Ecole Nationale Supérieure de Formation de l'Enseignement Agricole

Master 2

« Métiers de l'Enseignement, de l'Education et de la Formation »

Master MEEF Enseignant du Second Degré

Mémoire

Enseigner la programmation en langage Python

Alicia LAVERGNE

Jury :

Cécile GARDIES,
Professeure de Sciences de l'information et de la communication, ENSFEA
Directrice de la recherche

Sylvie SOGNOS,
Docteure en sciences de l'information et de la communication, ENSFEA
Co-directrice de mémoire

Laurent FAURE
Maître de conférences en science de l'éducation, ENSFEA
Examinateur

Mai 2020

Table des matières

Cadre théorique

Remerciements

Tout d'abord, je tiens à remercier Cécile GARDIES, Professeure de Sciences de l'information et de la communication, ainsi que Sylvie SOGNOS, Responsable de la bibliothèque, Ingénieure d'études en information-documentation. Elles m'ont, en tant que directrice et co-directrice de mon mémoire, aidé et donné de précieux conseils qui m'ont permis de le réaliser.

Je remercie également l'ensemble du personnel du LEGTA de l'Oisellerie dans lequel j'ai réalisé mon année de stage, pour son soutien et son aide précieuse. Mais cette recherche n'aurait pas été possible sans ma conseillère pédagogique, Muriel TINKA-LAVILLE, et sans l'ensemble des enseignants de mathématiques et de SNT qui m'ont donné leur avis. La participation de mes élèves à cette recherche a été très bénéfique et je tiens donc à les remercier eux aussi.

Pour finir, je remercie ma famille, mes amis et plus particulièrement mes parents qui m'ont toujours soutenu dans mes études et encore plus cette année de stage.

Introduction

J'ai réalisé ce mémoire dans le cadre de la deuxième année de Master « Métier de l'Enseignement, de l'Education et de la Formation » à l'Ecole Nationale Supérieure de Formation de l'Enseignement Agricole (ENSFEA) au cours de l'année scolaire 2019-2020. L'enseignement a toujours été mon objectif. En effet, depuis toute jeune, je me destine à exercer ce métier qui me passionne et les mathématiques ont toujours été une matière que j'adore. C'est donc tout naturellement que j'ai passé, en 2019, le CAPESA de mathématiques.

La programmation est une branche des mathématiques que j'affectionne tout particulièrement mais je me suis posée de nombreuses questions concernant son enseignement qui est présent depuis peu dans les programmes scolaires. Pensant que je n'étais pas la seule dans ce cas, j'ai décidé de réaliser mon mémoire sur l'enseignement de la programmation en langage Python dans le but d'apporter des réponses à ces différentes questions qui pourront, je l'espère, aider d'autres enseignants. Pour l'instant, peu de recherches ont été menées dans ce sens et je voulais donc, à travers ce mémoire, y contribuer à mon échelle.

La société ne cesse d'utiliser et de développer l'informatique ainsi que les automates et cela va continuer dans les prochaines années. Ces derniers ont besoins d'être programmés pour fonctionner et donc nous avons besoin de programmeurs. C'est pourquoi, il est important d'apporter des notions de programmation aux élèves dans leur scolarité et vont eux-mêmes faire évoluer ces différents outils. La programmation peut également être utilisée, durant la scolarité, comme un outil pour réaliser de nombreuses choses et pas seulement en mathématiques. Elle est utilisée dans de nombreux domaines et peut servir aux élèves dans leurs études supérieures voir même dans leur métier futur.

Cependant, j'ai pu constater que cet enseignement n'est vraiment pas évident puisqu'il fait intervenir des outils et des connaissances nouvelles. Je me suis donc questionnée sur ce que les élèves pouvaient ressentir face à cet enseignement, sur ce qui pouvait être mis en place pour

faciliter leurs apprentissages ainsi que ce que les enseignants utiliser pour que les élèves réussissent à atteindre l'objectif final qui est de réaliser des programmes dans le langage Python.

Nous allons donc nous demander, comment un enseignant peut-il enseigner le langage de programmation Python dans le cadre de ses cours de mathématiques ? De cette question générale, en découle de nombreuses concernant les méthodes qui peuvent être utilisées, les logiciels utilisés, de la programmabilité des savoirs ou encore de la motivation des élèves.

Pour répondre à ces différentes questions de recherches, nous allons, dans un premier temps, apporter des éléments théoriques au sujet de la motivation, de la didactique de l'informatique ainsi que du sens des apprentissages.

Dans la seconde partie de ce mémoire, le cadre méthodologique, nous présenterons et analyserons, à l'aide des éléments théoriques apportés, des données qui vont être recueillies auprès d'élèves et d'enseignants. Cela nous permettra ensuite, de répondre aux questions de recherches et donc d'apporter des éléments sur l'enseignement de la programmation en langage Python.

Problématique

De nos jours, l'informatique, au sens large du terme, est utilisée partout autour de nous. D'après le rapport de l'Unesco, « *Une part de plus en plus importante de l'information produite aujourd'hui dans presque tous les domaines de l'activité humaine est numérique et conçue pour être accessible sur ordinateur* » (Unescopresse, 2002). C'est pourquoi, l'informatique est de plus en plus présente dans les nouveaux programmes du lycée et est rentrée en vigueur depuis septembre 2019 avec l'apparition du nouveau module Sciences Numériques et Technologies (SNT) en classe de seconde. Cela est renforcé par la mise en valeur de la programmation dans le référentiel de seconde et de la spécialité mathématiques des classes de premières et terminales générales. Comme Anne Cordier le dit, les élèves vivent dans l'ère du numérique et utilisent tous, plus ou moins, l'informatique dans leur vie de tous les jours et depuis leur enfance. Ils utilisent l'informatique aussi bien chez eux qu'à l'Ecole, que ce soit pour faire une recherche sur internet, pour rédiger un traitement de texte, un diaporama ou même pour faire des jeux sur PC mais il est beaucoup plus rare que les élèves fassent de la programmation. C'est donc l'Ecole qui va leur permettre d'apprendre à programmer. Avec la réforme du collège, on a vu apparaître la programmation en Scratch mais au lycée, un langage spécifique de programmation est apparu récemment et pour la première fois dans les référentiels du lycée : le langage Python.

Python est un langage qui offre une multitude de fonctionnalités. Au lycée, les premières notions de Python vont être enseignées ce qui va permettre par la suite de produire des programmes plus complexes. Il va donc être très utile pour la poursuite d'études de certains élèves et éventuellement dans leur métier futur. En effet, la programmation est très utilisée pour la conception de téléphones, d'ordinateurs et tous les appareils électroniques.

Cependant, Python n'est pas un langage intuitif et peut être compliqué pour un débutant en programmation ou pour quelqu'un qui ne connaît pas le langage. Il demande donc un certain nombre de savoirs que les élèves vont devoir acquérir, inscrits dans les programmes. Les enseignants vont devoir maîtriser ce savoir pour pouvoir le transmettre aux élèves.

De plus, les élèves apprécient souvent cela puisque c'est l'occasion d'aller en salle informatique, ils peuvent tester eux-mêmes les programmes et obtiennent un résultat à la fin. Cela permet également de varier les activités, il peut donc s'agir d'une source de motivation pour les élèves. Il y a le côté ludique de l'informatique qui peut intéresser les élèves, c'est un

domaine qu'ils connaissent relativement bien puisqu'ils utilisent quotidiennement le numérique, ils se sentent donc plus à l'aise et voient plus facilement l'utilité de cet enseignement.

Or, la plupart des enseignants n'ont jamais suivi de formation pour pouvoir enseigner la programmation en Python et même, pour certains, n'ont jamais programmé dans ce langage. Les enseignants ne savent donc pas comment l'enseigner puisqu'eux-mêmes ne le maîtrisent pas forcément. Ils ont souvent peur que cela soit trop chronophage et ne savent pas comment l'incorporer à leur cours. Il est souvent difficile de l'évaluer comme le dit Fluckiger : *« c'est une chose de savoir faire quelque chose, c'en ai une autre que de comprendre sa description dans un référentiel de compétence suffisamment finement pour l'évaluer chez autrui, et encore une autre que de construire une situation d'évaluation de cette compétence. »* (Fluckiger, 2019)

L'algorithmique est une part importante des mathématiques puisqu'il permet de travailler la logique. Le fait de programmer en Python permet aux élèves de tester leurs algorithmes et de voir si ceux-ci renvoient ou non un résultat et si oui, si c'est bien ce que l'on voulait obtenir. Dans ce cas on va pouvoir le modifier pour obtenir le résultat voulu. On ne se serait pas forcément rendu compte de notre erreur sans le programmer sur ordinateur.

Cela nous amène à nous poser un certain nombre de questions. Tout d'abord, il faut se demander quels sont les savoirs à mobiliser pour programmer en Python ? Ensuite, une autre question qui vient se poser relève de la manière d'enseigner le langage Python et des méthodes à utiliser. C'est pourquoi on peut se demander quelle programmabilité des savoirs enseigner sur la programmation ? Quels logiciels utiliser ? Est-ce qu'il faut procéder par étapes en utilisant d'abord des langages plus abordables avant de passer à celui-là ou est-ce que l'on commence directement avec Python ? Comment l'évaluer ? Et qu'est ce qui pourrait être mis en place pour aider les enseignants qui doivent enseigner ce langage ? La question de la motivation va également pouvoir être posée, les élèves sont-ils motivés par cela ou faut-il trouver des moyens pour qu'ils soient motivés par cet enseignement ?

Cadre théorique

Afin de réaliser une recherche en didactique de l'informatique, nous allons nous intéresser à la question de la motivation des élèves qui constituera le premier chapitre de cette partie théorique. Le second chapitre présentera une approche de la didactique de l'informatique et plus particulièrement de la programmation ainsi que la transposition didactique. Cela nous amènera ensuite, au dernier chapitre, qui concernera le sens des apprentissages.

I- La motivation scolaire

La motivation est une question qui se pose rapidement dans le domaine de la didactique. L'informatique, et plus particulièrement la programmation, peut être une source de motivation pour certains élèves mais d'autres vont avoir besoin d'être motivés. Nous allons donc étudier la motivation scolaire dans sa généralité qui peut ensuite s'appliquer au domaine de la programmation.

a. La motivation : qu'est-ce que c'est ?

D'après le CNRTL[1], la motivation est l'« *ensemble des facteurs dynamiques qui orientent l'action d'un individu vers un but donné, qui déterminent sa conduite et provoquent chez lui un comportement donné ou modifient le schéma de son comportement présent.* ». Les définitions de la motivation font référence aux notions d'énergie et de force. En effet, si l'on reprend la définition donnée par Tardif : « *La motivation scolaire est essentiellement définie comme l'engagement, la participation et la persistance de l'élève dans une tâche* » (Tardif, 1992, p.91), on y retrouve bien ces deux notions. Ou encore, Bandura donne une autre définition de la motivation scolaire qui fait également référence à cela : « *la motivation en contexte scolaire est un état dynamique qui a ses origines dans les perceptions qu'un élève a de lui-même et de son environnement et qui l'incite à choisir une activité, à s'y engager et à persévérer dans son accomplissement afin d'atteindre un but* » (Bandura, 1986, p.7). Nous voyons dans cette

1 Centre National de Ressources Textuelles et Lexicales

définition apparaître le « but » qui a une grande importance dans la motivation, nous reviendrons sur cela plus tard dans ce mémoire. Si l'on résume ces différentes définitions, la motivation scolaire est l'ensemble des processus qui permettent à un élève de s'engager dans une tâche ou un travail afin d'atteindre un but.

Nous avons donc vu ici différentes définitions de la motivation scolaire, nous allons maintenant nous intéresser à son importance dans les apprentissages.

b. L'importance de la motivation dans les apprentissages

La motivation des élèves joue un rôle important dans leurs apprentissages. Un lien étroit existe entre la motivation et la réussite scolaire. En effet, plus un élève va être motivé par ce qu'il apprend, plus il va réussir à l'école. C'est elle qui va permettre d'atteindre nos objectifs et même de se surpasser. Ce concept est travaillé dans la théorie de la motivation humaine. D'après Vianin, « *Ce qui est particulier à la motivation humaine, c'est que, une fois le but atteint, l'individu a tendance à vouloir se dépasser encore et fixer un nouveau but qui va au-delà du but atteint.* ». (Vianin, 2006, p.95)

Le mot « motivation » vient du latin *movere* qui signifie se déplacer ce qui renvoie à la mise en mouvement. La motivation va donc permettre à l'élève de se mettre en mouvement et va lui permettre de rendre possible tout apprentissage.

Maintenant que nous avons vu à quoi sert la motivation dans les apprentissages, nous allons aborder les différentes composantes de la motivation.

c. Les composantes de la motivation

1) La pyramide de Maslow et le rôle des besoins

Dans sa pyramide (présentée dans la figure 1), Maslow propose les différents besoins d'une personne en allant des besoins physiologiques jusqu'à la réalisation de soi. Pour lui, pour pouvoir accéder à un besoin, il faut avoir satisfait les besoins des niveaux inférieurs ce qui veut dire que pour satisfaire le besoin de réalisation de soi, il faut que tous les autres soient satisfaits.

La motivation d'une personne ou plus particulièrement d'un élève vient du dernier étage de la pyramide de Maslow, c'est-à-dire du besoin de réalisation de soi. La motivation serait donc

suscitée par le désir de satisfaire ce besoin. Cet étage concerne également les apprentissages qui sont en lien direct avec la motivation comme nous avons pu le voir dans la partie précédente. D'après Vianin, « *la motivation naîtrait […] d'un désir non assouvi présentant quelque obstacle à son assouvissement.* » (Vianin, 2006, p.29). Cette pyramide nous permet donc de visualiser l'ensemble des conditions nécessaires à la motivation des élèves.

