nous pouvons utiliser le langage mathématique pour décrire un algorithme mathématique, nous pouvons aussi utiliser le langage naturel pour décrire un algorithme général. Mais comme le souligne Gilles Dowek [Dowek, 2011] : "*Décrire un algorithme dans un langage de programmation est aujourd'hui indispensable pour que cet*

algorithme soit exécuté par une machine paramétrable, car nous ne savons pas encore fabriquer de machines qui exécutent des algorithmes exprimés dans une langue naturelle."

La programmation est donc la traduction d'un algorithme dans un langage de programmation. Le programme sera donc le résultat de cette traduction qui sera exécutée sur une machine paramétrable.

Cette transposition informatique s'accompagne de la prise en compte de ce que Balacheff [Balacheff, 1994] nomme les contraintes de modélisation et d'implémentation informatiques.

Enfin en reprenant la conclusion de Briant et Bronner [Briant et Bronner, 2015], cette transposition d'un algorithme et son écriture dans un langage informatisé nécessite une transposition qui met en oeuvre une pensée distincte de la pensée mathématique que l'on peut nommer "pensée algorithmique" : "*L'existence de cette pensée spécifique nous semble être un élément didactique important qui permet de mieux comprendre la transposition qui se produit lors de la recherche d'algorithmes pour résoudre un problème*" [Briant et Bronner, 2015]. Cependant, nous pouvons préciser que cette pensée algorithmique peut aussi être mise en oeuvre sans traduction dans un langage informatique (ou pseudo-code) et être féconde pour résoudre une classe de problème mathématique.

2.2 La pensée informatique

Nous avons montré que la pensée algorithmique se distingue de la pensée mathématique. Modeste [Modeste, 2012 fev] précise en reprenant Knuth que deux aspects fondamentaux de la pensée algorithmique sont : *"la notion de complexité et l'opération d'assignation qu'il symbolise par* « := »".

D'un côté, nous avons la notion d'assignation qui introduit le concept de variable informatique qui est par nature très différentes à la variable mathématique. Cette différence doit être explicitée pour lever des obstacles didactiques lors des apprentissages liés à l'algorithmique.

D'un autre, côté, nous avons la notion de complexité d'un algorithme qui caractérise l'efficacité d'un algorithme et son effectivité pratique. Nous remarquons que cette notion n'est pas explicitement abordée dans les programmes d'algorithmique et de programmation du lycée.

A ce stade de notre étude, nous percevons que la dialectique entre la pensée algorithmique et sa transposition dans un système informatique préfigure un autre type de pensée qu'il est pertinent d'expliciter. Le terme "Computational thinking" ou "pensée informatique" est introduit par Wing J. en 2006. puis définit par elle dans les termes suivants : "*Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that*

can be effectively carried out by an information-processing agent". Dans une perspective similaire, Dowek donne les caractéristiques d'une pensée informatique : "*Ce souci du caractère algorithmique de la description des objets, du langage dans lequel ces descriptions sont exprimées, des flux d'information et des instruments [des machines] sont plus généralement caractéristiques d'une « pensée informatique »"*. [Dowek, 2011]. Cette pensée se fonde sur les quatre concepts de l'informatique : l'algorithme, le langage, l'information et la machine ; et les liens qui relient chacun de ces concepts.

Tchounikine explicite la pensée informatique en termes de compétences. Selon lui, l'enseignement de la pensée informatique doit viser à développer les compétences suivantes : "*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 (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*" [Tchounikine, 2017].

Tchounikine [Tchounikine, 2017] distingue cinq grandes catégories :

- **L'abstraction** : appréhender un problème et sa solution à différents niveaux,
- **L'algorithme** : réfléchir aux tâches à accomplir sous forme d'une série d'étapes,
- **La décomposition** : résoudre un problème complexe décomposé sous la forme de plusieurs problèmes plus simples
- **La reconnaissance de "schéma" ou de forme (pattern**) : réutiliser des éléments de solutions (algorithme ou partie d'algorithme) pour résoudre un problème nouveau.
- **La généralisation** : généraliser la solution à un problème à un ensemble de problèmes semblables

En conclusion, le concept de pensée informatique, nous permet de préciser davantage l'activité de programmation et les domaines de compétences à développer. La distinction entre une résolution mathématique d'un problème donné, sa traduction en termes algorithmiques et sa transposition en programme, nous permettra de mieux préciser les contours d'un dispositif didactique propice à l'enseignement de l'algorithmique et la programmation en prenant en compte les quatre concepts de l'informatique et les cinq grandes catégories dans la pensée informatique. Nous serons aussi attentifs aux obstacles didactiques tels que la notion d'affectation et le concept de variable informatique.

Et comme le conclut Tchounikine : "*La pensée informatique ne se réduit donc pas à l'algorithmique, mais le concept d'algorithme est au cœur de la pensée informatique*" [Tchounikine, 2017].

Et la définition de la pensée informatique semble en évolution à mesure que la discipline informatique et la technologie associée se développent : les approches itératives, les systèmes en réseaux, les capacités de traitement, les bases de données, etc...

2.3 Evolution de l'enseignement de l'algorithmique et de la programmation dans le secondaire

Nous nous proposons de faire une brève description de l'évolution de l'enseignement de l'algorithmique et de la programmation dans le secondaire depuis l'avènement de l'informatique.

Pelisset nous donne quelques repères historiques importants de l'évolution de l'enseignement de l'informatique en France. Au début des années 50 apparaissent les premiers ordinateurs dans les universités, les instituts et les laboratoires (les matériels sont principalement électromécaniques utilisant principalement des supports perforés) [Pélisset, 1985]. En 1952, dans un lycée technique se met en place une formation technique pour la maintenance de calculateurs électroniques (Bull - Gamma 3 - et IBM-604). En 1957, le Brevet de Technicien Supérieur de « la mécanographie à cartes perforées et de ses prolongements électroniques » est institué (arrêté du 26 septembre 1957). Par la suite dans les années 60, un certain nombre d'expériences seront conduites dans des lycées techniques (enseignement industriel, analyse et programmation) et très ponctuellement dans deux lycées de l'enseignement général (dès 1963 à Toulouse et Grenoble ; ainsi en 1966 et pendant deux ans, au lycée Bellevue de Toulouse, l'ordinateur est utilisé pour l'enseignement des mathématiques en 6e). L'enseignement de la programmation est donc principalement réservé aux filières techniques ou à l'enseignement supérieur.

Comme le montre Pelisset, *" Il est communément admis que l'introduction de l'informatique dans l'enseignement général français trouve son origine dans le séminaire du Centre d'études et de recherches pour l'innovation dans l'enseignement (C.E.R.I. de l'O.C.D.E. créé en 1968) au Centre international d'études pédagogiques (C.I.E.P.) de Sèvres en mars 1970 sur le thème « l'enseignement de l'informatique à l'école secondaire ». Les recommandations du séminaire soulignent l'apport de l'informatique à l'enseignement général (« une des caractéristiques de l'informatique est de créer chez les élèves une attitude algorithmique, opérationnelle, organisatrice, laquelle est souhaitable pour bien des disciplines ») et incitent les pays membres de l'O.C.D.E. à introduire l'informatique dans l'enseignement secondaire."* [Pélisset, 1985]

Nous pouvons remarquer la place centrale donnée à l'approche algorithmique dans cette réflexion sur l'enseignement de l'informatique : c'est bien "*bien la démarche informatique que l'on peut caractériser comme algorithmique, opérationnelle, organisationnelle*" [Association EPI - 2002] . La circulaire ministérielle 70-232 du 21 mai 1970 (BOEN n° 22 du 28 mai) précise que *« L'informatique est un phénomène qui est en train de bouleverser profondément les pays industrialisés et le monde moderne en général. (...) L'enseignement secondaire tout entier et dès la classe de 4e ne peut rester à l'écart de cette révolution. Il doit préparer au monde de demain dans lequel ceux qui ignoreront tout de l'informatique seront infirmes... ».* Par la suite, le MEN français mettre en place une politique d'expérimentation comportant trois volets : la formation à l'informatique d'enseignants de diverses disciplines (formation à l'algorithmique et à la programmation auprès d'industriels), une réflexion pédagogique disciplinaires sur les apports de l'informatique aux différentes disciplines, et l'équipement de matériels informatiques de 58 lycées et collèges (MITRA15 et T1600). Cette expérimentation est connue sous l'appellation "expérience des 58 lycées".

En 1976, l'expérience des 58 lycées est arrêtée par le Ministère de l'Education Nationale, notamment pour des questions de coûts relatifs à la formation des enseignants et aux matériels informatiques.

Mais, comme le souligne Baudé, *"cette expérience fut très riche, avec une implication importante d'enseignants formés (formations « lourdes »), avec une production nationale de logiciels issus d'une recherche pédagogique active... mais elle fut essentiellement axée sur «l'informatique outil ». Très peu, sauf exceptions, sur l'informatique « objet »* [Baudé, 2014]. En effet, l'expérience a contribué au développement de l'EAO (Enseignement assisté par Ordinateur) au sein de chaque discipline, c'est-à-dire l'utilisation de l'informatique comme un outil. Par contre, la sensibilisation à la démarche informatique et algorithmique ne semble s'être développée que dans les clubs informatiques créés dans le cadre de cette dynamique.

Baron et Bruillard illustrent cette dualité de l'informatique par la phrase de SeymourPapert : "*l'alternative est de programmer la machine ou d'être programmé par elle*" [Baron et Bruillard, 2011]. Cette dualité a continué à perdurer pendant les 20 années suivantes.

Au sein de l'EPI, la question "Comment envisagez-vous l'avenir de l'informatique dans l'enseignement secondaire ?" recueillait trois types de réponses :

1. *Utilisation de l'informatique comme méthode de pensée à l'intérieur des disciplines traditionnelles (44)*
2. *Utilisation de l'ordinateur comme outil dans ces mêmes disciplines (40)*
3. *Enseignement d'une discipline distincte avec horaire et programme (5)*

La troisième position est minoritaire malgré le constat de la difficulté de transmettre par les disciplines existantes une culture informatique solide et structurée.

En 1980, le rapport Simon remis au président de la République préconise un enseignement de l'informatique pour tous avec comme mesure transitoire des enseignements optionnels au collège et au lycée.

D'une part, l'enseignement de l'option informatique (OI) est introduite en 1981, puis généralisée en 1985 à tous les lycées. L'informatique est considérée comme une science, un objet d'enseignement et non comme un outil. La démarche algorithmique y est clairement introduite, l'analyse et la programmation ont une place centrale. Elle est supprimée une première fois en 1992, puis rétablie en 1995, avant d'être à nouveau supprimée en 1998. Selon Baron et Bruillard, le caractère optionnel et sélectif a conduit à cette suppression [Baron et Bruillard, 2011].

D'autre part, différents plans sont lancés pour équiper les établissements de matériels informatiques favorisant l'appropriation de l'outil informatique par le plus grand nombre. Citons le plan "10 000 micro-ordinateurs" en 1980 puis le plan "100 000 ordinateurs" en 1983 et enfin le plan "Informatique pour tous" à partir de 1984.

A partir de cette date, l'enseignement de l'informatique dans le secondaire est orienté vers la maîtrise des outils numériques : les progiciels et les outils internet. Nous retrouvons cette approche dans la mise en place du B2i (Brevet informatique et internet).

Cependant, le terme algorithme réapparaît dans les programmes du lycée dans le BO du 12 Août 1999 dans le chapitre "Statistique": "*Le calcul de la médiane nécessite de trier les données, ce qui pose de problèmes de nature algorithmique*"

Puis, en 2009, des éléments d'algorithmique sont introduits explicitement dans les programmes de mathématiques de seconde comme un axe transversal. Les enseignents doivent mettre en place des activités permettant une "*formalisation en langage naturel propre à donner lieu à traduction sur une calculatrice ou à l'aide d'un logiciel*". Il est également précisé « *Il s'agit de familiariser les élèves avec les grands principes d'organisation d'un algorithme*" sans qu'aucun langage, ni aucun logiciel ne soit imposé. La programmation semble ici plutôt rester au second plan, simplement comme une application des algorithmes étudiés.

En 2010, l'institutionnalisation d'une discipline de spécialité « informatique et sciences du numérique» (ISN) pour les classes de Terminale remet la science informatique au coeur de l'enseignement de l'informatique. Dans le prolongement des orientations formulées par l'académie des science , [Académie des sciences, 2013], le programme continue d'évoluer afin d'intégrer la pensée algorithmique au sein des

programmes. En 2017, le programme de mathématiques de classe de seconde fait l'objet d'un aménagement : l'axe transversal *algorithmique* devient une partie à part entière intitulée *algorithmique et programmation* (le programme rappelant toutefois que « le travail sur l'algorithmique et la programmation doit être réinvesti dans les trois autres parties »). Le langage Python est cité sans être explicitement imposé par le programme.

Les programmes applicables à la rentrée 2019 pour les classes de seconde expriment clairement l'orientation initiée par les enseigements ISN en affirmant clairement l'utilisation du langage de programmation python : "*Un langage de programmation est nécessaire pour l'écriture des programmes : un langage simple d'usage, interprété, concis, libre et gratuit, multiplateforme, largement répandu, riche de bibliothèques adaptées aux thématiques étudiées et bénéficiant d'une vaste communauté d'auteurs dans le monde éducatif est nécessaire. Au moment de la conception de ce programme, le langage choisi est Python version 3 (ou supérieure*)"

2.4 Les processus d'apprentissage en autonomie

Le point central de cet ESR est la question de l'autonomie de l'apprentissage. Aujourd'hui, beaucoup d'apprentissages peuvent se réaliser à un degré d'autonomie plus ou moins important, et c'est notamment le cas des langages de programmation. Les ressources étant abondantes et simples d'accès, nous avons remis en question le choix pédagogique de l'apprentissage par enseignement et la possibilité de cette remise en question en classe de seconde.

Dans un premier temps, nous définirons les différents modes d'apprentissage en autonomie, puis nous verrons les obstacles que peuvent présenter un tel choix pédagogique chez l'élève. Enfin, nous mettrons en perspective ces éléments théoriques avec l'outil France-ioi que nous avons choisi pour notre expérimentation.

2.4.1 L'apprentissage en autonomie

Comment apprendre un langage sans se le faire enseigner ? C'est la question qui découle directement de l'article de Holec [Holec, 1991] qui traite de l'apprentissage d'une langue vivante. Commençons par rappeler quelques définitions qui seront utiles pour la discussion qui va suivre. "L'apprentissage [...] est l'ensemble des activités [..] dans lesquelles s'engage quiconque qui a décidé d'acquérir une compétence". Holec le voit comme une succession d'actes permettant l'assimilation de savoirs et de savoirs faire.

Chacun de ces actes d'apprentissage se définit par les points suivants:

- **Objectifs** : Quel est l'ordre de l'acquisition visée ? acquisition d'un élément de syntaxe, d'une structure de contrôle, de résolution de problème à l'aide des moyens d'expression du langage, etc…
- **Un contenu** : Le support et son moyen d'utilisation, par exemple branché/débranché, le type de tâche (corriger un programme, le concevoir, etc...)
- **Les modalités de réalisation** : Durée de la tâche, individuelle, collective, lieu de réalisation
- **Les modalités d'évaluation** : Appréciation du résultat atteint par rapport à l'objectif visé.

Dans un apprentissage par enseignement, c'est l'enseignant qui fixe les points ci-dessus. Il choisit et agence les tâches à effectuer par l'apprenant ainsi que les modalités d'évaluation. L'apprenant n'a alors qu'à effectuer les activités proposées mais n'a pas de prise sur l'organisation, le support et le contenu de l'apprentissage. Il en est en quelque sorte "consommateur". Cette distribution des rôles est assez classique mais reste relativement rigide quant à la prise en compte des besoins individuels de chaque élève.
Il faut garder à l'esprit que cette manière d'enseigner (prédominante dans l'enseignement secondaire) reste un choix pédagogique délibéré qui doit être remis en question, à plus forte raison du fait que l'autonomie est entrée au socle commun de connaissances et de compétences sous l'intitulé "Autonomie et l'initiative".
Un mode d'enseignement, dit "**à distance**", permet plus de flexibilité : L'enseignant propose un contenu, fixe l'objectif mais c'est l'apprenant qui fixe ses propres modalités de réalisation et son évaluation. Le corollaire étant une progression plus différenciée et adaptée aux besoins de chacun.
Le troisième et dernier mode d'apprentissage selon Holec est un apprentissage **autodirigé**, qu'il ne faut pas confondre avec une autodidaxie dite "sauvage" sans guide ni repère. Ici, tous les actes d'apprentissages et leurs critères sont construits en commun par l'enseignant et l'apprenant, plaçant l'apprenant au centre de ses apprentissages.

2.4.2 L’autonomie scolaire et dispositif d’apprentissage

Dans l’analyse précédente, l’autonomie de l’apprenant est donc graduellement plus importante à mesure que l’apprenant prend une part plus forte à la construction du processus d’apprentissage avec l’enseignant.
Pour aller plus loin dans notre analyse, nous sommes amenés à nous questionner ce que pourrait être l’autonomie dans un cadre scolaire.

Selon Raab, "*nous définissons l'autonomie scolaire comme la capacité d'un élève, d'un groupe d'élèves ou d'une classe à mener une activité productive (la tâche) au service d'une activité constructive (les apprentissages) en dehors de la présence directe de l'enseignant*" [Raab, 2014].

Dans une première approche et dans le cadre de cet ESR, l'autonomisation de l'élèves n'est pas envisagée comme un objectif pédagogique a priori mais plutôt comme une modalité pédagogique s'insérant dans un dispositif pédagogique et didactique où l'élève peut agir en dehors de la présence directe de l'enseignant (réaliser les tâches définies dans le cadre) et peut récolter des feedbacks sur sa progression et valider les différentes étapes de son apprentissage en fonction des objectifs prescrits.

Cependant le processus d'autonomisation de l'élève soulève plusieurs questions : En quoi le dispositif d'apprentissage favorise une autonomisation de l'élève ? C'est-à-dire, dans quelle mesure, le dispositif permet à l'élève de franchir les obstacles qui lui seront présentés sans la présence de l'enseignant ?
Si l'on se réfère aux théories de Vygotsky, l'activité d'apprentissage autonome de l'élève doit se situer dans la zone proximale de développement de l'élève. L'enseignant doit donc veiller à ce que le dispositif fournisse les étayages pertinents au regard des apprentissages visés en fonction des élèves :

1. Le dispositif doit donc permettre à l'élève de rester dans sa zone proximale de développement. D'une part, le dispositif peut suggérer à l'élève les questions à se poser pour comprendre le problème, peut suggérer une piste de résolution, peut amorcer une étape de résolution ou encore donner un résolution complète que l'élève doit s'approprier et refaire . D'autre part, cela signifie que la progression pédagogique doit être conçue de manière progressive pour permettre à l'élève de réutiliser les apprentissages
2. De plus, un ensemble de feedbacks doit être donné à l'élève pour lui permettre de se situer dans la progression de ses apprentissages : invalider ou valider les réponses, donner des indices de la progression, donner des indices pour dépasser un obstacle.

Enfin, en quoi le dispositif permet à l'enseignant d'évaluer le niveau d'apprentissage des élèves ? Cette question se pose du point de vue de l'évaluation formative et de l'évaluation sommative. Est-ce que le dispositif permet de conduire une évaluation sommative que l'enseignant pourra exploiter pour conduire ensuite une évaluation sommative qui pourrait être complétée par une évaluation décontextualisée validant les compétences acquises par les élèves à l'issue du processus d'apprentissage en autonomie ?

2.4.4 Les stratégies d'évitement de l'obstacle

Comme le montre Raab "*les situations d'autonomie proposent un double obstacle. Le premier, de nature didactique, est propre au domaine d'apprentissage support de l'activité proposée. Il concerne l'acquisition de savoirs disciplinaires constitués : c'est l'objectif d'apprentissage (Brousseau, 1989 ; 1998). Le second est d'ordre pédagogique : il est pratique, psycho-affectif, socio-relationnel et relève de la situation elle-même*". [Raab, 2014].

De plus "*pour construire de nouveaux savoirs dans les situations d'apprentissage, l'élève doit rencontrer un obstacle dans la réalisation de la tâche. Il mobilise, pour le franchir, diverses opérations mentales – pensée déductive, inductive, dialectique, divergente, analogique – en faisant jouer les consignes données sur l'ensemble des contraintes et des ressources proposées*" ([Meirieu, 1987] ; [Astolfi, 1992]).

Le risque est donc que l'élève, sans la présence de l'enseignant, ne se confronte pas à l'obstacle didactique dans de bonnes conditions, et mette en place des stratégies d'évitement qu'il sera nécessaire de prendre en considération.

Nous pourrions observer les stratégies d'évitement suivantes:

- **L'élève refuse d'affronter l'obstacle** : Il ne cherche pas à produire et à réussir le travail demandé
    - Évitement non perturbateur : abandonne la tâche, se replie sur lui-même, travaille seulement sous la contrainte, fait semblant de travailler, refuse les aides proposées, récuse le modèle de ses pairs
    - Évitement perturbateur : attire l'attention sur lui, empêche les autres de travailler, détruit son travail
- **L'élève est stoppé dans son travail :**
    - ne comprend pas ou ne sait pas quelles actions enchaîner, ne dispose pas des connaissances requises,
    - est déstabilisé ou neutralisé par lui-même ou par les pairs : se défend, se place en spectateur s'auto-dévalorise,
    - dérive vers une autre tâche,
- **L'élève contourne l'obstacle :**
    - demande de l'aide pour que l'on finisse à sa place,
    - profite des corrections apportées à un pair pour avancer son travail et se désynchronise volontairement

2.5 La boucle essai-erreur

Astolfi rappelle qu' "*Apprendre c'est toujours prendre le risque de se tromper*" [Astolfi, 2014]. Il introduit un nouveau rapport à l'erreur est permet d'envisager d'utiliser les erreurs comme un indicateur de progression. Dans le cadre du paradigme de traitement scientifique de l'informations (PTSI) , "*l'erreur est perçue comme une information le plus souvent inattendue et susceptible d'engendrer un questionnement ou une nouvelle hypothèse*" [Favre, 1995].