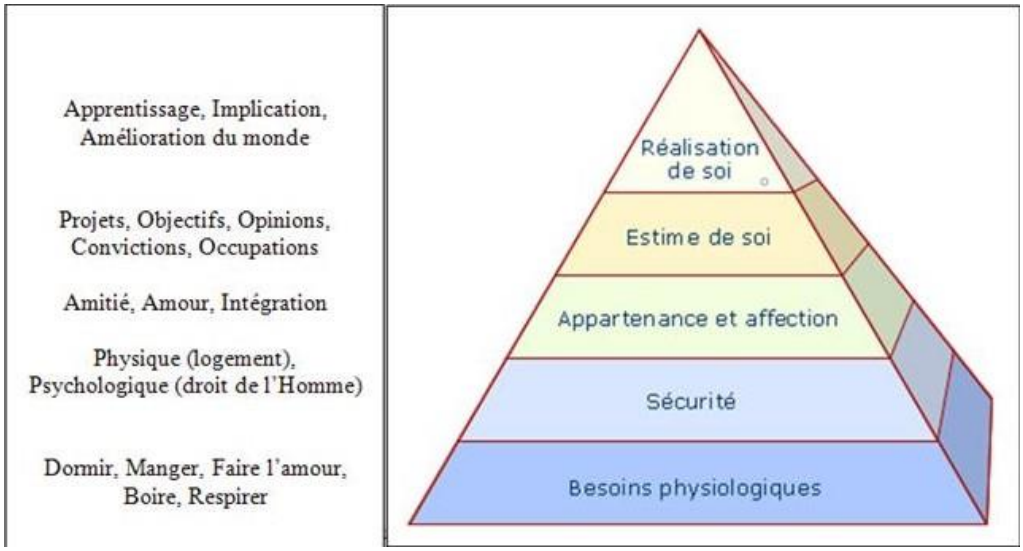

Figure 1 : Pyramide de Maslow

2) La motivation intrinsèque et la motivation extrinsèque

La motivation peut être **intrinsèque** ou bien **extrinsèque**.

La **motivation intrinsèque** peut être définie comme « *les forces qui incitent à effectuer des activités volontairement, par intérêt pour elles-mêmes et pour le plaisir et la satisfaction que l'on en retire* » (Roussel, 2000, p.7). L'élève va donc être motivé par l'activité en elle-même et va prendre du plaisir à la réaliser. Par exemple, un élève qui aime l'informatique et plus particulièrement la programmation va être motivé lorsque l'enseignant lui demandera de créer un programme. La motivation intrinsèque peut également être renforcée si l'enseignant confie une partie des initiatives aux élèves.

On parlera de **motivation extrinsèque** si l'individu ou l'élève est motivé par un facteur extérieur à la tâche à accomplir. Il sera donc motivé par une éventuelle récompense, une note, pour tirer un avantage de ce travail, pour obtenir de l'approbation d'un tiers ou encore pour éviter une punition.

Lorsque l'élève est motivé extrinsèquement, on parlera de **motivation positive** s'il cherche à atteindre son but en obtenant une satisfaction. L'élève a souvent un espoir de réussite. En revanche, la **motivation négative** procède, quant à elle, par la peur. L'élève va éviter un comportement négatif tel qu'une punition ou une mauvaise note. Elle va alors s'exprimer par la crainte de l'échec.

On a pu constater que les résultats sont meilleurs lorsque les élèves sont motivés intrinsèquement plutôt qu'extrinsèquement.

3) Niveau d'aspiration et niveau d'expectation

On va parler dans cette partie des niveaux d'aspiration et d'expectation.

Le **niveau d'aspiration** va représenter le but que l'individu va se fixer et le niveau qu'il souhaite atteindre lorsqu'il devra accomplir un travail. Cela correspond à un but idéal. Le **niveau d'expectation** est le niveau que l'élève peut espérer atteindre, il s'agit donc d'un niveau plus réaliste. Le niveau d'aspiration est souvent trop éloigné de ce que l'élève connaît, il vaut mieux que le but soit plus proche, les résultats seront alors meilleurs. En effet, si le but est trop éloigné, l'élève ne verra pas les progrès qu'il aura fait et pourrait se décourager. Vygotsky parle de « zone proximale de développement » :

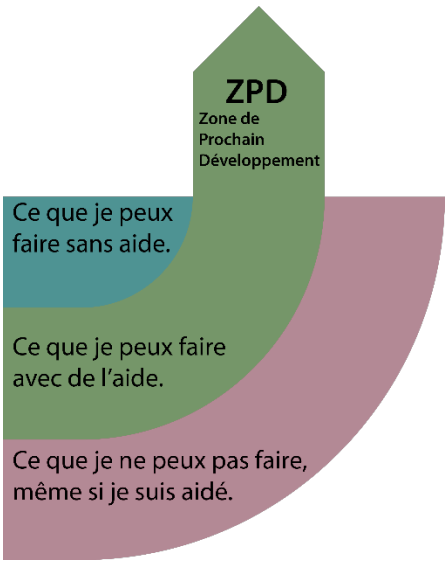

Figure 2 : Zone proximale de développement

Vygotsky définit trois zones, la première est ce que l'élève peut faire par lui-même et sans aide, la seconde est la « zone proximale de développement », il s'agit de ce que l'élève peut faire

mais, cette fois-ci avec l'aide de quelqu'un. Et la dernière est celle où l'élève ne peut pas faire la tâche demandée même avec de l'aide. Dans la première zone, l'élève ne va pas forcément être motivé puisqu'il n'y a pas de but à atteindre, il sait déjà réaliser la tâche. Dans la dernière zone, l'élève ne va pas être motivé non plus puisqu'il n'est pas capable de réaliser le travail, il va donc rapidement abandonner. Pour que les élèves soient motivés par une tâche, il va falloir que celle-ci se situe dans la zone proximale de développement. En effet, dans cette zone, les élèves vont avoir un but à atteindre, ils ne savent pas trouver par eux-mêmes le résultat mais avec de l'aide ils vont pouvoir réussir. Comme le dit Vianin, *« pour l'enseignant, le choix de la tâche proposée à l'enfant est par conséquent fondamentale pour susciter sa motivation et pour engager l'enfant dans ses apprentissages favorisant son développement.* » (Vianin, 2006, p.34). C'est donc à l'enseignant de proposer une tâche adaptée et qui se situe dans cette zone proximale de développement.

4) Motivation par les finalités et motivation par les moyens

Il existe deux types de motivation : la motivation par les finalités et la motivation par les moyens.

Un élève **motivé par les finalités** va concentrer son attention sur le but qu'il doit atteindre mais il risque de ne pas se donner les moyens d'y parvenir. En revanche, un élève **motivé par les moyens** « *risque de se perdre dans la maîtrise technique que requiert l'activité et de ne plus apercevoir le but que ces moyens sont sensés lui permettre d'atteindre.* » (Vianin, 2006, p.35).

On va pouvoir, par exemple, monter des projets concrets avec des élèves. En effet, cela permet de leur donner un but à atteindre (motivation par les finalités) ainsi que des moyens pour y parvenir. Nous reviendrons sur cela un peu plus tard dans ce mémoire.

Comme le dit Vianin, *« l'enseignant devra, idéalement permettre à l'élève de comprendre à la fois les objectifs proximaux de la tâche, leur inscription dans des buts distaux – leur donnant tout leur sens – et les moyens permettant de les atteindre.* » (Vianin, 2006, p.37)

5) Attributions causales et contrôlabilité

Les attributions causales sont les causes qui sont invoquées par les élèves lorsqu'ils sont confrontés à un échec ou un succès.

Il existe deux types de causes : la cause **interne** qui est liée à la personne en elle-même, à l'intelligence, aux capacités, au travail personnel, etc. Dans ce cas, l'élève a réussi car il a, par exemple, des facilités dans la matière ou parce qu'il a beaucoup travaillé. La seconde est la cause **externe** qui est liée à une source extérieure. Dans ce cas, l'élève n'aura pas réussi parce que l'épreuve était trop compliquée ou à cause de la qualité de l'enseignant ou au contraire, il aura réussi car l'épreuve était trop facile.

On va également pouvoir distinguer la cause « modifiable » de la cause « stable ». La cause **modifiable** est due par exemple au manque d'efforts que l'élève a fourni mais, la fois suivante, il pourra faire mieux en travaillant davantage. En revanche, la cause **stable** est, quant à elle due au fait que l'élève est « nul » ou « pas intelligent ». Dans ce cas, même si l'élève fait des efforts pour réussir, il n'y arrivera quand même pas. Un élève qui se sent « nul » ne va pas être motivé dans son travail et ne va pas vouloir fournir d'efforts puisque de toute façon « ça ne sert à rien, j'y arrive pas », dans tous les cas, il ne réussit pas. Cela va donc être lié à l'estime que l'élève a de lui-même. Si l'enseignant donne des stratégies d'apprentissage à un élève, cela peut aider à le remotiver.

La **contrôlabilité** est le sentiment que va avoir l'élève de pouvoir ou non contrôler la situation. Si l'élève constate que les résultats qu'il obtient sont incontrôlables quoi qu'il fasse, il va perdre sa motivation et va rester passif dans les apprentissages.

Viau résume tout cela dans le tableau suivant :

Tableau 1 : Attribution causale (Viau, 1997, p.67)

	Interne		**Externe**	
	Stable	**Modifiables**	**Stable**	**Modifiable**
Contrôlable	Stratégies d'apprentissages	Effort	Programme scolaire	Perceptions de l'enseignant
Incontrôlable	Aptitudes intellectuelles	Maladie	Niveau de difficulté d'une activité	Humeur de l'enseignant

6) Motivation et estime de soi

L'estime que l'élève a de lui-même va jouer un rôle important sur sa motivation. **L'estime de soi** est le jugement de valeur que l'on fait sur notre image. Cela peut être ce que l'on croit sur soi-même.

Les apprentissages et la motivation se font selon l'estime de soi. Le rapport entre les aspirations et les réussites va avoir une conséquence sur l'estime de soi et donc sur les apprentissages et surtout sur la motivation des élèves. Si un élève ne réussit pas à réaliser une tâche, cela risque de faire baisser l'estime qu'il a de lui-même. S'il est courant pour lui qu'il ne réussisse par dans ses apprentissages, il va se décourager et perdre sa motivation.

Si un élève a une bonne estime de lui-même, s'il réussit, il va penser que c'est grâce à ses capacités et/ou à son intelligence. Mais s'il échoue, ce sera la faute d'une cause externe comme nous l'avons vu dans la partie précédente. En revanche, cela va être le contraire pour les élèves qui n'ont pas une bonne estime d'eux. S'ils échouent, ce sera leur faute puisqu'ils ne sont « pas bons » et s'ils réussissent, ce n'est pas grâce à eux. Un élève qui a une bonne estime de lui sera souvent plus motivé que les autres. En effet, si l'élève ne sent pas capable de faire quelque chose, il ne va pas être motivé pour réaliser cette tâche.

Lorsqu'un enseignant se retrouve face à un élève qui n'est pas motivé car il succède les échecs, il va pouvoir aider l'élève à se remotiver en ayant une meilleure estime de lui. Il peut en effet le valoriser lorsque qu'il réussit à faire quelque chose pour lui montrer qu'il est capable de réussir. Si l'élève arrive à remonter son estime de soi, il va également augmenter sa motivation. Vianin donne un moyen de remonter cela : « *Si l'enseignant permet à l'élève de choisir l'activité d'apprentissage en s'assurant que ce dernier se sent capable de réussir et qu'il a les moyens de contrôler la situation, sa motivation sera très forte.* » (Vianin, 2006, p.46)

Dans cette partie, nous avons vu les différentes composantes de la motivation et les différentes causes qui font qu'un élève est motivé ou non. Nous allons voir dans la prochaine partie différentes théories explicatives ainsi que des moyens de remotiver des élèves qui ne le sont pas.

d. Les théories explicatives

Nous allons voir dans cette partie trois théories explicatives de la motivation scolaire : la théorie béhavioriste, l'approche humaniste et la psychologie cognitive.

1) La motivation dans la théorie béhavioriste

La théorie béhavioriste que nous allons étudier maintenant s'appuie sur le besoin de satisfaction de la pyramide de Maslow.

D'après Houssaye, « *le comportement des individus est modelé par les récompenses (ou leur absence) et les punitions (ou leurs absence) qui en découlent ; il peut ainsi être renforcé positivement ou négativement* » (Houssaye, 1993, p.224). Cette théorie s'appuie sur l'importance des récompenses et des punitions dans le domaine de la motivation.

Différentes études ont montré que les individus et plus particulièrement les élèves réussissent mieux lorsqu'une récompense est accordée. On pourra alors parler de motivation extrinsèque puisque les élèves vont être motivés par la récompense et non par la tâche à réaliser en elle-même. Si l'enseignant donne une punition dans le cas où le travail qu'il a demandé n'est pas correctement réalisé, les élèves peuvent également être motivés puisqu'ils auront dans ce cas la crainte de la punition. Cependant, les récompenses apportent de meilleurs résultats que les sanctions ou les punitions. Les notes sont importantes pour motiver les élèves. Pour beaucoup d'entre eux, s'il n'y a pas de note à la fin d'un travail, ils ne vont pas se donner à fond pour le réussir alors que s'ils en ont, ils vont tout faire pour avoir la meilleure possible. La note est donc vue comme une récompense à un travail si elle est bonne ou elle peut être considérée comme une punition pour les élèves qui ont des difficultés et qui obtiennent des « mauvaises » notes. D'après Viau, elle peut être une source de motivation pour certains élèves ou peut également les démotiver s'ils n'obtiennent pas les notes qu'ils considèrent comme bonnes. Pour que cela soit constructif et formateur, il faut que l'enseignant donne des explications plus précises qu'une note pour que les élèves puissent progresser.

Un « *facteur important pour la motivation est la réussite immédiate. La joie d'avoir triomphé d'une tâche et de posséder une certaine récompense suscite une espèce d'autorenforcement.* » (Minsel et Wimmer, 1990, p.141). Lorsqu'un élève réussit une tâche, si l'enseignant le félicite

et l'encourage, cela peut permettre de le remotiver et de lui redonner confiance en lui comme nous l'avons vu précédemment.