Drot-Delange et Tort [Drot-Delange et Tort, 2018] montrent que la boucle "essai-erreur" mise en oeuvre par un élève dans le cas de résolution de problème peut être liée à des démarches distinctes :

1. la boucle essai-erreur est utilisée pour vérifier/valider des hypothèses/un modèle : "*L'élève efficace fait des prévisions et organise des tests pour « vérifier une solution »* ". Nous retrouvons la mise en place d'une démarche d'abstraction qui révèle que « *...l'élève a bien compris la situation et est capable de visualiser pas à pas le déroulement du programme sans l'exécuter* ». Nous pourrions aussi qualifier cette démarche de « *stratégie informatique : essai sur peu de code pour vérifier au fur et à mesure que cela fonctionne, permet de gérer plus facilement les bugs*. »
2. la boucle essai-erreur est utilisée pour obtenir le résultat par tâtonnement sans mettre en place de démarche de réflexion : « *L'élève a besoin d'exécuter le programme et corrige au fur et à mesure. Cela révèle peut-être un manque de capacités d'abstraction* ». Dans ce cas, l'efficacité passe par la capacité à identifier les erreurs et à les corriger.

Nous pouvons retrouver dans ce type de démarche un apprentissage du type "learning by doing" théorisé par John Dewey. Comme le précise [Lui, 2015] : "[Selon Dewey] *Le faire était une entrée vers les apprentissages réflexifs*".

Lors de la résolution de problème, la boucle essai-erreur (doing) devient un apprentissage quand "*l'expérience devient expérientielle*", c'est à dire quand un retour réflexif est effectué par rapport l'expérience et aux apprentissages qui en découlent (learning). Cela pourrait être conduit sous la forme d'une institutionnalisation organisé par l'enseignant avec le concours des élèves.

Cette boucle essai-erreur induit un changement de posture de la part de l'enseignant afin de favoriser l'autonomie des élèves dans ce processus d'apprentissage basé sur la boucle essai-erreur :

"*Pour le professeur, cela signifie qu'il accepte pour ses élèves la nécessité de phases de tâtonnement et d'essais-erreurs, durant lesquelles il intervient le moins possible. En particulier, il est recommandé de*

résister à l'envie de taper sur le clavier ou de manipuler la souris à la place de l'élève, et de se contenter de répondre oralement aux questions de l'élève, de lui rappeler éventuellement tel ou tel script déjà écrit qui pourrait l'inspirer ou encore de lui suggérer de nouvelles pistes pour résoudre la difficulté qu'il rencontre, mais sans intervenir directement sur la machine de l'élève." ([Eduscol, 2016])

Par contre, l'enseignant (ou le dispositif d'apprentissage) peut inviter les élèves faire un travail de réflexion avant de rentrer dans la boucle essai-erreur : simulation sur papier, identification des variables, représentation des étapes, etc... Enfin, l'enseignant (ou le dispositif d'apprentissage) pourrait demander à l'issue de la démarche d'essais erreurs aux élèves d'expliciter les apprentissages réalisés, de prendre du recul sur cette phase d'expérimentation, de tests et de validation. Cette phase permettrait d'identifier les "schémas" utilisés ainsi que les stratégies de résolution.

2.6 Les ressources favorisant l'apprentissage de la programmation

A l'origine de notre réflexion, nous nous sommes demandés quelles pouvaient être les ressources ou les sources d'informations que nous pourrions fournir ou conseiller à nos élèves pour qu'ils puissent s'initier à la programmation de manière autonome. En effet nous constatons qu'il existe de plus en plus de ressources disponibles à la fois pour les élèves et pour les enseignants.

Traditionnellement, l'apprentissage de la programmation peut se faire selon deux modalités. On distingue habituellement les activités débranchées de l'activité branchée de programmation.

Les activités débranchées permettent de créer un contexte didactique dans lequel les élèves sont conduits à réaliser des algorithmes sans utiliser de machines, à mettre en oeuvre des méthodes de codage (codage binaire par exemple), voire même à découvrir des concepts propres à l'étude des algorithmes (complexité par exemple) . " *L'idée étant de distinguer le concept d'algorithme, permettant de résoudre un problème, de celui de programme, exécuté par une machine, associé à un langage de programmation [...] . Ces concepts (algorithme, machine, langage, information) sont les bases pour l'enseignement de la discipline informatique d'après la SIF (2016)*" [Frisona et al, 2018].

Il existe plusieurs ressources de référence permettant de mettre en place des activités débranchées. Citons pour mémoire le projet **csunplugged** [Ref] créé par Tim Bell, Ian H. Witten et Mike Fellows et traduit en français et les activités débranchées fournies par **1,2,3, codez !** [Ref]. L'une et l'autre fournissent des fiches d'activités et du matériel pédagogique aux enseignants. La mise en oeuvre de ces activités présuppose la présence de l'enseignant.

Dans le cas des activités branchées, nous pouvons distinguer trois types de ressources :

- **les fiches d'activités** : fiches élèves, fiches professeurs, matériels pédagogiques associés,
- **les environnements de développement** : Il existe plusieurs types environnements offrant différentes fonctionnalités : exécution , débogage, coloration syntaxique, etc… (EduPython, Python 3 et IDLE, Pyzo, Anaconda, etc…)
- **les ressources accessibles en ligne**

Avec le développement de sites web à visée pédagogique, de nombreuses ressources sont disponibles en ligne pour apprendre l'algorithmique et la programmation, facilement accessible et dans la majorité des cas accessibles. Les ressources peuvent être en anglais mais il en existe un certain nombre en français.
Nous pouvons les classer en différentes catégories :

- Cours et Vidéos en ligne
- Environnement de programmation : http://pythontutor.com , https://cscircles.cemc.uwaterloo.ca , https://repl.it/languages/python3
- Activités de découverte : https://hourofcode.com
- Parcours de formation : http://www.france-ioi.org , https://www.codecademy.com
- Défis / jeux : codecombat.com, codewars.com, etc...

Nous avons choisi dans le suite de cette expérimentation de mettre en oeuvre la ressource en ligne proposée par http://www.france-ioi.org . Nous décrirons dans la suite de cet ESR les différentes fonctionnalités permettant à un élève de s'initier à la programmation à Python.

3 Problématique

En classe de seconde, l'algorithmique et la programmation sont enseignés les professeurs de mathématiques. Nous avons vu que l'évolution des programmes conduit à concevoir des activités et des exercices autour de la mise en oeuvre du langage Python, notamment pour permettre aux élèves de développer une pensée informatique, ce qui sous-entend de développer une pensée algorithmique. En général, les activités branchées classiques consistent à utiliser un environnement de programmation en salle informatique pour réaliser un TP guidé par le professeur de mathématiques.

D'une part, nous avons remarqué que les enseignants de mathématiques (hors enseignants ICN) sont peu ou pas formés à la programmation sous Python, ce qui les freine à mettre en place des activités de programmation sous Python : en particulier, l'appropriation du langage et de l'environnement de développement.

D'autre part, la programmation dans un langage structuré tel que Python demande une grande rigueur de rédaction et une compréhension assez fine de la part des élèves et des enseignants de ce que l'interpréteur attend. Les élèves peuvent se sentir perdus lorsque que dans l'exécution de leur programme des erreurs de l'interpréteur apparaissent. La mise en place d'une démarche essai-erreur permettrait aux élèves de construire un processus d'apprentissage leur permettant de modéliser, d'expérimenter et de tester leurs programmes tout en étant au centre de leur apprentissage, c'est à dire avec un certain degré d'autonomie.

Enfin, nous avons constaté qu'il existe de nombreuses ressources accessibles en ligne permettant à un élève d'apprendre les notions de programmation, de faire des exercices pour s'entraîner, de participer à des challenges (Algorea, Castor, etc…), etc...

A l'image de l'école 42 qui propose en apprentissage à la programmation sans enseignant, nous nous sommes demandés dans quelle mesure il était pertinent de mettre en place au sein d'une classe de seconde un dispositif dans lequel l'élève pourrait apprendre la programmation du langage Python avec un grand degré d'autonomie.

Dans le cadre de cette expérimentation nous nous sommes demandés : En quoi le dispositif que nous avons choisi est en adéquation avec le programme de seconde ? Dans quelle mesure ce dispositif permet un apprentissage en autonomie de l'élève ? En quoi le dispositif contribue au développement d'une pensée informatique ?

4 Expérimentation

4.1 Description du dispositif

Le site http://www.france-ioi.org (désigné dans la suite par **France-IOI**) est une plateforme d'apprentissage en autonomie de la programmation, des bases jusqu'à la compétition, puisque ce site héberge et organise entre autres les concours Algoréa, les olympiades internationales et le concours Castor.

La progression est organisée en une succession de cours et d'exercices de difficulté progressive et s'adapte à de nombreux langages de programmation. La résolution des exercices se fait directement dans le navigateur et la validation est effectuée automatiquement grâce à une série de tests. La place de la communauté est importante puisque tout utilisateur peut demander soit des conseils automatiques, soit des conseils à un utilisateur plus expérimenté.

France-IOI, et c'est aussi pour cela que nous l'avons choisi, s'adresse particulièrement aux enseignants et propose un outil de gestion des groupes classes et de suivi de progression performant. De plus, le site propose une progression calquée sur les programmes du lycée.

Du point de vue de l'élève :

La résolution d'un exercice se présente sous la forme d'une page possédant plusieurs onglets :

L'onglet "**Sujet**" contient l'énoncé du problème à résoudre, ainsi que toute information utile à sa résolution. Les énoncés définissent une situation problème contextualisée .

L'onglet "**Résoudre**" contient la boîte de dialogue dans laquelle taper le code du programme, ainsi que "l'interpréteur-évaluateur" permettant de valider le programme soumis. "L'interpréteur-évaluateur" retourne soit un statut "Succès" soit une erreur précisant la nature de l'erreur (Cf Annexe 2) sans donner d'indications sur la correction.

L'onglet "**Conseils**" permet à l'élève de demander un conseil automatique ou à la communauté s'il est bloqué. Les conseils sont de différentes natures selon les situation-problèmes.

L'onglet "**Activité**" se présente sous forme d'un journal de toutes les soumissions de l'élève dans l'ordre chronologique, avec pour chaque soumission le code source et les résultats des différents tests automatiques.

Enfin, l'onglet "**Correction**" présente une correction du problème. Son contenu n'est accessible qu'une fois que le problème a été résolu. Cela permet à l'élève de vérifier si la solution qu'il a trouvée est une solution optimale. Cette phase est importante pour lui permettre de prendre du recul par rapport à la stratégie de résolution qu'il a choisi de mettre en place. En effet, la démarche expérimentale essai-erreur peut permettre de construire par itération une solution valide mais qui peut s'avérer non optimale du point de vue algorithmique.

" *Nous vous recommandons de lire bien attentivement la correction et ce même si le sujet ne vous a pas posé de problème. Vous y trouverez souvent des conseils qui pourront vous être très utiles par la suite*."