Lorsqu'un enseignant donne un travail à réaliser à sa classe, si l'activité proposée est attrayante ou ludique, les élèves vont avoir tendance à être plus motivés pour la faire. Cela peut être une tâche nouvelle, ou encore sous forme de défi. Le matériel mis à la disposition des élèves peut également permettre de rendre la tâche plus attrayante. Par exemple, pour la programmation, les élèves vont utiliser un ordinateur (ou une calculatrice) qui est un outil qu'ils apprécient particulièrement et qu'ils utilisent quotidiennement chez eux pour différentes raisons. Si l'enseignant leur donne également un défi, cela va permettre de rendre les élèves acteurs du travail et vont s'investir dedans.

Une autre approche de cette théorie est la progression des élèves par paliers. A chaque étape, les élèves doivent connaître précisément les objectifs pour savoir où ils vont arriver et quelle est la finalité du travail. Ces objectifs doivent être assez difficiles pour susciter la motivation mais pas trop pour qu'ils se sentent capables de les atteindre avec leurs capacités. Il faut donc trouver le juste milieu (zone proximale de développement).

Certains auteurs critiquent cette théorie, en effet, certains pensent que si les élèves ne sont motivés uniquement par la récompense, cela ne va pas être bénéfique dans leurs apprentissages. D'après Fenouillet, *« la régulation externe est la motivation extrinsèque la moins autodéterminée. Dans ce cas, l'individu agit uniquement pour obtenir une récompense ou pour éviter quelque chose de désagréable telle qu'une punition. L'élève qui travaille son cours uniquement pour éviter d'avoir une mauvaise note reflète exactement ce type de motivation*. » (Fenouillet, 2003, p.100). De plus, comme le dit Covington (2000), les notes ne motivent pas en priorité ceux qui en ont le plus besoin. Elles vont motiver davantage les élèves qui réussissent (qui sont souvent déjà motivés) que les élèves qui ont des difficultés et qui ont le plus besoin d'être motivés. Ces derniers vont plus avoir tendance à être démotivés si leurs notes sont relativement basses, ils vont les percevoir comme une menace. La note doit donc être utilisée comme une récompense et non pas comme une sanction. Vianin nous dit que : *« Les évaluations formatives et formatrices aident l'enfant à s'auto-évaluer et le rendent ainsi acteur de ses apprentissages. Le statut de l'erreur change également : alors que l'élève qui obtient une mauvaise note est mis en échec dans une démarche sommative, son droit à l'erreur est reconnu dans une démarche formative.* » (Vianin, 2006, p.61).

Dans la théorie béhavioriste, pour motiver les élèves, il est important de décrire les objectifs du cours et de le découper de manière progressive. Il faut également énoncer la récompense que l'élève va obtenir avant de réaliser l'activité pour qu'il sache à quoi s'attende. Il vaut mieux éviter les punitions et les sanctions qui peuvent au contraire démotiver les élèves s'ils en ont trop souvent. Il est important que l'enseignant fasse un retour aux élèves sur ce qu'ils connaissent ou non et les différentes capacités qu'ils ont acquises.

On peut retenir dans cette partie que les récompenses peuvent permettre à certains élèves d'être motivés. Les notes sont également motivantes principalement pour les élèves qui réussissent mais pour ceux qui sont en difficulté, cela peut au contraire les décourager. Les évaluations formatives peuvent permettre de remotiver les élèves en difficulté, ils vont pouvoir identifier leurs erreurs sans être pénalisés par une note. Cette approche peut être intéressante pour remotiver un élève qui a une motivation très basse mais elle ne peut pas être instaurée à long terme et il va falloir trouver d'autres moyens pour motiver un élève intrinsèquement.

2) La motivation dans l'approche humaniste

Dans l'approche humaniste, on considère que les enseignants ainsi que le ressenti des élèves jouent un rôle important dans le développement de la motivation et dans les apprentissages des élèves.

Rogers (1984) distingue trois qualités que doit avoir l'enseignant dans cette approche : la **congruence**, l'**acceptation inconditionnelle** et l'**empathie.** La congruence consiste à développer une réelle relation avec les élèves. L'enseignant va rester lui-même face à eux. L'acceptation inconditionnelle va permettre aux élèves d'être reconnus par l'enseignant comme des êtres de confiance. L'enseignant doit en effet faire confiance à ses élèves. L'empathie est « *la capacité de se mettre à la place de l'autre, de le comprendre et de lui communiquer cette compréhension* ». (Vianin, 2006, p.66)

La relation entre un enseignant et ses élèves est importante dans la motivation des élèves comme le dit André, « *c'est la relation humaine authentique qui est la source de la motivation* » (André, 1992, p.16). Si la classe se fait dans une ambiance positive, détendue et un environnement favorable, les élèves vont être davantage motivés.

Les élèves sont plus motivés lorsque les apprentissages viennent d'eux-mêmes. En effet, « *les élèves sont toujours motivés pour trouver des réponses aux problèmes qu'ils se posent, ils ne*

sont pas toujours motivés à solutionner des problèmes qu'on leur pose. » (Nachon et Wallian, 2005, p.22). L'enseignant pourra donc impliquer les élèves dans le choix des activités, des supports, ce qui pourra permettre aux élèves d'augmenter leur motivation intrinsèque.

Lorsqu'un élève accomplit une tâche, il voudrait que ceux qu'il estime soient fiers de lui. Il va donc souvent travailler dans ce but. Il peut s'agir des parents, de la famille de l'élève ou encore d'un professeur. L'enseignant lui-même et son attitude envers l'élève ainsi que le regard que lui porte l'enseignant va permettre à l'élève d'être motivé. Lacroix pense que l'enseignant doit s'inscrire « *en tant que personne active, épanouie dans son être autant que dans sa fonction et dans le monde. Ce faisant, il transmet bien plus qu'un savoir sur une matière spécialisée ; il transmet de la confiance, de l'espoir dans l'avenir, du bonheur à être adulte.* » (Lacroix, 2002, p.139). On pourra parler de l'effet Pygmalion, lorsque l'enseignant est convaincu qu'un élève va réussir, ce dernier va avoir tendance à mieux y arriver. L'enseignant va donc permettre aux élèves de s'améliorer.

Dans cette théorie, on va parler de deux types d'enseignants : les enseignants informants et les enseignants contrôlants. Les **enseignants informants** laissent le choix aux élèves, les informent des objectifs et du but des apprentissages. **L'enseignant contrôlant**, quant à lui, ne laisse que très peu de choix aux élèves. Ils vont utiliser davantage de punitions et de récompenses. Des études qui ont été menées ont montré que les élèves sont davantage motivés intrinsèquement avec des enseignants informants plutôt que contrôlants. Les élèves vont souvent préférer lorsque les enseignants leur laissent de l'autonomie dans leur travail, ils vont se sentir plus compétents et vont s'impliquer davantage dans la tâche à accomplir.

La motivation des enseignants va également se ressentir dans les apprentissages et dans la motivation de leurs élèves. Si l'enseignant est motivé lui-même par son cours, qu'il aime ce qu'il fait et qu'il le montre à ses élèves, cela va leur permettre d'être plus motivés. Comme le dit Viau, « *Comment peut-on accuser les élèves de ne pas être motivés à apprendre, lorsqu'ils se trouvent devant un enseignant qui bâille, ne rit jamais, regarde constamment sa montre, ne veut pas être dérangé, leur donne des exercices pour ne pas avoir à leur parler, etc.* » (Viau, 1997, p.120). On peut donc dire que plus les enseignants vont être motivés dans leur propre cours, plus les élèves vont l'être aussi.

Dans cette théorie, on nous propose plusieurs pistes qui peuvent permettre aux élèves d'être motivés. La première est de donner du sens au apprentissages, nous reviendrons sur cela dans

le dernier chapitre de la partie théorique. Une autre méthode est d'utiliser le jeu, Sestier et Hochet pensent que *« par sa proximité avec le monde de l'enfance, par sa complicité, par la rupture qu'il entraîne avec le quotidien […], le jeu est un formidable outil de mobilisation des élèves.* » (Sestier et Hochet, 2005, p.38). Les stimulations sociales vont aussi pouvoir motiver les élèves. On en distingue deux types : la **compétition** et la **coopération**. On va plutôt favoriser la coopération par son contexte davantage positif que pour la compétition. On va pouvoir faire coopérer les élèves en les faisant travailler par groupes. Les élèves vont pouvoir ainsi faire preuve d'entraide et de solidarité. Chacun va pouvoir s'améliorer grâce à la coopération, chaque élève a des choses à apporter aux autres puisqu'ils n'ont pas forcément acquis les mêmes compétences en même temps. L'élève qui va être plus à l'aise dans un domaine va pouvoir aider et expliquer aux autres la notion. Certains élèves vont mieux comprendre une notion si elle est expliquée par l'un de ses camarades. Le fait de réaliser des projets avec les élèves peut susciter leur motivation. En effet, ils vont avoir une certaine autonomie dans la réalisation de celui-ci, cela peut partir de ce qu'ils aiment faire ou d'une question qu'ils se posent. Les élèves vont donc s'investir dans le projet puisque l'idée vient d'eux. Ils s'investissent davantage lorsque le sujet les intéresse.

Pour motiver les élèves dans cette théorie, on va donc favoriser un climat de classe positif avec de l'échange entre les élèves et les enseignants ainsi que les élèves entre eux. On va mettre l'accent sur les relations et sur l'autonomie. On va laisser le choix aux élèves et on va les laisser faire par eux-mêmes.

3) La motivation dans la psychologie cognitive

Nous allons étudier dans cette partie, la dernière théorie abordée dans ce mémoire. Il s'agit de la motivation dans la psychologie cognitive qui a une approche différente des deux premières dont nous avons parlé.

D'après Tardif (1992, ch.II), il existe cinq ensembles de facteurs qui composent la motivation scolaire : la perception de l'élève des buts poursuivis par l'école, la conception de l'élève de ce qu'est l'intelligence, la perception de la valeur de la tâche à effectuer, la perception des exigences de la tâches et la perception de la contrôlabilité de la tâche. Il rappelle également que ce qui joue le plus dans la motivation scolaire, ce ne sont pas les capacités réelles de l'élève mais cela va être la perception qu'il a lui-même de ses capacités.

Dans cette théorie, il est question de donner du sens aux apprentissages. Nous verrons cela dans le troisième chapitre du cadre théorique.

Ici, pour susciter la motivation des élèves, *« l'enseignant veillera donc à donner à ses élèves un maximum d'informations, à leur offrir des choix, à leur faire éprouver l'utilité de ce qu'il enseigne, à les aider à se fixer des buts et des objectifs, à leur proposer des activités signifiantes et à les encourager.* » (Archambault et Chouinard, 1996).

Dans cette approche, on va dire que plus l'élève connaît les objectifs du cours, plus il sait à quoi va servir la tâche et d'où elle vient, plus il connaît les différentes stratégies efficaces, plus il pense maîtriser et contrôler la tâche et plus il va être motivé.

En conclusion, pour motiver les élèves dans la psychologie cognitive, il faut définir correctement les buts des apprentissages, informer les élèves des exigences de la tâche, les aider à comprendre à quoi vont leur servir les différentes connaissances que l'on va pouvoir leur apporter et leur fournir des stratégies efficaces pour les aider à progresser.

Dans cette partie, nous avons présenté trois théories sur la motivation scolaire bien différentes. La première repose sur l'importance de la récompense et de la note dans les apprentissages, la seconde sur l'importance du relationnel entre les élèves et les enseignants ainsi que sur le climat de la classe et la dernière sur le rôle de la connaissance des différents objectifs.

La motivation scolaire est donc une nécessité dans les apprentissages. Certains élèves vont déjà être motivés et d'autres vont avoir besoin d'aide pour l'être. Nous avons donc vu dans ce chapitre différentes définitions de la motivation scolaire, à quoi elle sert, ainsi que des façons de motiver les élèves. En informatique et plus particulièrement en programmation, les élèves vont être motivés par l'usage de différents supports et vont voir de nouvelles notions. Certains vont avoir besoin d'être motivés à l'aide par exemple de projets.

Nous allons parler dans le prochain chapitre de la didactique de l'informatique et plus particulièrement de la programmation.

II- Didactique de l'informatique

La programmation et son enseignement étant le sujet principal de ce mémoire, nous définissons, dans la première partie de ce chapitre la notion d'informatique scolaire et plus particulièrement la programmation ainsi que la didactique. Nous verrons ensuite l'état de l'art de la didactique de l'informatique. Pour finir, nous étudierons le concept général de transposition didactique.

1) Définitions

Nous utilisons un certain nombre de termes que nous allons définir dans cette partie. Nous commençons par donner une définition de la notion d'informatique, puis de programmation et d'algorithmique, de langage Python qui est le langage qui nous sert d'appui dans cette recherche et nous finirons par définir la notion de didactique.

a) Qu'est-ce que l'informatique ?

L'informatique est une science qui permet de traiter des informations en exécutant des *programmes* (ce terme sera défini dans la partie suivante) sur des machines qui peuvent être des systèmes embarqués, des robots ou encore des ordinateurs qui sont plus utilisés en milieu scolaire.
L'informatique est une science jeune puisqu'elle commence avant la Seconde Guerre Mondiale grâce au mathématicien Alan Turing avec la machine universelle de Turing qui a beaucoup évoluée depuis pour devenir les ordinateurs que nous connaissons aujourd'hui.
Elle n'a jamais été considérée comme une matière à part entière dans l'enseignement mais elle est présente dans l'ensemble des disciplines et tout particulièrement dans les mathématiques avec notamment la programmation que nous allons définir juste après. Elle est tout de même de plus en plus présente dans les programmes du lycée avec l'apparition de la nouvelle discipline Sciences Numériques et Technologie (SNT).
Dans les années 1980, l'informatique était principalement utilisée pour des calculs scientifiques, pour gérer des entreprises ou encore dans les télécommunications. Aujourd'hui, l'informatique concerne beaucoup plus de domaines. Elle est notamment utilisée dans la communication, pour les loisirs, dans l'industrie, dans la conception d'objets ainsi que pour le commerce. L'informatique est également très utilisée dans les domaines des sciences humaines,

de la santé, et des sciences, en particulier en mathématiques. En effet, l'informatique et les mathématiques sont très liés puisque l'informatique utilise de nombreuses notions mathématiques comme la logique, la combinatoire ou encore la statistique. De même, les mathématiques utilisent l'informatique comme un outil de visualisation, cela peut également permettre de réaliser des calculs compliqués à faire à la main. C'est pourquoi la programmation est inscrite dans le programme de mathématiques.

D'après le rapport de l'académie des sciences de mai 2013, *« Nombre des progrès technologiques les plus marquantes de ces dernières années sont des produits directs de l'informatique* » et « *L'informatique est d'une importance toujours grandissante en termes de création de richesses et d'emplois dans le monde ».* En effet, l'informatique a de plus en plus d'impacts sur notre société et il est donc primordial de commencer à enseigner cela dès l'école primaire. Plus les élèves vont être formés dans ce domaine et mieux ils pourront s'adapter au monde de demain.

L'informatique scolaire joue un rôle important dans la société puisqu'il s'agit de « *l'ensemble des manifestations des technologies informatiques ou de la science informatique dans le champ scolaire. »* (Fluckiger, p.17, 2019).

Cependant, l'enseignement de l'informatique n'est pas forcément évident. Nous proposons d'étudier cela dans la suite de ce mémoire.

b) Qu'est-ce que la programmation et l'algorithmique ?

Avant de nous intéresser à la façon d'enseigner l'informatique et plus particulièrement la programmation, nous définissons les notions d'algorithmique et de programmation. Pour pouvoir définir ces deux notions, nous avons besoin de définir deux autres termes : les **opérations** et les **instructions.**

Les **opérations** sont des processus qui visent à obtenir un résultat à partir d'un ou de plusieurs objets. Une **instruction** est une série d'actions contenant des mots clés et des connecteurs logiques.

Un **algorithme** est une suite finie d'opérations et d'instructions permettant de répondre à un problème. Les premiers algorithmes connus sont très anciens puisque le mot « algorithme » vient du nom du mathématicien Al-Kwarrizmi au IX^{ème} siècle. Un algorithme peut être par exemple une recette de cuisine, la résolution d'un Rubik's Cube ou peut être utilisé en mathématiques pour trouver par exemple le PGCD de deux nombres avec l'algorithme d'Euclide.

La **programmation** consiste à écrire un « programme informatique » dans un langage de programmation particulier. Un programme est alors un algorithme traduit dans un langage afin d'être exécuté par un ordinateur.

c) Le langage Python

Un langage de programmation est une notation utilisée pour écrire des algorithmes et des programmes. Il en existe une multitude qui sont plus ou moins faciles et intuitifs à utiliser. Au collège, depuis quelques années, le langage utilisé est le langage Scratch qui est assez intuitif pour débuter en programmation et qui permet de comprendre la logique de l'algorithmique. Au lycée, jusqu'à maintenant, aucun langage n'était réellement écrit dans les programmes, les enseignants étaient libres de faire seulement des algorithmes sur papier, ou alors de choisir un langage eux-mêmes. La plupart des enseignant utilisaient le langage prédéfini dans la calculatrice utilisée en classe (généralement CASIO ou TexasInstruments) ou certains utilisaient le logiciel Algobox.

Dans les programmes du lycée rentrés en vigueur en septembre 2019, un langage de programmation spécifique doit être utilisé par les enseignants en classe : le langage **Python**. Ce langage offre de nombreuses possibilités et est très utilisé dans les études supérieures. Le logiciel le plus utilisé dans les établissements pour pouvoir programmer dans ce langage est le logiciel EduPython. (Voir figure 3). Ce logiciel est divisé en deux parties : la partie du haut permet d'écrire les programmes et les différentes fonctions et celle du bas (la console) permet de les exécuter.

Figure 3 : Logiciel EduPython

d) Didactique

Avant de pouvoir nous intéresser à la didactique de l'informatique, nous avons besoin de définir le terme « didactique ».

La didactique est un domaine des sciences qui étudie les méthodes d'enseignement et la diffusion des savoirs. Reuter donne la définition suivante : les didactiques sont des « *disciplines de recherche qui analysent les contenus (savoirs, savoir-faire, …) en tant qu'ils sont objets d'enseignement et d'apprentissages, référés/référables à des matières scolaires […] Plus précisément encore, les didacticiens sont des spécialistes de disciplines scolaires (français, mathématiques, sciences, musique…)* » (Reuter, 2007/2013, p.65). Brousseau donne une autre définition : *« La didactique comme science, étudie la diffusion des connaissances utiles aux hommes vivant en société. Elle s'intéresse à la production, à la diffusion et à l'apprentissage des connaissances ainsi qu'aux institutions et aux activités qui les facilitent. »* (Brousseau, 2004).

Maintenant que nous avons défini les différents termes qui nous seront utiles dans ce mémoire, nous allons pouvoir étudier la didactique de l'informatique et les différentes recherches qui ont été menées dans ce sens.

2) Etat de l'art de la didactique de l'informatique

Dans ce mémoire, nous nous intéressons uniquement à la didactique de l'informatique et de la programmation (qui peut éventuellement se rapporter à la didactique des mathématiques). D'après Fluckiger, « *La didactique de l'informatique est la science qui identifie des contenus informatiques d'enseignement/apprentissage, les construit en tant qu'objets scientifiques, et étudie leurs conditions d'élaboration, de diffusion, de structuration et/ou d'appropriation par les différents acteurs d'un système éducatif.* » (Fluckiger, 2019, p.52). Comme nous l'avons précisé précédemment, l'informatique est considérée comme une science jeune et elle n'a jamais été une matière à part entière dans l'enseignement secondaire. C'est pourquoi, peu de recherches ont été menées en didactique de l'informatique.

L'informatique a été introduite dans l'enseignement secondaire en 1970 suite au colloque de l'OCDE[2] sur « l'enseignement de l'informatique à l'école secondaire ». Ils ont choisi, lors de ce séminaire, de ne pas faire de l'informatique une matière à part entière. Certains enseignants volontaires ont suivi des formations pour pouvoir ensuite gérer des parcs informatiques dans leurs établissements ainsi que pour enseigner les bases de la programmation. L'objectif de cet enseignement était de « *développer chez les élèves des aptitudes algorithmiques, opérationnelles et organisatrices.* ». Les établissements ont pu utiliser le langage de programmation Langage Symbolique d'Enseignement (LSE) qui était destiné à l'apprentissage des bases de la programmation.

A partir de 1981, une option informatique a été mise en place dans les établissements secondaires. La didactique de l'informatique a donc été abordée en France à partir de cette année-là. Les recherches menées dans ce sens ont duré jusqu'à la fin de cette option (dans les années 1990). Cette option a donné naissance à différents colloques organisés par l'Association Francophone de Didactique de l'Informatique de 1988 à 1996. A la fin de cette option, l'informatique était vue en cours de mathématiques ce qui est toujours le cas aujourd'hui.

a) Typologie des enseignements de l'informatique

En 1988, Janine Rogalski propose un état de l'art de la didactique de l'informatique dans la revue de Recherche en Didactique des Mathématique (RDM). Elle répertorie dans cette revue six typologies différentes des enseignements de l'informatique que nous allons différencier maintenant.

La première d'entre elle est **l'informatique éducative.** Celle-ci est présente à tous les niveaux de la maternelle aux études supérieures.

Nous avons ensuite **l'informatique outil.** L'objectif de cette dernière est d'utiliser l'outil informatique dans le but d'acquérir de nouvelles connaissances. Par exemple, en mathématiques, les enseignants utilisent régulièrement le logiciel GeoGebra permettant de représenter certaines situations et pour les élèves de comprendre certaines notions. Cela peut également être un outil de travail ou de calcul.

[2] OCDE : Organisation de coopération et de développement économiques

La troisième typologie d'enseignement de l'informatique d'après elle est **l'informatique technologique ou objet**. Le but de celle-ci est d'utiliser l'informatique en technologie pour, par exemple, programmer des automates.

La typologie suivante est **l'alphabétisation de l'informatique objet et outil.** Il s'agit d'utiliser l'informatique dans le but de *« construire une représentation de l'informatique avec les activités professionnelles et citoyenne ».* (Viallet, 2009, p.13)

Les deux dernières typologies sont **l'initiation pré-professionnelle** et la **formation professionnelle.** La première permet d'acquérir des outils informatiques tels que la programmation. Cela est généralement enseigné en classes préparatoires et en faculté. La deuxième est identique à la première mais celle-ci est délivrée dans les IUT et elle permet également d'acquérir des méthodes et des outils informatiques qui peuvent être utilisés dans le milieu professionnel.

b) Les spécificités de l'enseignement de l'informatique

L'informatique n'est pas une matière à part entière, il s'agit plutôt d'une matière transversale avec des objectifs qui sont communs à l'ensemble des matières enseignées. Il s'agit en effet d'un outil que les enseignant vont utiliser pour aider les élèves à comprendre certaines notions. L'informatique est tout de même plus présente dans les matières scientifiques et principalement en mathématiques que dans les autres matières. En programmation, elle va demander l'utilisation d'outils spécifiques tels que les tests, les itérations ou encore la récursivité. L'une des spécificités de l'informatique est que cela demande du matériel particulier : des ordinateurs, des calculatrices, etc. De plus, chaque établissement, voir chaque salle utilise du matériel, des systèmes d'exploitation ainsi que des logiciels différents. Les élèves utilisent également du matériel qui peut être différent de celui de l'établissement et il est difficile de demander aux élèves de réaliser un travail à la maison en informatique et plus particulièrement en programmation puisqu'ils ne disposent pas tous du matériel nécessaire. Il faudrait réussir dans l'idéal à s'harmoniser pour que tout le monde utilise les mêmes logiciels.

c) Recherches réalisées

Comme nous l'avons dit plus tôt dans ce chapitre, peu de recherches ont été menées en didactique de l'informatique. Nous allons tout de même étudier dans cette partie les recherches de Fabienne Viallet et de Cédric Fluckiger qui ont été réalisées récemment.

En 2009, Fabienne Viallet a réalisé un mémoire sur l'« Analyse transpositive de la boucle : Une contribution à la didactique de l'informatique ». D'après elle, afin de mener une recherche en didactique de l'informatique, on peut se référer à la didactique des mathématiques. Cependant, il faut différencier un certain nombre de choses puisque certains problèmes vont être propres à l'informatique et ne seront donc pas étudiés en didactique des mathématiques. Les savoirs à enseigner sont différents, on doit donc se demander quels sont les savoirs à enseigner en informatique. Fluckiger revient également sur ce point dans ses recherches. En effet, en 2019, il publie le livre « Une approche didactique de l'informatique scolaire » dans lequel il explique que pour réaliser une recherche en didactique de l'informatique, il est important d'étudier les contenus. Les contenus ne sont pas uniquement des connaissances, il s'agit également des savoir-faire, des valeurs, des pratiques ou encore des comportements. « *Un contenu n'est pleinement identifiable par le didacticien qu'au sein de la situation didactique et dans la mesure où il est saisi par un acteur : dans un manuel ou une ressource didactique, en situation de préparation de cours, en situation de classe, etc.* » (Fluckiger, 2019, p.83). Le fait de s'appuyer sur les contenus en didactique permet « *d'analyser des pratiques, des représentations, des modes d'actions des différents sujets didactique* ». (Fluckiger, 2019, p.83).

Lorsque l'on parle de « faire de l'informatique », les élèves pensent seulement à l'utilisation d'ordinateurs alors qu'en réalité, il y a derrière cela une réelle réflexion qui devra passer par le papier avant de se lancer dans la rédaction d'un programme. C'est donc à l'enseignant de montrer qu'il ne s'agit pas uniquement de l'utilisation d'un ordinateur. Fluckiger part du principe que les élèves ont des connaissances et il va s'appuyer sur cela dans l'apprentissage de l'informatique puisque ce sont des outils que les élèves connaissent et qu'ils utilisent régulièrement. En informatique, les élèves doivent être capables de s'adapter à la machine dont ils disposent ainsi qu'aux logiciels qui peuvent être différents selon la machine que l'on utilise.

Dans son mémoire F. Viallet réalise une recherche autour de l'enseignement de la boucle en programmation. Après avoir présenté le concept de boucle, elle énonce les difficultés rencontrées par les élèves lors de l'apprentissage de la programmation et plus particulièrement de la boucle. Elle arrive à la conclusion que la plus grande difficulté vient du langage de

programmation utilisé ainsi que de la notion de récursivité qui est compliquée à faire comprendre aux élèves. Elle réalise donc une étude auprès d'une classe de première année d'IUT pour voir comment cela était enseigné aux élèves. L'enseignant a donc commencé par présenter le langage de programmation ainsi que la notion de boucle. Il a ensuite donné la méthodologie pour réaliser un programme. Pour cela, à partir d'un problème donné, il commence par décomposer cela en plusieurs actions complexes qu'il va traiter une à une. Le but est de réaliser un algorithme qui peut être ensuite programmable dans n'importe quel langage de programmation. Il différencie donc bien l'algorithme et le programme. Une fois que l'algorithme est écrit, il peut ensuite le programmer.

La didactique de l'informatique apporte en ce sens aux enseignants des moyens de transmission des savoirs aux élèves. Pour Viallet, il est important de mener une recherche en didactique de l'informatique au travers de la transposition didactique que nous allons étudier maintenant.