Du point de vue de l'enseignant :

L'outil de gestion de la classe est un des points forts de **France-IOI**. Chaque enseignant peut créer ce que le site appelle un groupe, et permettre à des utilisateurs de le rejoindre. L'outil de gestion des groupes se présente comme suit :

Il est possible de laisser chaque groupe créé ouvert, ou bien devoir gérer chaque demande d'adhésion, ou encore le protéger par un mot de passe, qui donnera l'accès à toute personne connaissant le nom du groupe et ce mot de passe. C'est l'option que nous avons choisi, afin de ne pas avoir à gérer une à une les inscriptions. Ensuite, vous pouvez suivre la progression de l'ensemble des élèves du groupe via le tableau

de progression, qui affiche le nombre de soumission à chaque problème, ainsi que le temps passé. En cliquant sur le pseudonyme de l'élève, on a accès à son profil et on peut voir sa courbe de progression, ainsi que ses propositions de soumission. (Cf Annexe 4).

4.1.1 France-IOI et le programme de seconde

Nous allons faire ici un parallèle entre le programme de la classe de seconde et les chapitres proposés par **France-IOI**. Nous verrons quels chapitres sont utiles pour couvrir l'ensemble du programme.

Contenus	**Capacités attendus**	**Chapitres France-IOI**	**Commentaires**
Variables et instructions élémentaires	• Choisir ou déterminer le type d'une variable (entier, flottant ou chaîne de caractères) ; • Concevoir et écrire des affectations à des variables ; • Écrire une formule permettant un calcul combinant des variables.	**Le chapitre 3, intitulé « Calculs et découverte des variables »** est vraisemblablement le plus adapté à ce contenu. Il permet aux élèves de jongler avec les variables et les opérations possibles. Note : Les boucles bornées doivent avoir été vues avant.	Dans le **chapitre 4**, ("lecture d'une entrée"), le type **int** est abordé. L'utilisation de la fonction int() est faite pour traiter les entrées retournées par la fonction input() Les float sont abordés au **chapitre 1 du niveau 2 intitulé "Nombres à virgule et autres outils"**
Boucle et itérateur, instruction conditionnelle	• Programmer une instruction conditionnelle ; • Programmer une boucle bornée ; • Programmer une boucle non bornée.	Les instructions conditionnelles sont abordées au **chapitre 5 « tests et conditions »**. Le chapitre 2 traite l'utilisation des boucles for exclusivement. Pour les boucles while, il faudra attendre le **chapitre 8 « répétitions conditionnées »**	
Notion de fonction	• Programmer des fonctions simples, ayant un petit nombre d'arguments.	Les fonctions arrivent tardivement dans la progression France-ioi. Il faut attendre le **chapitre 4 du niveau 2** pour aborder cette notion.	

4.1.2 Participants

L'expérimentation a été mise en oeuvre dans trois classes de secondes : deux classes du lycée Elie Cartan de La Tour-Du-Pin, et une au lycée Pierre du Terrail de Pontcharra. Les classes étant très chargées (36 élèves), nous avons mené les séances en demi groupe afin que chaque élève puisse s'impliquer individuellement.

4.1.3 Mise en oeuvre matérielle

Le dispositif choisi nécessite de travailler en salle informatique pour que chaque élève puisse bénéficier d'un poste de travail. Il est nécessaire que chaque poste puisse être connecté à Internet pour pouvoir se connecter sur l'URL du site http://www.france-ioi.org**.**

Par la suite les élèves peuvent continuer à se connecter à leur compte à partir de leur poste personnel.

4.1.4 Séquence de travail

Cette section présente la séquence que nous avons mise en place dans nos différentes classes.

La séquence se compose de 5 séances, chacune portant sur un thème du programme. Les objectifs sont multiples : Il s'agit d'abord de maîtriser les instructions dans un contexte très épuré, puis de monter en difficulté pour imbriquer toutes les connaissances acquises au cours de chaque séance pour résoudre les problèmes de type "**validation**", qui sanctionne une maîtrise satisfaisante de la notion présentée dans le chapitre.

Posture de l'enseignant : Durant cette séquence, nous allons tester plusieurs postures, allant d'un guidage relativement prononcé à une tentative d'autonomie complète, et ce graduellement sur les trois séances que nous souhaitons expérimenter. Lors de la première séance, nécessite un guidage était d'ordre pratique, afin de catalyser les procédures d'inscription et permettre une mise au travail la plus rapide possible. Dans un second temps, nous accompagnerons les élèves à prendre en main l'outil afin de résoudre les problèmes techniques liés au fonctionnement du site. L'activité de l'enseignant pourra être en soutien pour faciliter la mise en place de la boucle essai-erreur et sur l'utilisation de "l'interpréteur-évaluateur" en particulier sur la nature des différents messages d'erreurs. Lors de la troisième séance, nous veillerons à laisser les élèves en autonomie totale, en s'assurant simplement qu'ils ne se désengagent pas de la tâche.

Les objectifs de ces séances sont les suivants, et sont ceux des capacités attendues fournies par le programme :

[BoucleB] : Programmer une boucle bornée.

[BoucleNB] : Programmer une boucle non bornée.

[Type] : Choisir ou déterminer le type d'une variable.

[Aff] : Concevoir et écrire des affectations à des variables.

[Calc] : Ecrire une formule permettant un calcul combinant des variables.

[Cond] : Programmer une instruction conditionnelle

[Fonct] : Programmer des fonctions simples, ayant un petit nombre d'argument.

Le découpage de la séquence peut se faire comme suit. Les dates sont données à titre indicatif, chaque enseignant étant libre de les organiser comme il le souhaite dans le temps.

Séance	**Date**	**Durée**	**Contenu**	**Objectifs travaillés**
Découverte et prise en main de l'outil et du langage Python	Janvier	1h	- Inscription sur la plateforme - Niveau 1, chapitre 1 : Affichage du texte, suite d'instruction.	
Boucles bornées	Février/Mars	1h	- Niveau 1, chapitre 2 : Répétition d'instructions	[BoucleB]
Calcul et découverte des variables	Mars-Avril	1h	- Niveau 1, chapitre 3 : Calcul et découverte des variables	[BoucleB] [Affect] [Calc]
Instructions conditionnelles	Avril/Mai	1h	- Niveau 1, chapitre 5 : Instructions conditionnelles.	[Cond] [Aff] [BoucleB]
Répétitions conditionnées	Mai/Juin	1h	- Niveau 1, chapitre 5 : Répétitions conditionnées	[BoucleNB] [Type] [Aff] [Calc] [Cond]
Fonctions	Mai/Juin	1h	- Niveau 2, chapitre 4 : Fonctions	[Fonct] [Aff] [Calc]

4.2 Déroulement des séances

4.2.1 Séance 1 : Découverte et prise en main de l'outil et du langage Python

Durée : 1h

Objectifs de la séance :

- [Inscr] : Savoir s'inscrire sur un site internet et conserver ses identifiants et mots de passe
- [Affich] : Comprendre comment afficher un texte à l'écran en langage Python
- [Lire] : Savoir lire un énoncé complètement pour répondre à un problème posé
- [Instr] : Comprendre une instruction élémentaire documentée et l'utiliser correctement.

Déroulé :

- Arrivée en classe, connexion sur les postes, accès au site France-ioi (5 min)
- Inscription sur le site **France-IOI.org** : (10 min)

  Cette étape nécessite une vigilance accrue de l'enseignant. Il est conseillé d'éviter d'utiliser le compte Facebook/Google, qui amène des vérifications lourdes et chronophages, impliquant souvent que l'élève sorte son téléphone.

  L'élève doit veiller à renseigner son nom et son prénom dans sa fiche publique, afin de pouvoir s'inscrire dans le groupe classe créé par l'enseignant.

Déroulé des exercices dans l'ordre donné par **France-IOI** :

Exercice	Durée	Commentaires	Objectifs
(Découverte) Hello World	2 min	La principale difficulté rencontrée ici est de lire correctement l'énoncé, de comprendre ce qu'il y a à faire et où le faire.	[Affich] [Lire] [Instr]
(Cours) Représentation de la sortie d'un programme	2 min	Il s'agit d'une mise au point sur l'affichage de texte à l'écran.	[Affich] [Lire]
(Entraînement) Présentation	3 min	Il s'agit maintenant d'utiliser l'instruction d'affichage avec plus de texte à afficher	[Affich] [Lire]
(Cours) Afficher du texte : erreurs possibles	2 min	Le site insiste même sur les erreurs fréquentes que l'on peut faire. Les élèves les feront plus d'une fois malgré ce cours.	[Affich] [Lire]

(Découverte) Plan de la montagne	2 min	Plus beaucoup d'information, simplement un texte à réécrire.	[Affich] [Lire] [Instr]
(Validation) Dans le Fourré	7 min	Le problème majeur est de lire l'énoncé jusqu'au bout, et de s'y reporter autant de fois que nécessaire pour utiliser correctement les instructions	[Affich] [Lire] [Instr]

À l'issue de cette séance, nous avons demandé à nos élèves de commencer le chapitre 2 du niveau 1 à la maison, cette séance tombant avant les vacances d'hiver. La consigne était de faire les trois premiers exercices.

4.2.2 Séance 2 : Boucles bornées

Durée : 1h

Objectifs de la séance :

- [For] : Comprendre la syntaxe Python de la boucle "pour"
- [Lire] : Savoir lire un énoncé complètement pour répondre à un problème posé
- [Instr] : Comprendre une instruction élémentaire documentée et l'utiliser correctement.
- [Auto] : Être autonome dans la progression proposée
- [Corr] : Corriger un code faux ou incomplet

Consigne donnée aux élèves : Commencer au premier problème découverte. Le but de la séance est de faire autant d'exercices de type "**découverte**" et "**entraînement**" que nécessaire, pour pouvoir résoudre l'exercice de type "**validation**", qui est l'objectif final de la séance.
Les exercices proposés ici le sont à titre indicatif, les élèves ayant chacun un rythme différent.

Déroulé :

- Accès à sa session et connexion au site (5 min)

Exercice	**Durée**	**Commentaires/Difficultés rencontrées**	**Objectifs**
(Découverte) Punition	2 min	Découverte de la syntaxe de la boucle "for".	[For] [Lire] [Instr]
(Cours) Répétitions : erreurs	2 min	Présentation des erreurs fréquentes sur	[Corr]

possibles (Cours) Indentation : la touche tabulation.		l'utilisation de la syntaxe de la boucle for. Les élèves les feront quand même très souvent malgré tout.	[Auto]
(Entraînement) Mathématiques de base	5 min	À la lumière du cours précédent, les élèves doivent corriger un code pour le faire fonctionner.	[Lire] [Instr] [For] [Corr]
(Entraînement) Transport d'eau	8 min	La compréhension complète de l'énoncé pose problème et les instructions parfois sont mal comprises Exemple : haut(15) pour répéter 15 fois l'instruction haut().	[Lire] [Instr] [For] [Auto] [Corr]
(Découverte) Le secret du Goma	8 min	Similaire à l'exercice précédent, avec les mêmes difficultés.	[Lire] [Instr] [For] [Auto] [Corr]
(Cours) Cohérence de l'indentation	2 min	Le site met l'accent sur un problème récurrent : l'indentation des blocs de boucle. L'erreur est tellement récurrente qu'il est bon de faire un rapide point au tableau pour être certain que tous les élèves ont eu une explication à ce sujet.	[Lire] [For] [Auto]
(Entraînement) Sisyphe	8 min	Énoncé beaucoup moins guidé, avec deux boucles à la suite. Ceux qui sont arrivés là en général y arrivent.	[Lire] [Instr] [For] [Auto] [Corr]
(Validation) Vendanges	10 min	Les élèves comprennent que l'on peut imbriquer plusieurs boucles. Ceux qui n'ont pas encore compris sont invités à faire les exercices précédents.	[Lire] [Instr] [For] [Auto] [Corr]

De la même manière qu'après la première , nous avons proposé à nos élèves de poursuivre le travail chez eux.