3) La transposition didactique

Les enseignants doivent apporter un certain nombre de savoirs aux élèves. Ces « savoirs savants » sont décrits dans les programmes et référentiels et ne vont pas être abordables comme tel pour les élèves. Il va donc falloir que l'enseignant fasse subir des transformations à ces savoirs savants pour qu'ils puissent être enseignés et compris par les élèves. C'est ce que l'on appelle la « transposition didactique ».

a) Naissance de la transposition didactique

La transposition didactique est une branche de la didactique inventée initialement en 1975 par Michel Verret en sociologie qui a été ensuite reprise en didactique des mathématiques par Yves Chevallard en 1985. Nous allons étudier les travaux de Michel Verret puis ceux de Yves Chevallard.

Verret définit cinq conditions pour pouvoir élaborer des savoirs qui vont pouvoir être enseignés :

- La **désyncrétisation** du savoir. Il s'agit de la division des savoirs pour donner lieu à des pratiques d'apprentissages spécialisées.
- La **dépersonnalisation** du savoir, il s'agit de la séparation de l'enseignant et du savoir. Comme le dit Chevallard cela permet de pouvoir dire aux élèves « vous pouvez me croire, parce que ce n'est pas de moi. » (Chevallard, 1985).
- La **programmabilité** des savoirs, il s'agit de la programmation des apprentissages et de l'organisation des enseignements, des devoirs afin que les élève puissent acquérir les connaissances progressivement.
- La **publicité** du savoir.
- Le **contrôle social** de l'apprentissage.

Ces différentes contraintes vont permettre aux enseignants d'élaborer les enseignements.

b) La transposition didactique selon Yves Chevallard

Yves Chevallard a repris la théorie de Verret. Il définit la transposition didactique de la façon suivante : *« Le passage d'un contenu de savoir précis à une version didactique de cet objet de savoir peut être appelé plus justement « transposition didactique stricto sensu ». Mais l'étude scientifique du processus de transposition didactique suppose la prise en compte de la « transposition didactique sensu lato » représentée par le schéma : → objet de savoir → objet à enseigner → objet d'enseignement »* (Chevallard, p.39, 1991)

On identifie donc trois types de savoirs : le **savoir savant,** il s'agit de celui des chercheurs ; le **savoir à enseigner,** c'est celui qui est écrit dans les manuels, dans les programmes et le **savoir enseigné** qui est l'adaptation des savoirs par l'enseignant pour pouvoir les transmettre aux apprenants.

La transposition didactique est donc composée de deux processus : la transposition didactique **externe** qui va permettre de passer du savoir savant au savoir à enseigner et la transposition didactique **interne** qui va permettre de passer du savoir à enseigner au savoir enseigné.

i) Construction du texte de savoir

La première étape est la transposition didactique externe. Elle permet de construire le « texte de savoir ». Ce dernier est réalisé par ce que Chevallard va appeler la **noosphère**. Celle-ci est constituée des parents, des savants, de l'instance décisionnelle et exécutive, d'associations, etc.

« *La noosphère opte prioritairement pour un rééquilibrage par les moyens de manipulation du savoir*[3] *[en procédant] à une sélection des éléments du savoir savant qui, désignés comme « savoirs à enseigner » seront alors soumis au travail de transposition »* (Chevallard, 1985, p.12). La noosphère procède en deux étapes pour construire le texte de savoir : elle va d'abord sélectionner les savoirs savants qui vont être enseignés puis elle va pouvoir mettre ce savoir en texte. Ces savoirs à enseignés vont être choisis de façon à ce qu'ils soient proches des savoirs savants mais assez éloigné des « savoirs communs » pour qu'ils soient jugés utiles. D'après Verret, c'est la société permet de déterminer si un savoir est enseignable ou non.

ii) Les différents types d'objets du savoir

En didactique des mathématiques, Chevallard différencie trois types d'objets de savoirs, on peut ensuite appliquer cela à la programmation puisqu'il s'agit d'une branche des mathématiques. Le premier d'entre eux est les **notions mathématiques,** il s'agit d'objets de savoirs identifiés et enseignés. Les enseignants construisent et donnent aux élèves des connaissances. Les élèves vont ensuite devoir être capables de donner la définition et des propriétés de ces objets. Ces notions mathématiques vont ensuite pouvoir être évaluées par les enseignants. Le second objet est les **notions para mathématiques.** Ce sont des savoirs essentiels pour pouvoir faire des mathématiques mais qui ne sont pas enseignés en tant que tels. Cela peut être par exemple les équations ou les démonstrations. Ces dernières ne sont pas évaluées directement. Le dernier objet décrit par Chevallard est constitué par les **notions proto mathématiques.** Ce sont des notions qui sont utiles dans les apprentissages, qui sont déjà connues par les élèves et qui ne feront pas l'objet d'un cours. Il s'agit par exemple de la reconnaissance des identités remarquables. Cependant, un même objet de savoir peut être aussi bien considéré comme une notion mathématique, para mathématique ou encore une notion proto mathématique selon le niveau où l'on va sa situer. On peut prendre l'exemple des additions qui sont considérées en école primaire comme une notion mathématique et qui va devenir dans le secondaire une notion para voir proto mathématique. Cela correspond à la désyncrétisation décrite par Verret.

[3] Et non des méthodes d'apprentissage.

iii) La programmabilité des savoirs

Tout comme Verret, pour Chevallard, l'une des étapes de la transposition didactique est la programmabilité des savoirs. Les savoirs doivent être organisés de façon progressive, cumulatives et irréversibles. On détermine le temps que l'on va passer à l'enseignement d'un objet de savoir. Selon lui, un objet de savoir évolue au cours du temps de l'enseignement. En effet, lorsque l'enseignant introduit un objet de savoir en classe, il faut qu'il apparaisse *« comme un objet à deux faces, contradictoires l'une de l'autre. D'une part (c'est le premier « moment », la première « face ») il doit apparaître comme nouveau, opérant une ouverture dans les frontières de l'univers de connaissance déjà exploré ; sa nouveauté permet que se noue, à son sujet, entre enseignant et enseigné, le contrat didactique : il peut être l'objet d'un enseignement et l'enjeu d'un apprentissage. Mais d'autre part, en un second moment de la dialectique d'enseignement, il doit apparaître comme ancien, c'est-à-dire autorisant une identification (par les enseignés) qui l'inscrivent dans la perspective de l'univers des connaissances ancien. […] L'objet d'enseignement est [donc] un objet transactionnel entre passé et avenir.* » (Chevallard, 1991, p.66). Cela signifie que lorsque l'on introduit une notion, elle va d'abord être considérée comme nouvelle par les apprenants et elle va devenir avec le temps quelque chose de connu servant d'outil pour les autres notions qui arriveront plus tard.

Conclusion sur la didactique de l'informatique :

Dans cette partie, nous avons tout d'abord apporté des définitions qui nous permettent de poser clairement les différents termes dont nous avons besoin. Cela nous a amené à la question des spécificités de l'enseignement de l'informatique. Nous avons ensuite réalisé un état de l'art en didactique de l'informatique avec les recherches de Fluckiger qui insiste notamment sur l'importance des contenus et Viallet qui a mené une recherche sur l'enseignement de la boucle et qui nous en donne une méthode. Pour cette dernière, il importe d'étudier la didactique de l'informatique au travers de la transposition didactique.

Nous sommes ensuite arrivés à la transposition didactique qui permet à l'enseignant de transformer les savoirs savants en savoirs enseignés en deux processus : la transposition didactique externe et la transposition didactique interne. Ce concept a vu le jour grâce à Michel Verret puis a été repris en didactique des mathématiques par Yves Chevallard. Il faut donc commencer par construire les textes de savoir et définir le type des objets de savoir. Il convient également de faire un travail de programmation des savoirs de façon à ce qu'ils soient cohérents et progressifs.

III- Sens des apprentissages

Pour un enseignant, il est courant d'entendre « Pourquoi on fait ça ? » ou « à quoi ça va nous servir ? » de la part des élèves. Il est donc important de donner du sens aux apprentissages afin de répondre à leurs questions. Cela permet aux élèves de comprendre le but final des apprentissages et de s'investir dans l'activité proposée. Nous allons donc nous intéresser dans ce chapitre à cela. Nous commencerons par donner une définition du mot « sens » puis nous aborderons la construction des apprentissages chez les « élèves récepteurs » et nous finirons par donner des méthodes qui peuvent permettre de donner du sens aux enseignements apportés.

1) Définition

Le mot sens vient du latin *sensus, « il désigne en général l'action de sentir, de percevoir d'où de nombreuses acceptions : « perception par les sens », « sentiment », dans le domaine intellectuel « manière de voir », « faculté de penser, comprendre » »* (Rey, 2000). Il peut avoir plusieurs significations selon comment on l'emploi, nous allons donc nous intéresser uniquement au « sens-signification », au « sens-direction » et au « sens-sensation ».

Le **sens-signification** va sa construire grâce à la communication « *à partir d'une culture, d'un ensemble de valeurs et de représentations* ». Cela va permettre à l'élève de « *penser l'effort, le but, les récompenses, les risques, à évaluer à ce qu'il en coûte de se donner du mal face à une tâche scolaire et ce qu'on peut attendre de cet investissement.* » (Perrenoud, 2005). Ce sens-signification va être propre à chaque élève, c'est lui qui va le construire ou non dans une situation d'apprentissage.

Le **sens-direction** est le but que l'on cherche à atteindre, les visées de l'enseignement comme le dit Perrenoud, « *L'école n'a de sens pour l'élève que si les situations qu'il rencontre correspondent à sa visée de l'existence.* » (Perrenoud, 2005) L'enseignant a pour objectif de faire comprendre à ses élèves quel est le but et la finalité du travail. Le sens reçu par les élèves n'est pas nécessairement le même que celui que l'enseignant essaie de faire passer. Ce dernier doit donc adapter sa séance et son niveau d'exigence pour pouvoir atteindre des objectifs communs.

Le **sens-sensation,** dont le mot « sensation » est décrit par Rey comme un « *état psychologique à forte composante affective* » (Rey, 2000) et peut être également décrit comme une

« *perception* ». Pour De Vecchi et Carmona-Magnaldi, « *donner du sens à une activité, ce sera agir de telle sorte que l'apprenant soit présent et qu'il ressente l'intérêt du savoir abordé.* ». Pour eux, donner du sens aux apprentissages c'est « *faire éprouver du plaisir* » aux élèves. (De Vecchi & Carmona-Magnaldi, 2004). Meirieu va relier le sens du savoir au désir, en effet, « *ce qui mobilise l'élèves, l'engage dans un apprentissage, […] c'est le désir de savoir et la volonté de connaître.* » (Meirieu, 1995). Pour lui, il est important de ne pas révéler la totalité des connaissances, il faut qu'il en donne une partie de manière à susciter l'attention et le désir d'en apprendre davantage des élèves.

Maintenant que nous avons défini le mot « sens » dans les apprentissages, nous allons voir de quelle manière est construit ce sens par « l'élève récepteur ».

2) L'élève récepteur

Pour qu'un élève construise le sens des apprentissages, il va falloir que celui-ci soit « récepteur ». Le mot « récepteur » vient du latin *receptum* qui signifie retirer, ramener, reprendre, accepter et accueillir. L'élève récepteur va donc accueillir un certain nombre de connaissances, il va être acteur de cette réception d'informations. Pour qu'un élève perçoive une information, il va falloir que celui-ci voit un intérêt dans son apprentissage et qu'il puisse la réutiliser plus tard, en effet, « *l'information n'est identifiée que si elle est déjà, d'une certaine manière saisie dans un projet d'utilisation, intégrée dans la dynamique du sujet et que c'est ce processus d'interaction entre l'identification et l'utilisation qui est générateur de signification* » (Meirieu, 1983). L'enseignant va donc devoir développer des stratégies afin d'aider les élèves à construire ce sens des apprentissages avec par exemple la pédagogie de la ruse qui est une manière détournée d'atteindre l'objectif fixé. Nous étudions maintenant comment se construit le sens des apprentissages chez ces « élèves récepteurs »

3) Construction des apprentissages

Nous allons, dans un premier temps, faire la différence entre les termes « apprendre », « comprendre » et « s'approprier » qui permettent de construire le sens des apprentissages. D'un point de vue étymologique, les termes « apprendre » et « comprendre » sont proches puisqu'ils viennent respectivement des mots latins *apprehendere* qui signifie appréhender au sens et *comprehendere* qui signifie « saisir l'ensemble ». Apprendre est donc « *un acte*

conscient et de conscientisation d'une transformation d'un état de soi, allant d'une transformation de sa base de connaissances, à la possible évolution ou transformation de ses valeurs, éthiques et identité. » (Paquelin, 2004). Pour apprendre, il est nécessaire que les élèves s'approprient la notion étudiée. Cela signifie qu'il faut « *intégrer quelque chose dans son expérience (un fait, un évènement, une situation, une connaissance, une technique…) par sa compréhension, donc par le sens qui lui est donné, en le rapportant à ce qui nous concerne, à ce qui nous soucie. »* (Paquelin, 2004).

Pour construire les apprentissages, il est utile que l'enseignant fasse « *varier les stratégies d'enseignement de manière à ce que les sujets puissent utiliser leur stratégie d'apprentissage. »* (Meirieu, 1995). En effet, chaque élève utilise des méthodes et des stratégies différentes pour apprendre. L'enseignant, en proposant diverses méthodes d'apprentissages offre à chaque élève la possibilité de trouver la stratégie qui lui correspond le mieux.

Pour que les élèves aient l'envie d'apprendre et pour qu'ils s'investissent en classe, l'enseignant doit donner du sens à ce qu'il enseigne. Nous allons donc voir différentes méthodes qui peuvent permettre de donner du sens aux enseignements.

4) Donner du sens aux enseignements

Comme nous l'avons vu précédemment, il est important de donner un sens aux enseignements que l'on apporte pour que les élèves soient davantage motivés. Pour susciter cette motivation d'apprendre, l'enseignant peut employer différentes méthodes.