4.2.3 Séance 3 : Calculs et découverte des variables

Durée : 1h

Objectifs de la séance :

- [Affect] : Comprendre le symbole d'affectation d'une valeur à une variable
- [Corr] : Être capable de débugger son programme, corriger un code faux ou incomplet.
- [Lire] : Savoir lire un énoncé complètement pour répondre à un problème posé
- [Instr] : Comprendre une instruction élémentaire documentée et l'utiliser correctement.
- [Auto] : Être autonome dans la progression proposée
- [Calc] : Réaliser un calcul simple mais long à l'aide de l'informatique
- [BB] : Réinvestir l'utilisation des boucles bornées

Consigne donnée aux élèves : Idem séance 2

Déroulé :

- Accès à sa session et connexion au site (5 min)

Exercice	Durée	Commentaires/Difficultés rencontrées	Objectifs
(Découverte) Réponds !	2 min	On affiche un nombre à l'écran, en tant que nombre et non en tant que chaîne de caractère.	[Instr] [Auto] [Lire]
(Découverte) L'éclipse	2 min	On calcule une durée. Python est alors utilisable comme une calculatrice. Certains élèves font le calcul à la main, puis affichent le résultat.	[Calc] [Lire] [Auto] [Corr]
(Entraînement) Bonbons pour tout le monde	5 min	Un calcul simple mais long à faire à la main est à effectuer. Beaucoup d'élèves font une erreur dans les priorités de calcul.	[Calc] [Lire] [Auto] [Corr]
(Découverte) L'algoréathlon	8 min	On suggère ici l'introduction d'une variable, qui contient une valeur	[Calc] [Lire]

		réutilisable. Les élèves qui lisent l'énoncé comprennent l'enjeu, les autres continuent de le faire "à la main" ou au moins sans utiliser de variables comme à l'exercice précédent.	[Auto] [Affect] [Corr]
(Cours) Nom d'une variable	8 min	Explication des noms de variables, qui ne peuvent pas être un mot clé du langage.	[Lire]
(Entraînement) Cours de récréation	2 min	Calcul du même acabit que l'exercice précédent. La différence de rapidité de résolution se creuse entre les élèves qui utilisent une variable et les autres. Leur introduction devient de plus en plus nécessaire.	[Calc] [Lire] [Auto] [Affect] [Corr] [Instr]
(Découverte) Une partie de cache-cache	8 min	On utilise ici les boucles bornées, du moins on espère que les élèves y penseront, bien qu'on ait pu observer des "copier/coller" qui laissaient apparaître le contraire. Une nouvelle chose également, le compteur de la boucle sous forme de variable.	[Calc] [Lire] [Auto] [Affect] [Corr] [Instr] [BB]
(Cours) Variables : supplément	5 min	Présentation des erreurs fréquentes liées à l'utilisation des variables.	[Lire]

5 Résultats

5.1 Prise en main

Lors de l'expérimentation, nous avons constaté que les élèves ont été désorientés par le premier contact avec la plateforme **France-IOI**. La plateforme n'est en effet pas destinée exclusivement à l'enseignement de la programmation et propose de nombreux liens à des fonctionnalités multiples : la navigation au sein du site pour atteindre les différentes parcours d'enseignement n'est pas triviale.

A cela s'ajoute la question de la création de compte personnel pour chacun des élèves de la classe. Le choix a été fait que chacun des élèves crée son compte afin de pouvoir continuer à l'utiliser tout au long de sa scolarité. **France-IOI** propose des exercices de différentes niveaux s'étendant au delà du lycée.

La connexion et la gestion de son identifiant personnel a été un problème pour certains élèves au cours des différentes séances : oubli de son identifiant ou de son mot de passe.

Malgré le fait que chacun de nos élèves possède un smartphone, nous avons noté que cela ne faisait pas d'eux des personnes habiles avec l'outil informatique, loin s'en faut. Certains ne connaissaient pas l'existence du presse-papiers et des commandes copier/coller au cours de ces séances. Or ces outils sont très utiles pour faciliter la résolution de certains exercices : copie des instructions de base proposées par les exemples du cours.

5.2 Obstacles didactiques

5.2.1 Les énoncés

Les énoncés des exercices sur **France-IOI** sont souvent longs, et contiennent beaucoup d'informations.

Ils sont structurés en deux parties : un mise contexte de la situation-problème et un énoncé du problème à résoudre. A chaque nouvelle étape, un nouvelle situation est proposée avec son contexte et son problème.

Les élèves doivent être concentrés et attentifs à toutes les informations fournies : elles sont toutes utiles et répondent à toutes les questions que les élèves se posent naïvement.

Nous avons constaté un défaut de lecture chez de nombreux élèves, cependant, une fois renvoyés à la lecture de l'énoncé réussissent correctement l'exercice.

Nous avons aussi remarqué que certains élèves étaient perturbés par certains énoncés. La nature descriptive et textuelle de certains énoncés demandent aux élèves de réaliser plusieurs transpositions : une modélisation mathématique, une traduction algorithmique puis une traduction en instructions informatiques. Sans étayage spécifique à propos de chacune de ses transpositions, certains élèves entrent dans une phase de résolution par tâtonnement en utilisant la boucle essai-erreurs. Les différentes natures d'obstacles didactiques peut conduire ces élèves dans une impasse si l'élève ne demande pas une aide.

Dans le problème décrit ci-après dont l'objectif est d'utiliser des variables dans des opérations mathématiques, l'élève s'est engagé dans la résolution sans avoir compris la nature mathématique du problème. Il a cependant utilisé correctement les variables nommés, c'est-à-dire qu'il a compris l'usage de ces dernières dans un programme. Cependant, la boucle essai-erreur ne lui permet pas d'identifier la nature de son erreur et son manque de modélisation mathématique.

Enoncé : *Niveau 1 - Calcul et découvertes des variables - 5) Cour de récréation*

"*La cour carrée a été mesurée avec quatre bâtons de longueurs respectives 17 m, 7 m, 5 m et 2 m. La longueur du côté de la cour est égale à 5 fois le premier bâton plus 2 fois le second plus 1 fois le troisième plus 2 fois le quatrième.*

Votre programme doit afficher deux lignes : la première doit contenir la surface de la cour, et la seconde ligne doit contenir son périmètre. Les résultats doivent être exprimés en mètres carrés et en mètres, respectivement, mais vous ne devez pas afficher l'unité après la valeur numérique."

5.2.2 La compréhension des erreurs (Feedback)

France-IOI permet aux élèves de soumettre leur programme pour en évaluer la validité.

Lorsqu'un élève soumet un programme, la plateforme valide la réponse ou lui renvoie une erreur.

Les erreurs sont de trois niveaux différents (*Annexe 3*).

1. **Erreur d'interprétation** : erreur de syntaxe, erreur d'indentation, variable non définies, etc…
2. **Erreur d'exécution** : fonction non définie, variable non définies, etc…
3. **Erreur d'évaluation** : Différence entre la sortie du programme et ce qui est attendu.

Nous avons noté que certains élèves ont rencontrés des difficultés à interpréter les erreurs qui leur ont été retournées. Dans la première séance, l'intervention de l'enseignant a été nécessaire pour permettre à certains élèves de comprendre la signification des erreurs d'interprétation et d'exécution. D'une part, la formulation en langue anglaise dans l'expression des erreurs d'interprétation et d'exécution a été une

difficulté pour certains. Ils n'ont pas su faire les liens avec les éléments de cours fourni en amont. D'autre part, l'entrée dans un processus de rédaction syntaxique rigoureux a été difficile pour d'autres.

Ensuite dans les séances suivantes, il a été nécessaire d'expliciter certaines erreurs d'évaluation. Ces dernières se présentent sous différentes formes (Annexe 3) en fonction de la nature des problèmes (textes, positions, numériques). Certains élèves n'ont pas été en mesure de comprendre et d'exploiter les informations fournies par le feedback et par conséquent ils ne pouvaient pas entrer dans une boucle essai-erreur pertinente.

Erreur retournée :

Test 1	Échec	Message de l'évaluateur : Nombre total de valeurs affichées : 2 Votre valeur numero 1 (784) est à une distance supérieure à 10000 de la re Erreur dans le résultat.	0%

5.2.3 Instructions versus fonctions

Le chapitre "Affichage de texte, suite d'instructions", a pour objectif d'introduire la notion de module ("*from robot import *"*) et l'import de fonctions. Nous avons remarqué que certains de nos élèves ont été bloqués. Les élèves ont interprété que l'instruction "haut()" était une fonction avec un paramètre, en se plaçant probablement dans le registre mathématique et non dans une registre informatique. Cela les a conduit à écrire le code suivant pour déplacer un robot vers le haut de 3 cases et à droite de 2 cases :

```
from robot import *
haut(3)
droite(2)
```

La difficulté des élèves est compréhensible dans la mesure où le dispositif n'a pas encore expliciter la définition d'un paramètre dans une fonction définie en Python. L'onglet "conseil" n'a pas répondu à cette nature d'obstacle. Nous avons dû expliciter cette notion en invitant les élèves à bien relire l'énoncé et la nature des instructions qui n'étaient pas définies avec un paramètre.

5.3 Point de vue pédagogique

5.3.1 Stratégies de résolution et comportements

Globalement, l'accueil des séances sur **France-IOI** fut bon dans l'ensemble mais très vite, beaucoup d'élèves ont été confrontés à des obstacles et des difficultés de natures différentes.

Elèves "experts": Les élèves qui d'habitude sont performants en mathématiques, dans le sens où ils se dirigent rapidement vers une résolution que l'on pourrait qualifier d'experte des problèmes proposés ont

cherché sur ces problèmes également des solutions optimales, parfois avec succès, parfois plus difficilement. En effet, certains sont restés bloqués longtemps devant certains exercices, cherchant à le résoudre en une seule soumission, sans proposer de solution partielle et bénéficier d'un retour de l'interpréteur amélioré que propose **France-IOI**. Un travail sur papier avant de programmer l'algorithme a été une stratégie mise en oeuvre avec succès.

Elèves inhabituellement motivés : Le changement de dispositif par rapport à un cours de mathématique classique a permis de motiver certains élèves, qui se sont investis dans la tâche avec un enthousiasme habituellement rare. Ceux-ci n'ont pas hésité à tester et re-tester leurs solutions, sans accorder d'importance au nombre de soumissions. Ces derniers ont su valider par étapes les différentes instructions pour converger vers une solution pertinente. Ces élèves ont été en général plus performants que les élèves dits "experts", renforçant au passage leur motivation, voyant que la "hiérarchie" des niveaux dans la classe était remise en question.