Tout d'abord, lorsque les élèves connaissent le but à atteindre et qu'ils connaissent l'utilité de ce que l'enseignant leur apporte, ils sont davantage investis en classe. L'enseignant peut également proposer des méthodes assez ludiques qui peuvent passer par le jeu, par l'élaboration de projets. Il peut essayer de favoriser la coopération entre les élèves en faisant par exemple des travaux de groupes où chacun peut apporter des connaissances aux autres. En effet, « *les élèves apprennent dans des moments privilégiés où leur propre réflexion rencontre les savoirs constitués [...] et qu'il est utile de créer des situations complexes dans lesquels on se confronte aux autres pour trouver des solutions* » (Natanson & Berthou, 2013)

Une autre approche afin de donner du sens serait de faire vivre aux élèves la découverte d'un outil, de quelle manière il a été trouvé. Cela peut permettre de mieux comprendre à quoi cela leur sera utile. Une approche historique peut donc intéresser les élèves.

Conclusion sur le sens des apprentissages :

Donner du sens aux apprentissages permet aux élèves de s'investir dans une tâche, d'être motivés et de se donner les moyens de réussir. Nous avons donc vu dans cette partie que le mot sens peut être appréhendé sous plusieurs angles : le sens-signification, le sens-direction et le sens-sensation qui sont tous les trois liés. Pour que les élèves puissent apprendre, l'enseignant peut fournir plusieurs stratégies pour que chaque élève trouve celle qui lui convient. Il s'agit de donner du sens aux apprentissages. Pour cela, la ludification des apprentissages parait être une bonne méthode pour certains élèves ainsi que la coopération entre élèves. D'autres vont cependant préférer savoir comment a été découverte la notion et d'autres encore vont préférer connaître l'objectif et la réutilisation de la notion.

Synthèse du cadre théorique

Dans ce cadre théorique, nous avons étudié différents concepts qui vont nous servir dans la partie méthodologie. Nous avons en effet parlé de la question de la motivation avec différentes théories explicatives qui permettent de comprendre ce qui motive les élèves et ce qui peut permettre de remotiver ceux qui ne le sont pas ainsi que de son importance dans les apprentissages. Cela nous a donc permis d'identifier les différents indicateurs qui nous permettront dans la partie méthodologique de déterminer si les élèves sont motivés ou non et pour quelles raisons. Nous avons ensuite fait une étude de la didactique de l'informatique avec les travaux de Fluckiger et de Viallet. Nous avons donc vu à travers leurs recherches l'importance des contenus et de la transposition didactique qui permet de transformer les « savoirs savants » en « savoirs enseignés ». Nous nous appuierons sur ces recherches pour étudier les différents contenus qui doivent être enseignés ainsi que sur la manière de les apporter et donc cela va nous aider à travailler sur l'enseignement de la programmation en langage Python.

Dans le dernier chapitre, nous avons abordé la notion de sens des apprentissages. Nous avons ainsi défini le mot « sens » sous trois angles complémentaires. Il est important pour les élèves de comprendre le sens de ces apprentissages. Nous avons ainsi proposé quelques méthodes à des enseignants volontaires pour tester l'implication des élèves.

Dans le cadre méthodologique que nous allons voir maintenant, nous mettons en perspective les différents indicateurs que nous avons relevés durant cette approche théorique et les éléments pratiques pour répondre aux différentes questions de recherches.

Cadre méthodologique

L'enseignement de la programmation en langage Python étant nouveau dans les programmes du lycée, nous nous interrogeons sur les manières possibles de l'enseigner. Pour répondre aux différentes questions de recherches que nous avons posées au début de ce mémoire, nous allons recueillir des données afin de mettre en perspective les différents éléments théoriques que nous avons étudiés précédemment et des éléments pratiques relatifs à cet enseignement. Pour ce faire, nous commençons par présenter le contexte de cette étude ainsi que les différentes méthodes de recueils de données utilisées. Nous présenterons ensuite les résultats obtenus que nous analyserons.

I- Présentation du dispositif méthodologique

Nous allons dans un premier temps présenter le dispositif méthodologique, pour cela nous commencerons par faire une présentation rapide du public avec lequel nous avons mené cette étude. Nous verrons ensuite les outils de recueils de données que nous avons utilisés ainsi que la manière dont nous allons les analyser.

1) Public concerné

Nous menons cette étude auprès d'une classe de Première Générale de 22 élèves. Dans les années futures les élèves de première générale auront déjà programmé en Python en classe de seconde en mathématiques ainsi qu'en Sciences Numériques et Technologie (SNT) mais cette année, les élèves de première n'ont, pour certains d'entre eux, jamais programmé en Python. En effet, ces élèves ont suivi les cours avec l'ancien programme de seconde dans lequel le langage Python n'était pas clairement écrit. Certains enseignants en ont tout de même fait en classe mais ce n'est pas le cas de tous. Par conséquent, les élèves ne sont pas tous arrivés avec le même niveau et il a donc fallut, en début d'année, réaliser quelques séances pour apprendre les bases de la programmation en langage Python à ceux qui n'en avaient encore jamais fait, ce qui a également permis aux autres d'en faire des révisions. Une seconde spécificité de cette classe est qu'ils ne disposent pas de calculatrices Python et n'ont donc pas la possibilité de programmer avec en classe.

Nous allons également nous appuyer sur des enseignants de mathématique et de SNT qui enseignent en classe de seconde. Dans ces classes, les élèves disposent chacun d'une calculatrice permettant de programmer en Python ce qui permet de faire de la programmation en classe sans avoir de matériel supplémentaire.

2) Outils utilisés

Dans le tableau 2 ci-dessous, nous mettons en perspective les questions de recherche ainsi que les indicateurs présentés dans le cadre théorique qui ont permis de mener cette étude et de recueillir des données appropriées.

Tableau 2 : Les outils utilisés

Questions	Cadre théorique	Indicateurs	Outils méthodologiques
Les élèves sont-ils motivés par la programmation ? Ou faut-il trouver des moyens pour qu'ils le deviennent ?	La motivation scolaire	Motivation intrinsèque/extrinsèque Niveau d'aspiration / d'expectation Motivation par les moyens ou par les finalités Attributions causales et contrôlabilité Motivation et estime de soi	Observation des séances de TP et de cours Questionnaire élève Questionnaire enseignant
Quels sont les savoirs à mobiliser pour programmer en Python ?	Didactique de l'informatique	Connaissance de l'algorithme et de la programmation Fluckiger : l'importance des contenus	Analyse de référentiels Questionnaire élève Questionnaire enseignant
Quelles sont les méthodes à utiliser pour enseigner ce langage ? Quelle programmabilité des savoirs ?	Sens des apprentissages Didactique de l'informatique Transposition didactique	Donner du sens : Pourquoi on fait cela ? Ludification des apprentissages Les différents types d'objets du savoir Programmabilité des savoirs Construction des apprentissages	Questionnaire enseignant Questionnaire élève Synthèse des observations réalisées en classe Analyse des programmes de maths et de SNT
Quels logiciels utiliser ?			Questionnaires élèves et enseignants Observations en classe
Comment l'évaluer ?			Questionnaire enseignant

Dans le cadre théorique, nous avons étudié les travaux de Fluckiger pour qui il était important de s'appuyer en didactique de l'informatique sur les différents contenus à enseigner. Nous commençons donc par analyser les programmes de mathématiques et de SNT de la classe de seconde générale et technologique ainsi que ceux de la spécialité mathématiques de la classe de première générale. Cette analyse a donc pour but d'identifier les différents contenus qui doivent être enseignés et à quel niveau. Cela va permettre d'aborder la question de la programmabilité des savoirs et de la transposition didactique étudiés dans le cadre théorique.

Nous avons, tout au long de l'année, fait de la programmation en langage Python avec la classe de première présentée précédemment. Nous présentons les méthodes que nous avons utilisées ensemble pour enseigner la programmation en Python. Nous avons, à travers ces cours, pu observer un grand nombre de choses, nous en ferons une synthèse. Nous avons réalisé ensemble des séances de Travaux Pratiques sur ordinateurs où les élèves ont pu manipuler le logiciel et nous avons fait des exercices avec des programmes à compléter en classe. Nous avons observé, lors de ces séances, les difficultés rencontrées par les élèves, les contenus qu'ils maîtrisent ou qu'ils ne maîtrisent pas, leur motivation (à travers divers indicateurs) ainsi que leurs réactions et leurs demandes vis-à-vis de cet enseignement.

Pour réaliser une telle recherche, il semble important de recueillir l'avis des élèves qui sont les mieux placés pour répondre à certaines questions que nous nous posons. Pour recueillir ces informations, nous avons utilisé un questionnaire en ligne pour que les élèves puissent y répondre facilement de chez eux. Le questionnaire semblait l'outil le plus adapté puisqu'il permet de recueillir davantage de données que l'entretien par exemple et nous pouvons recueillir aussi bien des données quantitatives que qualitatives. De plus, les élèves se sentent généralement plus à l'aise pour répondre honnêtement à un questionnaire en ligne et anonyme. Cela permet de recueillir des données relativement fiables. Le questionnaire donné aux élèves se trouve en Annexe 1. Ce questionnaire a permis de recueillir un grand nombre de données sur l'ensemble des indicateurs que nous avons étudiés dans le cadre théorique. Ce sont des données quantitatives et qualitatives qui vont nous permettre d'analyser ce que pensent et ce que ressentent les élèves face à la programmation en langage Python. Le questionnaire a donc été rempli par la classe de première générale sur laquelle nous avons mené une partie de cette étude donc par 19 élèves au total. Dans ce questionnaire, les élèves ont été interrogés à travers une vingtaine de questions à propos de leur motivation, de ce qu'ils se sentent capable de faire, de leurs préférences vis-à-vis de la fréquence des séances, des logiciels utilisés, des méthodes employées par l'enseignant, mais également de leur ressenti et de leurs difficultés.

Chaque enseignant utilise des méthodes différentes pour enseigner la programmation à des classes et à des publics différents. Il était donc intéressant d'interroger également des enseignants pour pouvoir analyser leurs pratiques ainsi que les réactions de leurs élèves. Nous avons, tout comme pour les élèves, réalisé un questionnaire en ligne destiné aux enseignants (Annexe 2). Ce questionnaire d'une trentaine de questions a été rempli par trois enseignants du lycée dans lequel cette étude a été menée, deux de mathématiques et un de SNT, ainsi que par cinq enseignants stagiaires de mathématiques pour avoir une vision plus large et pouvoir comparer les différentes pratiques. Ce questionnaire a permis de récolter des données quantitatives et qualitatives à propos de la motivation des élèves, des contenus enseignés qui sont ou non maîtrisés par les élèves, des logiciels utilisés, de la fréquence des séances et des pratiques enseignantes en générale. Pour les enseignants, nous aurions pu choisir de réaliser des entretiens individuels mais nous n'aurions pas pu en interroger autant et il semblait plus intéressant d'avoir plusieurs avis différents plutôt que de se limiter à seulement un ou deux.

Les avis des élèves et des enseignants peuvent souvent être différents, vis-à-vis de leur ressenti face à un même cours par exemple. En interrogeant les élèves et les enseignants, nous pouvons croiser les impressions et les mettre en perspective. Nous pourrons ainsi trouver des méthodes susceptibles d'enseigner d'une façon satisfaisante la programmation tout en tenant compte de l'avis des élèves.

Maintenant que nous avons présenté les différents outils qui vont permettre de répondre aux différentes questions de recherches, nous allons voir de quelle manière ces données vont être analysées.

3) Les modalités d'analyse

Après avoir recueilli et présenté les données, nous les analysons. D'après Mucchielli, l'analyse est une « *méthode capable d'effectuer l'exploitation totale et objective des données informationnelles.* » (Mucchielli, 1998). Cela permet de classer, résumer, quantifier et interpréter ces différentes données dans le but de répondre à nos questions de recherches. Laurence Bardin explique qu'il faut « *savoir pourquoi on analyse, et l'expliciter, pour savoir comment analyser.* » (Bardin, 1993). Selon elle, pour réaliser une analyse de contenu, il faut procéder en trois étapes : la **préanalyse**, l'**exploitation du matériel** et pour finir, le **traitement** et l'**interprétation des résultats.** La phase de préanalyse consiste à choisir les documents, à formuler des objectifs ainsi qu'à déterminer les indicateurs qui permettront d'interpréter les

résultats. La phase d'exploitation du matériel est celle qui va permettre d'appliquer le programme défini dans la première phase avec tout le matériel informatique dont nous avons besoin. Dans notre cas, nous utilisons des questionnaires en ligne comme nous l'avons décrit précédemment. Nous avons défini l'élaboration de ces deux phases dans la partie précédente. La dernière phase de l'analyse a pour but de rendre les résultats significatifs et valides. Nous allons maintenant nous intéresser à la manière d'analyser et d'interpréter les résultats que l'on a recueillis.

Dans un premier temps nous analysons les programmes de seconde et de première. Il s'agit ici d'analyser des documents officiels sur lesquels l'ensemble des enseignants doivent s'appuyer pour réaliser leurs cours. Ils doivent enseigner l'ensemble des contenus décrits dans les programmes mais ils ont la liberté de choisir leur programmabilité et la manière de les enseigner. Nous avons donc analysé les programmes de seconde et de première générale afin de déterminer les différents contenus qui doivent être apportés aux élèves. Ces différents contenus vont être le départ de l'ensemble des enseignements et des choix pédagogiques. Nous pouvons ainsi les classer selon le niveau et la matière (maths ou SNT). Nous pourrons ensuite confronter ces différents contenus que l'on aura trouvés aux réponses des élèves et des enseignants au sujet des contenus que les élèves sont capables ou non de programmer.

Comme nous l'avons dit précédemment, nous avons observé tout au long de l'année les séances de programmation en langage Python ainsi que la réaction des élèves face à cet enseignement et leurs demandes. Nous décrirons dans un premier temps le fonctionnement des cours et l'ensemble des observations des élèves qui ont été faites. Nous pouvons tirer de ces observations les difficultés des élèves et les contenus sur lesquels il faut insister davantage. Nous pouvons également voir les réussites des élèves, ce qui a fonctionné dans la manière de faire ou ce qu'il faut améliorer. Nous pourrons également confronter cela aux réponses obtenues dans les questionnaires des élèves puisque ces derniers peuvent avoir un avis différent.