Différents comportements d'évitement de l'obstacle ont aussi été observés. En particulier, nous avons observé les comportements suivants :

- travaille seulement sous la contrainte, fait semblant de travailler, abandonne la tâche,
- demande de l'aide pour que l'on finisse à sa place,

Les mises en contexte, la boucle essai-erreur et les étayages proposés (conseils) n'ont pas permis à certains de sortir de cette stratégie d'évitement. L'intervention de l'enseignant a été nécessaire pour rompre avec ces comportements.

5.3.2 Utilisation de la boucle essai-erreur

La boucle essai-erreur est très rapide sur la plateforme **France-IOI.** Elle permet à l'élève de corriger instantanément et sans intervention du professeur (a priori) les erreurs éventuelles. Habituellement en classe mathématiques, la fréquence du feedback est beaucoup plus lente du fait des effectifs et la validation se fait obligatoirement soit par un pair, soit par le professeur. **France-IOI** ne permet pas actuellement de visualiser les différentes itérations de programmes produits par chaque élève. Cela nous aurait permis de conduire une analyse plus fine des obstacles franchis au cours de la boucle essai-erreur. De plus, nous ne pouvons pas déterminer à quel moment un élève demande un conseil à **France-IOI**. Ce qui nous aurait permis de valider la pertinence de l'étayage.

Par contre, nous avons recensé, sur les problèmes effectués par nos élèves le nombre d'essais moyens avant réussite dans le tableau suivant :

	Nombre de personnes	Nb essais	Moyenne C1	Nombre de personnes	Nb essais	Moyenne C2	Moyenne 2 classes
Hello World	35	59	1,686	34	73	2,147	**1,91**
Présentation	33	70	2,121	32	48	1,500	**1,82**
Plan de la montagne	30	36	1,200	33	44	1,333	**1,27**
Dans le fourré	37	62	1,676	35	83	2,371	**2,01**
Cylindres	31	150	4,839	29	304	10,483	**7,57**
Recette secrète	9	26	2,889	11	49	4,455	**3,75**
Punition	27	37	1,370	30	70	2,333	**1,88**
Mathématiques de base	29	40	1,379	25	39	1,560	**1,46**
Transport d'eau	27	125	4,630	25	135	5,400	**5,00**
Le secret du Goma	19	60	3,158	18	62	3,444	**3,30**
Sisyphe	15	26	1,733	11	25	2,273	**1,96**
Page d'écriture	7	21	3,000	6	26	4,333	**3,62**
Découverte jeu de dames	2	3	1,500	4	2	0,500	**0,83**
Mont Kailash	4	4	1,000	1	1	1,000	**1,00**
Vendanges	21	80	3,810	12	48	4,000	**3,88**
Réponds	29	37	1,276	26	29	1,115	**1,20**
L'éclipse	27	32	1,185	26	29	1,115	**1,15**
Bonbons pour tout le monde	28	66	2,357	26	97	3,731	**3,02**
L'algoreathlon	27	96	3,556	24	82	3,417	**3,49**
Cour de récréation	21	54	2,571	17	49	2,882	**2,71**
Une partie de cache cache	9	60	6,667	8	21	2,625	**4,76**

Nous n'avons pas mis en évidence de relation évidente entre la difficulté des problèmes et le nombre moyen de soumissions, hormis pour les exercices de type "Challenge" (comme le problème des tours de Hanoi, qui a compté en moyenne 7,57 essais avant d'arriver à une solution), qui arrivent après les exercices de validation et qui sont d'une difficulté plus prononcée. Le nombre moyen de soumissions sur l'ensemble des problèmes est de 2,74. Ce résultat est un peu biaisé dans le sens où les élèves, bien qu'en général à un par poste (et encore pas toujours) n'ont pas hésité à regarder l'écran des voisins, soumettant parfois leur solution en un nombre d'essais bien moindre à ce qu'ils auraient fait réellement seuls.

Même dans la troisième séance, nous avons été amené à intervenir pour débloquer certains élèves mais nous avons constaté qu'environ un tiers des élèves ne nous ont pas sollicités et ont avancé en autonomie, cela concerne en particulier les élèves **"expert"**.

Nous avons observé que les conseils n'étaient pas proposés à chaque situation-problème, notamment quand le problème pouvait se modéliser par une expression algébrique simple. Par contre, quand ils étaient proposés, la nature du conseil amenait à développer une pensée informatique : soit en proposant une approche algorithmique exprimée en langage naturel, soit en proposant de décomposer le problème en un problème plus simple, soit en proposant d'utiliser une partie d'algorithme déjà vue (reconnaissance de schéma : utilisation de boucle par exemple).

A l'issue de la boucle essai-erreur, nous avons noté que les élèves, malgré la consigne de consulter la correction, ne respectaient pas systématiquement cette étape de prise de recul par rapport au processus de soumission et de boucle essai-erreurs. D'une part, **France-IOI** n'impose pas cette étape dans les étapes de progression et d'autre part aucune trace n'est fournie aux enseignants leur permettant de vérifier si chaque élève a pris un temps pour vérifier si son programme est la solution optimale au problème. Enfin, les élèves avait la possibilité de consulter toutes les étapes de leur boucle essai-erreur mais aucun élève n'a pris l'initiative de revoir le processus itératif de résolution et les apprentissages qui y sont liés.

5.3.3 Autonomie et motivation

France-IOI nous paraissait présenter un réel intérêt pour stimuler la motivation de nos élèves notamment par la possibilité pour chacun d'eux de travailler à son rythme, de pouvoir prendre le temps nécessaire pour lire les énoncés et surtout les corrections pour identifier les bonnes stratégies de résolution.

Bien qu'après chaque séance, nous ayons incité nos élèves à continuer ce travail en autonomie complète, nous avons constaté que les élèves n'ont pas ou peu utilisé **France-IOI** chez eux (8 élèves sur l'ensemble des trois classes) . Ce constat nous amène à nous questionner sur l'effet motivationnel de ce dispositif sur les élèves. Nous observons que nos élèves restent sur motivation extrinsèque (liée à une évaluation future) et non sur une motivation intrinsèque (améliorer ses capacités à résoudre une problème algorithmique en Python).

De plus, nous avions envisagé que cette différentiation sur le temps puisse être mise à profit dans les temps de travail en autonomie par les élèves plus lents ou plus en difficulté. Mais ceux-ci n'ont pas utilisé ces temps pour tenter d'atteindre les objectifs de validation fixés pour chaque chapitre.

5.4 Évaluation formative des élèves

Du point de vue de l'enseignant, **France-IOI** offre une évaluation formative tout au long du parcours. Des étapes de **d'entraînement** et de **validation** sous forme de problème sont insérées dans chaque chapitre pour vérifier si l'élève a acquis les notions enseignées dans le chapitre considéré. Nous avons utilisé les indicateurs de progression pour mesurer l'avancement des apprentissage de chaque élève et vérifier les étapes validées. (Cf Annexe 4).

Du point de vue de l'élève, le dispositif lui permet de visualiser sa progression et récapitule l'ensemble des problèmes qu'il a résolu (Cf Annexe 4) . L'indicateur de progression (réussites en nombre de points) a motivé certains élèves à continuer l'activité pour améliorer ses performances. Certains l'ont utilisé pour comparer leurs performances avec leurs pairs.

En complément de cette évaluation et dans un souci d'automatisation de la pensée algorithmique et de son développement et afin de garder un contact avec celle-ci tout au long de l'année en dehors des séances spécifiques, nous suggérons de définir un recueil de questions flash faisant intervenir de l'algorithmique et le langage Python tout au long de l'année.

5.5 Évaluation sommative des élèves

Le temps nous a manqué pour mettre en place une évaluation des savoir-faire et savoirs acquis au cours des séances sur **France-IOI**. En revanche, nous pouvons proposer plusieurs pistes concrètes afin d'évaluer au mieux les élèves. Ces évaluations pourront prendre plusieurs formes :

- **TP noté en salle informatique** utilisant les structures de contrôle apprises et pouvant s'appliquer à de nombreuses parties du programme de mathématiques. On pensera notamment à des simulations dans la partie statistiques et probabilités, à l'approximation de la longueur d'une courbe, à l'encadrement d'une solution d'équation par l'algorithme de dichotomie, qui sont explicitement au programmes mais que **France-IOI** ne propose que dans des chapitres ultérieurs. L'un des obstacles que nous voyons à la mise en place d'un tel dispositif est la migration vers un environnement de développement autre que celui proposé par **France-IOI**, ce qui nécessite un temps de prise en main non négligeable.
- **Exercice ou partie d'un exercice dans un devoir surveillé**. On demande à l'élève de comprendre, de faire fonctionner ou de modifier un algorithme écrit en langage python afin de tester sa compréhension de la syntaxe en terme algorithmique. On peut également penser à la traduction d'un algorithme écrit en langage naturel en Python et réciproquement. Les algorithmes présentés seront moins difficiles et moins long qu'en TP, et auront un lien avec une étude mathématique préalable.
- **Programme ou TP à distance**, en devoir à la maison sur lequel les élèves travailleraient en autonomie complète. Cela nécessite de s'assurer que chaque élève a bien à disposition un ordinateur avec un éditeur **Python** accessible hors classe (un éditeur en ligne pourrait-être proposé).

6 Discussion et conclusion

6.1 Recontextualisation

Les objectifs de ce travail étaient de proposer un dispositif d'acquisition des bases du langage Python via la plateforme **France-IOI** à des classes de seconde. Nous voulions d'abord savoir si cet outil était adapté au programme de seconde. Ensuite, nous avons étudié le comportement des élèves en situation d'autonomie et enfin nous avons tenté d'évaluer la contribution de ce dispositif dans le développement d'une pensée informatique chez nos élèves. Nous avons conduit ce travail d'expérimentation avec un a priori positif, en supposant qu'en effet, **France-IOI** serait adapté à l'apprentissage du langage Python tout en stimulant l'autonomie des élèves.

6.2 Apports vis-à-vis des recherches antérieures

Depuis les années 1970, les réflexions autour de la structuration de la programmation et de son enseignement ont conduit à une approche didactique spécifique à la programmation distincte du champ didactique des mathématiques. Les recherches autour de la pensée informatique mettent en lumière certains principes fondateurs qu'il est pertinent d'apprendre assez tôt pour développer cette pensée informatique et sa déclinaison dans un langage de programmation. Cependant, ces enseignements sont conduits au sein de dispositif où l'enseignant est présent. Peu de recherches se sont spécifiquement intéressées à l'apprentissage en autonomie partielle ou totale. L'expérience de l'Ecole 42 manque encore de recul pour fournir des résultats étayés par la recherche.

Dans le cadre de notre expérimentation, nous avons cependant mis en évidence que l'apprentissage en autonomie d'un langage de programmation se heurte à un ensemble d'obstacles didactiques et pédagogiques qui ont été dans d'autres types d'expérimentation bien identifiés : en particulier les comportements et stratégie d'évitement ainsi que le manque de motivation dans le cadre d'un travail en autonomie hors du cadre scolaire. Ces résultats sembleraient invalider partiellement notre hypothèse de conduire un apprentissage en autonomie complète pour l'ensemble d'une classe dans le cadre de **France-IOI**. Le processus d'apprentissage en autonomie est une processus en tant que tel, notre expérimentation nous a conduit à mélanger deux processus : un processus d'apprentissage de la

programmation et un processus d'autonomisation. La difficulté méthodologique est de ne pouvoir analyser la production des élèves a posteriori.

Nous avons aussi analysé que la conception du parcours d'apprentissage de **France-IOI** respectait à la fois une approche incrémentale propre à introduire les différents concepts fondateurs de la programmation tout en stimulant une pensée informatique (algorithme, décomposition, reconnaissance de schémas). La variété des situations-problèmes permet aux élèves de se confronter plusieurs fois à des algorithmes ou partie d'algorithme et leur permet de percevoir une nouvelle manière de résoudre des problèmes de nature algorithmique. Nous avons observé qu'une phase préalable de cours ou de TP en mode débranchée (cas des classes du lycée Elie Cartan) semble induire une meilleure appropriation du parcours d'apprentissage en autonomie (réactivation) ce qui va dans le sens des recherches actuelles. Nous ne pouvons cependant pas valider cette hypothèse tant qu'une évaluation sommative ne sera pas conduite pour évaluer les compétences développées.

6.3 Limites et perspectives

Nous avons théoriquement validé que le dispositif permet de couvrir l'ensemble du programme de seconde mais notre expérimentation n'a pas permis de conduire toutes les séances nécessaires pour balayer l'ensemble des chapitres requis. Le temps et les moyens nécessaires nécessitent de démarrer ce type d'expérimentation plus tôt dans l'année scolaire.

D'une part, l'expérimentation ne permet pas d'invalider complètement la pertinence du dispositif autre que **France-IOI** dans un cadre de l'apprentissage en autonomie de la programmation pour l'ensemble d'une classe. Dans une certaine perspective, nous pouvons même envisager que **France-IOI** puisse prendre en compte certaines des observations réalisées afin d'améliorer le dispositif du point de vue didactique, pédagogique et ergonomique. Nous pensons en particulier à la nature des énoncés, à la pertinence des feedbacks et aux fonctionnalités permettant aux enseignants de consulter plus précisément la progression des élèves.

Cependant l'expérimentation sur **France-IOI** permet d'envisager des séquences d'algorithmique et de programmation alternant diverses modalités pédagogiques offrant un cadre d'apprentissage plus pertinent

pour certains élèves, ceux qui mettent en place plus facilement un processus d'apprentissage basée sur la boucle essai-erreur.
En complément, nous pouvons formuler certaines recommandations quant à la mise en oeuvre d'un tel dispositif. Dans un premier temps, la mise en place d'activités débranchées permettant aux élèves de réactiver une approche algorithmique de la résolution de problème. Ensuite nous recommandons d'initier la mise en oeuvre du dispositif assez tôt dans l'année afin de permettre aux élèves de s'approprier le dispositif (connexion, type d'énoncé, boucle essai-erreur) et de pouvoir développer leur autonomie tout au long de l'année. En complément, nous recommandons de réaliser des temps de synthèse avec les élèves sur les différentes stratégies et méthodes qu'ils ont mis en oeuvre pour résoudre les différentes situations-problèmes. Enfin, nous recommandons de réaliser des évaluations sommatives permettant de d'évaluer les apprentissages réalisés par les élèves et précisant aux élèves les capacités qu'ils doivent maîtrisées.

En tant qu'enseignants, nous avons beaucoup appris de cette expérience, bien qu'elle reste incomplète sur beaucoup d'aspects. Tout d'abord d'un point de vue didactique, nous avons pu observer les différentes stratégies des élèves face à des problèmes de nature différentes que ceux rencontrés dans la classe de mathématiques, et en particulier par rapport à l'autonomie. Cela nous a permis de mieux cerner leur fonctionnement face à un problème nouveau auquel ils font face seuls, ce qui pourra être pris en compte dans nos futurs travaux impliquant l'autonomie des élèves. Nous avons pu mieux distinguer les différentes pensées inhérentes à l'enseignement de l'algorithmique et de la programmation, le rôle de l'informatique débranchée et l'approche structurante de l'algorithmique au delà de l'activité de programmation.

D'un point de vue pédagogique, ce dispositif qui met au centre l'activité de l'élève nous a permis de nous questionner sur notre posture en tant qu'enseignant qui accompagne à l'autonomie : l'inconfort vécu à certains moments, se retenir de ne pas donner la réponse, ne pas prendre la souris dans la main !
Cette posture est délicate à mettre en place dans un cadre scolaire où nos élèves sont habitués à être "encadrés".

Bibliographie

[Académie des sciences, 2013] Avis de l'Académie des sciences (2013), L'enseignement de l'informatique en France - Il est urgent de ne plus attendre - www.academie-sciences.fr/pdf/rapport/rads_0513.pdf

[Astolfi, 1992] Astolfi, J. -P. (1992). L'école pour apprendre. Paris : ESF.

[Astolfi, 2014] Astolfi, J. -P. (2014). L'erreur, un outil pour enseigner. Paris : ESF.

[Balacheff, 1994] Balacheff N. (1994) - Didactique et intelligence artificielle -Recherches en Didactique des Mathématiques, La Pensée Sauvage, 1994, 14, pp.9-42.

[Baron et Bruillard, 2011] Baron G., et Bruillard E. (2011) - L'informatique et son enseignement dans l'enseignement scolaire général français : enjeux de pouvoir et de savoirs, Joël Lebeaume éd., Recherches et expertises pour l'enseignement scientifique. Technologie - Sciences – Mathématiques. De Boeck Supérieur, 2011, pp. 79-90.

[Baudé, 2014] Baudé, J. (2014) - L'expérience des « 58 lycées » - 1024 – Bulletin de la société informatique de France – numéro 4, octobre 2014

[Briant, 2013] Briant N. (2013) - Étude didactique de la reprise de l'algèbre par l'introduction de l'algorithmique au niveau de la classe de seconde du lycée français. (Thèse de doctorat, université Montpellier 2) 2013.

[Briant et Bronner, 2015] Briant N. – Bronner A. (2015) - Étude d'une transposition didactique de l'algorithmique au Lycée : Une pensée algorithmique comme un versant de la pensée mathématique - Colloque Espace Mathématique Français 2015.

[Dowek, 2011] Dowek G. (2011) "Les quatre concepts de l'informatique" - Didapro 4 - Dida & STIC - Actes du colloque International, 24-26 octobre 2011, Université de Patras.

[Drot-Delange et Tort, 2018] Drot-Delange, B. , Tort,F (2018). Résolution de défis et pensée informatique. 10èmes rencontres scientifiques de l'ARDIST, Mar 2018, Saint-Malo, France.

[Eduscol, 2016] Eduscol (2016) - Algorithmique et programmation - Cycle 4

[Favre, 1995] Favre, D. (1995, avril-mai-juin). Conception de l'erreur et rupture épistémologique. Revue Française de Pédagogie n°111, pp. 85-94.

[Frisona et al, 2018] Frisona, P. & Daoudb, M. & Adamb M. (2018) - Transition didactique de l'activité débranchée à la programmation avec AlgoTouch

[Haspekian et Nijimbéré, 2016] Haspekian M. et Nijimbéré C. (2016) - Favoriser l'enseignement de l'algorithmique en mathématiques : une question de distance aux mathématiques ?, Éducation et didactique, 10-3 | 2016, 121-135.

[Holec, 1991] Holec, H (1991) - Autonomie de l'apprenant : de l'enseignement à l'apprentissage, Education permanente. N° 107 - 1991

[Lui, 2015] Lui, T. (2015). L'école 42: la liberté au coeur de l'apprentissage?, Didapro6-Didactique de l'informatique et des STIC-Quelles éducations au numérique en classe et pour la vie? 5-27 janv. 2016 Namur (Belgique).

[Meirieu, 1987] Meirieu, P. (1987). Apprendre…, oui mais comment ? Paris : ESF.

[Modeste, 2012] Modeste S. (2012) Enseigner l'algorithme pour quoi? Quelles nouvelles questions pour les mathématiques? Quels apports pour l'apprentissage de la preuve?, Histoire et perspectives sur les mathématiques [math.HO]. Université de Grenoble, 2012.

[Modeste, 2012 fev] Modeste S. (2012) - La pensée algorithmique : Apports d'un point de vue extérieur aux mathématiques, actes du colloque EMF 2012.

[Pélisset, 1985] Pélisset, E. (1985) - Pour une histoire de l'informatique dans l'enseignement français. Premiers jalons. « Système éducatif et révolution informatique », Les cahiers de la FEN, collection Recherches.

[Raab, 2014] Raab, R. (2014) - Apprentissage en autonomie et stratégies d'évitement de l'obstacle, Questions Vives, n° 22 | 2014

[Tchounikine, 2017] Tchounikine P. (2017) - Initier les élèves à la pensée informatique et à la programmation avec Scratch. http://lig-membres.imag.fr/tchounikine/PenseeInformatiqueEcole.html

Sitographie

Georges-Louis BARON, “ Cinquante ans de Technologies de l’information et de la communication (TIC) en éducation” - http://ritpu.org/pages/entrevues/

Jean-Pierre Archambault - "C’est un changement de paradigme que l’informatique devienne une discipline scolaire"
https://www.letudiant.fr/educpros/entretiens/jean-pierre-archambault-president-d-enseignement-public-et-informatique-cest-un-changement.html

Wing J. (2006) - La pensée informatique - https://interstices.info/la-pensee-informatique/

Wing J. (2010) - Computational Thinking-What and Why?
https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Association EPI - 21 octobre 2002 - L'ENSEIGNEMENT DE L'INFORMATIQUE À L'ÉCOLE SECONDAIRE (extrait) - https://edutice.archives-ouvertes.fr/edutice-00276158/file/h70ocde.htm

BO spécial n°8 du 13 octobre 2011 - Enseignement de spécialité d’informatique et science du numérique de la série scientifique - classe terminale
https://www.education.gouv.fr/pid25535/bulletin_officiel.html?cid_bo=57572

Le projet **csunplugged** traduit en français - http://interstices.info

Débuter avec le langage python - Ressources et tutoriels
https://ent2d.ac-bordeaux.fr/disciplines/mathematiques/des-tutoriels-pour-debuter-avec-le-langage-python/

Débuter avec le langage python - Ressources et tutoriels
https://www.pedagogie.ac-nantes.fr/mathematiques/mutualisation/logiciels/python-1049620.kjsp?RH=MATH

Annexes

Table des annexes :

- **Annexe 1** : Extraits des programmes de 2nde générale et technologique (BO n°30 du 23 juillet 2009).
- **Annexe 2** : Cours et Problèmes proposés au Niveau 1.
- **Annexe 3** : Liste des erreurs d'exécution.
- **Annexe 4** : Indicateurs de progression

Annexe 1 :

Extraits des programmes de 2nde générale et technologique (BO n°30 du 23 juillet 2009)

Les capacités attendues dans le domaine de l'algorithmique d'une part et du raisonnement d'autre part, sont transversales et doivent être développées à l'intérieur de chacune des trois parties. […]
La démarche algorithmique est, depuis les origines, une composante essentielle de l'activité mathématique. Au collège, les élèves ont rencontré des algorithmes (algorithmes opératoires, algorithme des différences, algorithme d'Euclide, algorithmes de construction en géométrie). Ce qui est proposé dans le programme est une formalisation en langage naturel propre à donner lieu à traduction sur une calculatrice ou à l'aide d'un logiciel. Il s'agit de familiariser les élèves avec les grands principes d'organisation d'un algorithme : gestion des entrées-sorties, affectation d'une valeur et mise en forme d'un calcul.