Quant aux questionnaires remplis par les élèves et par les enseignants, nous commencerons par faire une analyse purement statistique. Nous pouvons mettre en perspective ces réponses et les éléments théoriques que nous avons étudiés pour tirer des conclusions sur l'enseignement de cette matière. Nous pouvons, par exemple, selon les réponses apportées des élèves voir s'ils sont motivés et de quelle manière (intrinsèque ou extrinsèque). L'ensemble des indicateurs relevés dans le cadre théorique nous serviront à analyser ces réponses et nous permettront finalement de trouver des réponses à nos questions.

Nous avons maintenant décrit le dispositif méthodologique avec le public qui nous a permis de mener cette recherche à bien, les outils que nous avons utilisés et la manière d'analyser les données recueillies. Nous allons donc pouvoir présenter les différents résultats obtenus lors de cette étude.

II- Présentation des résultats

1) Programmes

Nous commençons par nous intéresser aux programmes de mathématiques des classes de seconde générale et technologique et de première générale (spécialité mathématiques) ainsi qu'à ceux de SNT de la classe de seconde. Le but est d'extraire les différents contenus concernant la programmation en langage Python qui doivent être enseignés dans ces classes.

Les programmes de mathématiques des deux classes sont structurés de la même façon et comportent beaucoup de choses identiques en matière d'algorithmique et de programmation. Il est bien écrit dans les deux programmes, dans la partie « Algorithme et programmation » que les algorithmes qui sont traités doivent être en relation avec l'ensemble des parties du programme ainsi qu'avec les autres disciplines et la vie courante. L'enseignement de la programmation ne doit donc pas se faire sous la forme d'un seul chapitre mais doit être fait durant toute l'année à travers les autres chapitres. En effet, dans la plupart des chapitres qui sont présentés dans les programmes, on nous propose des exemples d'algorithmes à réaliser avec les élèves et qui permettent de travailler les différents contenus présents dans la partie « Algorithme et programmation ».

Pour les deux niveaux, l'enseignement de l'algorithmique et de la programmation doit se baser sur deux notions essentielles, la notion de fonction qui est une notion très utilisée dans ce domaine et la « *programmation comme production d'un texte dans un langage informatique* »[4]. Les élèves doivent être capables de décrire des algorithmes en langage naturel ou des programmes dans des langages de programmation (le langage à utiliser obligatoirement est

[4] BO Programme de mathématiques de seconde générale et technologique et Programme de mathématiques de première générale

Python mais il est possible d'utiliser également d'autres langages). Ils doivent également être capables de passer de l'un à l'autre. A l'issue de cet enseignement, il faudrait que les élèves réussissent à réaliser quelques programmes à partir de programmes simples écrits dans un langage de programmation et à « *interpréter, compléter ou modifier des algorithmes plus complexes* ». Ces compétences générales sont présentes dans les programmes de mathématiques des deux niveaux, ce sont les éléments qui suivent qui vont se différencier.

A la suite de ces compétences, on peut observer les contenus à apporter aux élèves et les différentes capacités qu'ils doivent maîtriser. Pour la classe de seconde, nous différencions deux axes : « *Utiliser les variables et les instructions élémentaires* » et « *Notions de fonction* ». Dans le premier, on retrouve les notions de variables (entiers, flottants, booléens et chaînes de caractères), les affectations, les séquences d'instructions, les instructions conditionnelles et les boucles bornées et non bornées (For et While). Ce sont les premières choses à apprendre pour pouvoir commencer à programmer en Python, nous retrouvons toutes ces notions dans beaucoup d'algorithmes. Nous avons ensuite les différentes capacités qui sont associées à ces notions. Dans le second axe, nous retrouvons la notion de fonctions avec les fonctions à un ou plusieurs arguments et les fonctions renvoyant un nombre aléatoire. Les élèves doivent être capable d'écrire, de compléter ou de modifier des fonctions simples et faire appel à des fonctions dans des programmes. Ils doivent également être capables d'utiliser des fonctions dans des programmes en statistique et en probabilité.

En classe de première, on retrouve les mêmes compétences générales mais les contenus vont ensuite être différents. Les notions vues en secondes devront être connues des élèves puisqu'ils vont en avoir besoin dans un grand nombre de programmes. A cela, va se rajouter la notion de « listes ». Ils devront alors apprendre cette nouvelle notion en utilisant toujours ce qu'ils ont vu précédemment. Cette notion doit être mise en lien avec la notion d'ensemble. Les élèves vont devoir être capables de générer une liste de plusieurs manières différentes (en extension, par ajouts successifs, en compréhension), de manipuler les éléments d'une liste ainsi que leurs indices, de parcourir une liste et d'itérer sur les éléments d'une liste.

Le programme de SNT est construit différemment de ceux de mathématiques mais les contenus ne diffèrent pas du programme de mathématiques de seconde générale. C'est dans la manière de les apporter que l'on remarque des différences. L'algorithme et la programmation sont présents dans l'ensemble du programme, utilisés comme un outil pour réaliser d'autres tâches. Les applications de la programmation se font notamment en physique et en économie. Nous pouvons observer dans l'ensemble des notions qui sont présentées, des exemples d'activités

possibles. On retrouve par exemple de la programmation dans le traitement de données ou dans le traitement d'images.

Nous avons, dans le tableau suivant, un récapitulatif des contenus et des capacités spécifiques à chaque niveau et à chaque matière :

Niveau/Matière	Contenus	Capacités
Seconde : mathématiques	- Variables - Affectations - Séquences d'instructions - Instructions conditionnelles - Boucles bornées et non bornées - Fonctions à un ou plusieurs arguments	Choisir ou déterminer le type d'une variable (entier, flottant ou chaîne de caractères). Concevoir et écrire une instruction d'affectation, une séquence d'instructions, une instruction conditionnelle. Écrire une formule permettant un calcul combinant des variables. Programmer, dans des cas simples, une boucle bornée, une boucle non bornée. Dans des cas plus complexes : lire, comprendre, modifier ou compléter un algorithme ou un programme. Écrire des fonctions simples ; lire, comprendre, modifier, compléter des fonctions plus complexes. Appeler une fonction. Lire et comprendre une fonction renvoyant une moyenne, un écart type. Aucune connaissance sur les listes n'est exigée. Écrire des fonctions renvoyant le résultat numérique d'une expérience aléatoire, d'une répétition d'expériences aléatoires indépendantes.
Seconde : SNT	Identiques à ceux de mathématiques	Ecrire et développer des programmes pour répondre à des problèmes et modéliser des phénomènes physiques, économiques et sociaux
Première Générale : Mathématiques	Notions de listes	Générer une liste (en extension, par ajouts successifs ou en compréhension). Manipuler des éléments d'une liste (ajouter, supprimer…) et leurs indices. Parcourir une liste. Itérer sur les éléments d'une liste.

2) Observations réalisées en classe

Nous avons, tout au long de l'année, fait de la programmation en Python avec la classe de première générale. Les élèves n'étaient pas à égalité dans ce domaine puisqu'ils venaient tous de classes (voir d'établissement différents). En effet, certains d'entre eux avaient déjà programmé un petit peu en Python en classe de seconde alors que d'autres pas du tout. Ce type d'inégalités ne devrait plus avoir lieu dans les années futures puisque cet enseignement est maintenant inscrit dans les programmes dès la seconde. Nous avons donc, en ce début d'année fait en sorte que tout le monde ait vu les mêmes notions. Pour cela, nous avons réalisé quelques exercices en classe avec des programmes déjà écrits, qu'il fallait « faire tourner » à la main afin de comprendre la démarche de l'algorithme et de voir comment on programme les différentes fonctionnalités dans le langage Python.

Nous avons également réalisé une séance de TP courant septembre en salle informatique. Les élèves étaient par deux sur un PC en raison du nombre de postes disponibles dans la salle. Le but de cette séance était d'apporter les bases du langage Python aux élèves qui n'en avaient encore jamais fait et cela a permis également de faire des rappels aux autres sur les différentes notions qu'ils devaient normalement voir en seconde (variables, affectations, fonctions, instructions conditionnelles, boucles bornées et non bornées). La programmation est une science cumulative, c'est à dire que les élèves ont besoin de connaître ces différentes notions pour pouvoir continuer à en apprendre de nouvelles. Ce TP de début d'année semble donc important pour revoir les bases et pour pouvoir continuer à avancer. Pour réaliser ce TP, un document a été distribué aux élèves avec les différentes notions présentées et des exemples pour chacune d'entre elles. Les élèves ont ainsi pu manipuler ces dernières par eux-mêmes et ont pu les tester d'où l'importance de les emmener en salle informatique.

Une fois que les élèves ont été capables de manipuler ces différents outils, nous avons pu passer à la suite en abordant la nouvelle notion au programme de première : les **listes**. Nous avons donc abordé cela au cours d'une séance de TP au mois d'octobre. Cette séance a été réalisée en début d'année pour qu'ils puissent s'approprier les listes et pouvoir les réutiliser tout au long de l'année. Les élèves ont reçu un document présentant les listes (Annexe 3), leurs utilisations et quelques exemples pour illustrer cela. Sur ce document, un tableau était présenté avec un certain nombre de commandes utiles pour utiliser les listes ainsi que des exemples. Pour finir, différents exercices étaient proposés pour qu'ils puissent s'entraîner. Durant cette séance, nous

avons commencé par regarder les exemples donnés en début du document afin de comprendre le fonctionnement de ces listes ainsi que les indices. Nous sommes ensuite passés aux exercices pour que les élèves puissent s'exercer rapidement. Nous avons réalisé le premier exercice tous ensemble pour qu'ils comprennent le fonctionnement, ils ont ensuite réalisé les suivants en prenant le premier pour exemple et en le modifiant. Lors de ces séances, l'enseignant circule dans la classe pour guider les élèves, répondre à leurs questions et corriger les éventuelles erreurs. La correction se faisait au vidéo projecteur en expliquant la démarche, ce qui a permis à la majorité d'entre eux de comprendre. Nous avons remarqué, lors de cette séance que les élèves avaient des difficultés pour utiliser les différentes commandes et notamment l'utilisation des boucles qui permettent de parcourir ou de générer une liste. Dans l'ensemble, grâce aux explications, les élèves ont compris comment cela fonctionne mais ont du mal à le faire seul. Certains ne comprennent pas pourquoi on utilise toutes ces fonctionnalités alors qu'on pourrait le faire à la main. En expliquant que l'on pouvait avoir des listes comportant des centaines voire des milliers de variables, ils ont mieux compris l'utilité et se sont davantage investis. Certains élèves étaient assez motivés par cet enseignement et participaient bien mais d'autres ne l'étaient pas du tout et ne faisaient rien sans y être poussé. Cependant, la programmation prend beaucoup de temps aux élèves et nous n'avons pas pu réaliser beaucoup d'exercices.

Nous avons ensuite principalement réalisé des exercices en classes ou à faire à la maison. Ce sont généralement des programmes à compléter ou à modifier sur différents chapitres du programme de mathématique. Dans ces exercices, les élèves devaient utiliser les notions de Python qu'ils ont vu en début d'année pour réaliser des programmes qui permettent d'automatiser des calculs liés à des connaissances variées. Cela permet également de consolider les différentes capacités demandées en termes de programmation en langage Python. Nous avons ainsi pu travailler la programmation mais nous n'avons pas pu réaliser beaucoup de TP à cause du manque de temps et des difficultés d'accès aux salles informatiques. Les élèves n'ont donc pas pu programmer énormément par eux-mêmes. Ils ont donc des difficultés dans ce domaine et ne réussissent pas à identifier leurs erreurs. Nous avons réalisé des programmes ensemble sur Python mais cela ne permet pas à l'ensemble de la classe de s'exercer. Il faudrait pour cela, qu'ils aient tous accès à du matériel informatique ou à une calculatrice permettant de programmer en Python pour le faire. Suite aux difficultés qu'ils ont rencontré, les élèves ont perdus pour certains la motivation qu'ils avaient en début d'année mais étaient tout de même toujours demandeurs de séances de TP pour pouvoir s'améliorer et pour pouvoir réussir les examens qui les attendaient en fin d'année.

Nous avons tout de même pu réaliser quelques séances de TP durant lesquelles les élèves ont pu s'exercer. Durant ces séances, le manque de temps s'est fait ressentir mais les élèves étaient motivés et ont essayé de faire le travail demandé en posant des questions pour avancer. Nous avons tout de même pu remarquer durant cette séance, que les élèves avaient progressés durant l'année et réussissaient à faire plus de programmes par eux-mêmes.

En fin d'année, en raison des conditions sanitaires, les cours à distance ont permis d'utiliser d'autres moyens pour enseigner la programmation. Les élèves ont eu quelques exercices à faire en Python (sur EduPython ou Python en ligne selon leurs possibilités). Cela leur a permis de tester les programmes mais ils ont eu quelques difficultés pour le faire et personne pour les aider. Après quelques explications, ils ont compris la démarche et la correction. Comme nous ne pouvions pas faire de cours en présentiel, les corrections des exercices ont été mis sur Pronote pour que les élèves y aient accès. Cela a permis de trouver une autre manière de faire qui pourra être utile pour la suite.

Cependant, nous avons évalué les élèves qu'une seule fois lors d'un devoir sur les suites où ils devaient compléter un programme. Cela a été réussi par une majorité d'entre eux.

En résumé, nous pouvons constater que certains élèves sont motivés par la programmation et veulent progresser dans ce domaine tandis que d'autres ne comprennent pas son utilité. Cet enseignement étant nouveau pour eux, ils rencontrent certaines difficultés pour réaliser des programmes eux-mêmes mais globalement, ils comprennent les programmes réalisés avec quelques explications. Ils sont très demandeurs de séances de TP ce qui n'est pas toujours possible en raison du manque de temps et de matériel.

3) Questionnaires élèves

Nous allons présenter les réponses des élèves au questionnaire donné à remplir. Sur les 22 élèves de la classe de première, 19 ont répondus au questionnaire (les trois autres n'ont pas pu le faire pour des raisons techniques).