Dans le cadre de cette activité algorithmique, les élèves sont entraînés :

- à décrire certains algorithmes en langage naturel ou dans un langage symbolique ;
- à en réaliser quelques-uns à l'aide d'un tableur ou d'un petit programme réalisé sur une calculatrice ou avec un logiciel adapté ;
- à interpréter des algorithmes plus complexes.

Aucun langage, aucun logiciel n'est imposé.

L'algorithmique a une place naturelle dans tous les champs des mathématiques et les problèmes posés doivent être en relation avec les autres parties du programme (fonctions, géométrie, statistiques et probabilité, logique) mais aussi avec les autres disciplines ou la vie courante.

À l'occasion de l'écriture d'algorithmes et de petits programmes, il convient de donner aux élèves de bonnes habitudes de rigueur et de les entraîner aux pratiques systématiques de vérification et de contrôle.

Instructions élémentaires (affectation, calcul, entrée, sortie).
Les élèves, dans le cadre d'une résolution de problèmes, doivent être capables :

- d'écrire une formule permettant un calcul ;
- d'écrire un programme calculant et donnant la valeur d'une fonction ; ainsi que les instructions d'entrées et sorties nécessaires au traitement.

Boucle et itérateur, instruction conditionnelle
Les élèves, dans le cadre d'une résolution de problèmes, doivent être capables :

- de programmer un calcul itératif, le nombre d'itérations étant donné ;
- de programmer une instruction conditionnelle, un calcul itératif, avec une fin de boucle conditionnelle.

Annexe 2 : Cours et Problèmes proposés au Niveau 1

France-IOI

ıes

Cours et problèmes

Voir les cours et résoudre les problèmes en :

C | C++ | Pascal | OCaml | Java | JavaScool | Python

Parcours général | Parcours lycée | Méthodes | Problèmes non classés

Le parcours général permet de progresser à son rythme, en partant des bases de la programmation et en allant jusqu'à un niveau avancé en algorithmique.

Niveau 1

1 – Affichage de texte, suite d'instructions	**6** problèmes	**6** résolus
2 – Répétitions d'instructions	**10** problèmes	**10** résolus
3 – Calculs et découverte des variables	**13** problèmes	**13** résolus
4 – Lecture de l'entrée	**10** problèmes	**7** résolus
5 – Tests et conditions	**8** problèmes	**3** résolus
6 – Structures avancées	**8** problèmes	
7 – Conditions avancées, opérateurs booléens	**10** problèmes	
8 – Répétitions conditionnées	**5** problèmes	

Annexe 3 : Liste des erreurs d'exécution

1) Erreurs d'interprétation :

Erreur de compilation :

```
File "exe", line 1
    print(Hello world!)
                     ^
SyntaxError: invalid syntax
```

Erreur de compilation :

```
File "exe", line 1
    print("Hello world)
                      ^
SyntaxError: EOL while scanning string literal
```

Erreur de compilation :

```
Sorry: IndentationError: expected an indented block (exe, line 3)
```

2) Erreurs d'exécution :

Test 1	Erreur	Erreur d'exécution. Voici ce qui a été affiché :
		`Traceback (most recent call last):`
		`  line 1`
		`  print(Hello)`
		`NameError: name 'Hello' is not defined`
			0%

Test 1	Erreur	Erreur d'exécution. Voici ce qui a été affiché :
		`Traceback (most recent call last):`
		`  line 3`
		`  ramasser()`
		`NameError: name 'ramasser' is not defined`
			0%

3) Erreurs d'évaluation:

Test 1	Échec	La réponse donnée par votre programme est incorrecte. Il a affiché :
		`Hello world`
		au lieu de :
		`Hello world!`
		Surligner le premier caractère différent
			0%

Test 1	Échec	Message de l'évaluateur :
		`Le robot n'a pas accompli sa mission`
		`Legende :`
		`- '.' : case libre`
		`- 'H' : position de la hotte`
		`- 'C' : position de la charette`
		`- 'R' : position actuelle du robot`
		`R.............C`
		`Le robot ne porte pas la hotte`
		`Le robot a deja fait 19 allers et 19 retours`
		Erreur dans le résultat.
			0%

Annexe 4 : Indicateurs de progression

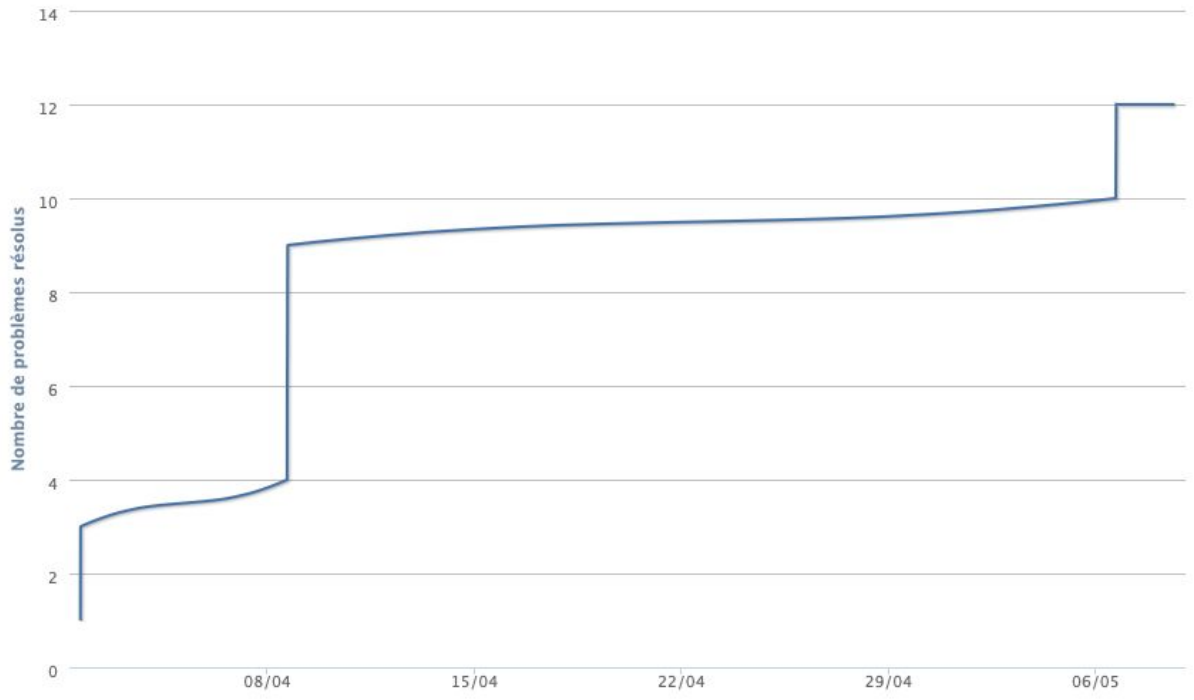
Progression des membres

Cliquez sur une ligne pour obtenir des détails.

Fida [redacted]	Emmanuelle [redacted]	Marie [redacted]	kenan [redacted]
1 essai — 4 min	1 essai — 1 min	5 essais — 3 min	2 ess
1 essai — 3 min	2 essais — 2 min	2 essais — 5 min	3 ess
1 essai — 1 min	1 essai — 1 min	1 essai — 1 min	1 es
2 essais — 3 min	1 essai — 2 min	1 essai — 4 min	1 es
6 essais — 34 j 22 h 59 min	14 essais — 18 min	10 essais — 21 min	0 essai –
Non commencé	12 essais — 9 min	0 essai — 34 j 22 h 59 min	Non

Fida [redacted]	Emmanuelle [redacted]	Marie [redacted]	kenan [redacted]
1 essai — 4 min	1 essai — 2 min	1 essai — 13 min	1 es
1 essai — 2 min	1 essai — 5 min	1 essai — 1 min	1 es
3 essais — 8 min	4 essais — 9 min	18 essais — 42 min	0 essai –
13 essais — 12 min	0 essai — 34 j 22 h 59 min	1 essai — 25 j 2 h 33 min	0 essai –
2 essais — 5 min	0 essai — 34 j 22 h 59 min	1 essai — 6 min	0 essai –
3 essais — 34 j 22 h 59 min	0 essai — 34 j 22 h 59 min	0 essai — 34 j 22 h 59 min	0 essai –
Non commencé	0 essai — 34 j 22 h 59 min	Non commencé	0 essai –
Non commencé	0 essai — 34 j 22 h 59 min	Non commencé	1 ess
Non commencé	14 essais — 21 min	2 essais — 6 min	3 ess
Non commencé	Non commencé	Non commencé	Non

Courbe de progression



Année universitaire 2017-2018

DU *Métiers de l'enseignement, de l'éducation et de la formation*

Mention Second degré

Parcours : Mathématiques

Titre de l'écrit scientifique réflexif : Apprendre le langage Python en autonomie en seconde

Auteur : Thomas RENAUD, Yann MAMODE

Résumé : Actuellement, l'algorithmique et la programmation sont de nouveau au coeur des programmes de mathématiques de l'enseignement secondaire. Ces enseignements posent la question d'un développement d'une pensée algorithmique et informatique chez nos élèves. Par ailleurs, de nombreuses études sur l'autonomie scolaire nous ont poussées à nous interroger sur la faisabilité de l'apprentissage du langage Python en autonomie en classe de seconde. Par le biais de la plateforme France-IOI, nous avons mis en oeuvre un tel dispositif, afin d'étudier les postures et les comportements aussi bien des élèves que de l'enseignant. Nous en avons analysé les points positifs et les éventuels écueils, tant d'un point de vue didactique que pédagogique. Dans le même temps, nous avons porté un regard critique sur l'utilisation de la plateforme France-IOI afin de nous permettre d'en connaître les forces et les faiblesses, dans le but d'en faire une utilisation la plus pertinente possible dans les années à venir.

Mots clés : Algorithmique, Programmation, Python, Enseignement secondaire, Autonomie.

Summary :

Currently, algorithmics and programming are once again at the heart of highschool math programs. These teachings raise the question of developing an computer thinking for our students. Besides, many studies on school autonomy have led us to question the feasibility of learning Python language autonomy in highschool classes. Through France-IOI platform, we have implemented such a device to study the postures and behaviors of both students and teachers. We have analyzed the positive points and possible pitfalls, both from a didactic and pedagogical point of view. At the same time, we took a critical look at the use of the France-IOI platform to allow us to know its strengths and weaknesses, with the aim of making the most relevant use possible in the years to come.

Key words : Algorithmics, Programming, Python, Highschool, Autonomy.