Les premières questions concernaient la motivation des élèves. Nous pouvons voir que parmi ces 19 élèves, 15,8 % d'entre eux sont intéressés par la programmation, 47.4 % le sont mais ils rencontrent des difficultés dans cette matière et 36.8 % ne sont pas du tout intéressés. 42.1 % des élèves interrogés sont motivés pour faire de la programmation en Python en raison de

l'utilisation de l'informatique, du côté ludique de ce type d'enseignement. Une autre raison donnée est le fait que cela change de l'enseignement des mathématiques plus classiques qu'ils ont l'habitude de faire depuis leur enfance. Un peu plus de la moitié des élèves qui sont motivés par cette matière ont des difficultés et n'arrivent pas forcément à programmer seuls mais ils pensent pouvoir y arriver avec de l'aide. Cependant, 57.9% d'entre eux ne sont pas motivés pour en faire. Les principales raisons données sont qu'ils trouvent cela trop compliqué et donc qu'ils ont arrêtés de faire des efforts, qu'ils ne comprennent pas à quoi ça sert ou alors qu'ils n'aiment pas cela. Une majorité d'entre eux (63.6 %) ont été motivés à un moment donné par cet enseignement mais comme ils n'y arrivaient pas ou comme cela ne correspondait pas à leurs attentes, ils ont arrêté.

Les questions suivantes concernent les logiciels utilisés en classe et la fréquence des séances que les élèves voudraient avoir. La majorité des élèves utilisent le logiciel EduPython en classe. Seul deux d'entre eux ont une calculatrice permettant de programmer dans ce langage. Chez eux, la plupart des élèves (16 d'entre eux) n'utilisent aucun logiciel pour programmer, ce qui signifie que peu d'entre eux s'exercent à cela en dehors des cours. Pour les autres qui utilisent un logiciel, il s'agit du logiciel EduPython ou de la calculatrice. Pour ce qui est de l'utilisation en classe, une grande partie des élèves pensent que l'utilisation d'EduPython est le moyen le plus adapté pour faire de la programmation et environ un quart de la classe pense que la calculatrice peut être un outil adapté. Pour programmer de chez eux, un peu plus de la moitié d'entre eux pensent que le logiciel EduPython est le plus adapté. Pour chacun des autres outils proposés (calculatrice, application du Smartphone et Python en ligne), environ 30 % des élèves pensent qu'ils sont adaptés. La moitié de la classe préfère faire des TP pour apprendre cette matière, 15.8 % préfèrent faire uniquement des exercices en classe incorporés au cours et le reste de la classe préfère réaliser un petit peu des deux. La plupart des élèves préfèrent travailler la programmation uniquement en classe et non à la maison. Pour ce qui est de la fréquence des TP, ce qui ressort des questionnaires est que les élèves voudraient réaliser des séances de TP environ une fois toutes les deux semaines. Il en est de même pour faire des exercices de programmation en cours même si certains voudraient en faire une fois par semaine (environ 20%) ou une fois par mois (15%). Dans la classe, 68.4% des élèves pensent que la programmation est utile.

Nous avons ensuite interrogé les élèves sur des méthodes qui pourraient être utilisées pour enseigner la programmation, en particulier l'élaboration d'un projet. Les résultats sont assez mitigés puisqu'environ la moitié de la classe serait intéressée par l'apprentissage de la

programmation sous la forme d'un projet et l'autre moitié ne le serait pas. Parmi ceux qui ont répondu que cela les intéresserait, la majorité le sont pour le côté ludique de cette méthode d'apprentissage et pour apprendre de nouvelles choses par eux-mêmes. Ils voudraient réaliser ce projet par groupe de 4 et voudraient tout de même réaliser des cours théoriques qui permettraient d'acquérir les bases avant de démarrer le projet. La principale raison donnée par les élèves qui ne voudraient pas réaliser de projet est qu'ils préfèrent faire des cours et des TP classiques. 78.9 % de la classe voudrait que l'enseignant mette sur Pronote des petits programmes qu'ils pourraient exécuter et comprendre de chez eux. Cela pourrait également leur servir pour réaliser des programmes plus complexes. Une autre méthode qui pourrait être utilisée pour enseigner la programmation serait que les élèves arrivent en classe avec un programme sur clé USB pour pouvoir le montrer à la classe et en discuter. Cette méthode plairait à un peu plus de la moitié des élèves mais ils ne se sentent pas capable de le faire. 26.3 % des élèves ne voudraient pas utiliser cette méthode et seul le reste de la classe pense que cela pourrait être enrichissant pour tout le monde.

Les questions suivantes concernaient ce que les élèves se sentent capable de réaliser en fin d'année. Nous pouvons constater que très peu d'élèves se sentent capable de réaliser des programmes seuls. 52.6 % d'entre eux ne pensent pas pouvoir le faire seul mais ils comprennent ceux qui sont réalisés en classe et 36.8 % ne comprennent pas du tout ceux qui sont faits. Les contenus qui sont les mieux maîtrisés par les élèves sont les boucles bornées (For), les listes et les fonctions. En effet, 6 élèves pensent être capables d'utiliser ces notions seuls mais 8 d'entre eux ne se sentent pas capables d'en utiliser seul. Les deux principales notions que les élèves comprennent lorsque l'enseignant les explique mais qu'ils ne se sentent pas capable de réaliser seul sont les listes et les fonctions. Seulement 3 élèves ont répondu qu'ils ne comprennent aucune de ces notions. Cependant, il reste encore plusieurs notions que les élèves ne comprennent pas du tout. Seulement 6 élèves ont répondu qu'il n'y avait aucune de celles-ci qu'ils ne comprenaient pas.

Pour 42 % d'entre eux, la programmation leur a servi à améliorer leur logique et pour 47 %, cela permet de faciliter certains calculs qui sont trop compliqués à réaliser à la main. Cela a tout de même permis à quelques-uns d'entre eux de mieux comprendre certaines notions mathématiques comme par exemple les fonctions. En revanche, pour 7 d'entre eux, ils n'ont pas l'impression que ça leur soit utile.

Globalement, les élèves voudraient réaliser plus de séances de TP et avoir plus de temps pour travailler la programmation. Ils voudraient que les séances soient plus rapprochées car ils

oublient certaines notions d'une séance à l'autre. Un élève a également demandé des fiches de révisions synthétiques.

Nous avons donc vu dans ce questionnaire l'avis des élèves concernant la programmation en Python et nous analyserons ces réponses par la suite. Nous allons maintenant présenter les résultats des questionnaire remplis par les enseignants.

4) Questionnaires enseignants

Afin de différencier les réponses aux questionnaires des enseignants de l'établissement et des enseignants stagiaires, nous avons recueilli les réponses séparément. Les enseignants du lycée interrogés interviennent en mathématiques, en SNT ou en enseignement scientifique avec des classes de seconde, de première générale, première STAV et terminale scientifique tandis que les stagiaires interviennent seulement en mathématiques avec des classes de seconde, première générale ou première STAV.

Les premières questions portaient, comme pour les élèves, sur la motivation. Globalement, tous les enseignants interrogés étaient en accord avec le fait que la motivation pour la programmation dépendait des élèves ou alors qu'ils ne l'étaient pas du tout. D'après eux, les principales raisons pour lesquelles les élèves sont motivés sont l'utilisation de l'informatique, le changement par rapport aux cours de mathématiques « classiques » et la fait qu'ils ne réussissent pas encore à programmer seuls mais qu'ils pensent pouvoir réussir avec de l'aide. Les principales raisons données pour lesquelles certains élèves ne sont pas motivés par la programmation sont qu'ils trouvent que c'est trop compliqué, qu'ils n'aiment pas cela, qu'ils ne comprennent pas à quoi cela va leur servir et qu'ils se sentent « nul » en maths et donc en programmation aussi.

Les questions suivantes portaient sur les pratiques enseignantes. Une majorité des enseignants interrogés font des séances de TP environ une fois toutes les deux semaines et des exercices en classe toutes les semaines voir plus rarement pour certains. Quelques-uns d'entre eux donnent également aux élèves des exercices à faire à la maison. Parmi l'ensemble des enseignants qui ont répondu au questionnaire, un seul a déjà réalisé un projet avec l'une de ses classes. Celui-ci a duré un mois et les élèves étaient seuls pour le réaliser. Cela a permis de faire le lien avec la séquence sur l'algorithmique. Les élèves ont globalement apprécié ce projet pour son côté

ludique et pour pouvoir apprendre des choses par eux-mêmes. Ce dernier serait donc prêt à en réaliser un autre dans le futur « à cause des multiples applications dans la poursuite des études. ». Pour ceux qui n'en n'ont jamais réalisé, ils pensent que la réalisation d'un projet peut être difficile à organiser et qu'ils risquent de manquer de temps. Ils ont tout de même identifié des moyens pour faciliter la réalisation d'un tel projet comme faire régulièrement des séances de TP en accompagnant les élèves dans son élaboration. Un enseignant a également suggéré « une collaboration avec un enseignant de sciences physiques et/ou de SNT pour la programmation d'un appareil simple ». Ils pensent tous que cela pourrait permettre aux élèves d'être davantage motivés et qu'ils vont s'investir dans le projet. Une grande partie d'entre eux (71 %) seraient prêts à en réaliser avec leurs classes. Ils privilégient plutôt des petits groupes de cinq élèves au maximum afin d'avoir plusieurs idées différentes mais pas trop nombreux pour permettre de les encadrer plus facilement. Tous pensent que la réalisation de quelques séances sur les bases de la programmation serait appropriée avant de démarrer un projet dans lequel l'enseignant peut aider individuellement les différents groupes. Les outils les plus utilisés sont EduPython et les calculatrices permettant de programmer en Python pour ceux qui en ont, certains utilisent également Python en ligne. Le logiciel EduPyton semble tout de même le plus adapté pour enseigner la programmation. Parmi les huit enseignants, cinq pensent qu'il faut commencer directement à programmer en Python alors que les autres pensent qu'il vaudrait mieux procéder par étape en commençant avec un langage plus facile comme par exemple Algobox avant de passer au langage Python. La plupart d'entre eux pensent également qu'il vaut mieux que les élèves commencent par faire un algorithme sur papier avant de passer sur la machine pour le programmer.

Nous avons ensuite interrogé les enseignants sur les contenus qu'ils enseignent et qui sont maîtrisés par les élèves. Les contenus enseignés sont ceux qui sont décrits dans les programmes et qui sont exposés dans la partie II-1) du cadre méthodologique. Sur ces différents contenus, les boucles bornées (For) et non bornées (While) ne sont pas forcément bien maîtrisées par les élèves même si les bornées le sont un peu plus. Cependant selon les enseignants, les élèves semblent capables d'utiliser les fonctions et, pour ceux qui en font, les listes.

Pour la question de l'évaluation, une bonne partie des enseignants a déjà évalué les élèves en programmation principalement en incorporant des exercices ou des questions dans un devoir qui n'est pas spécifiquement sur la programmation mais qui peut être sur une autre notion au programme. Cela a également été fait sous forme de devoir maison ou plus rarement en séance

de TP. Deux des enseignants pensent que leurs élèves ne sont pas encore prêts pour être évalués sur la programmation et n'ont donc pas encore fait de devoirs.

Nous avons ensuite proposé deux méthodes pour enseigner la programmation. La première était de mettre des petits programmes sur Pronote (ou autre selon ce qui est utilisé dans l'établissement) que les élèves pourront utiliser pour réaliser des programmes plus complexes. Les avis à propos de celle-ci sont plutôt mitigés, en effet, certains pensent que cela pourrait être une méthode possible tandis que d'autres pensent que ce n'est pas réalisable à cause du manque d'autonomie de beaucoup d'élèves et le manque de suivi. Ils étaient cependant beaucoup plus favorables à la deuxième méthode qui était de laisser un élève arriver en classe avec un programme sur clé USB (qui aurait été donné en exercice) et de le montrer au reste de la classe afin d'en discuter. En effet, cela permettrait, selon les enseignants interrogés, de favoriser la prise de parole ainsi que la compréhension puisqu'une explication donnée par un élève peut sembler plus accessible que lorsqu'elle est donnée par le professeur.

Nous avons terminé ce questionnaire sur les difficultés rencontrées lors de l'enseignement de la programmation. Les principales difficultés rencontrées sont le manque de temps pour réaliser des séances de TP, ce qui fait que les élèves oublient tout d'une séance à l'autre. Nous avions déjà retrouvé ce type de remarques dans les réponses données par les élèves. Les autres difficultés sont au niveau du matériel, il n'est pas toujours évident d'avoir accès à une salle informatique et l'utilisation de la calculatrice est exigeante en rigueur et peu pratique ce qui est décourageant pour les élèves.

III- Analyse des données

Nous avons maintenant présenté l'ensemble des données recueillies au cours de l'élaboration de ce mémoire. Nous pouvons donc procéder à leur analyse selon les modalités exposées précédemment. Nous commencerons avec la motivation des élèves puis nous analyserons la question du sens des apprentissages pour finir par l'analyse des différentes pratiques enseignantes en nous appuyant sur la didactique de l'informatique.

1) La motivation des élèves

Nous avons précédemment parlé de l'importance de la motivation dans les apprentissages. En utilisant les différents indicateurs relevés dans le cadre théorique, nous allons pouvoir analyser la motivation des élèves dans le but de répondre aux questions de recherche. Nous pouvons pour cela nous appuyer sur les observations des différentes séances réalisées au cours de cette année ainsi que sur les réponses obtenues aux questionnaires remplis par les élèves et par les enseignants.

En regardant les réponses obtenues au questionnaire destiné aux élèves, nous avons pu constater que 63.2 % des élèves sont intéressés par la programmation. Nous pouvons en déduire que les élèves motivés le sont intrinsèquement puisqu'ils montrent de l'intérêt dans le fait de programmer. Mais nous pouvons voir qu'ils sont également motivés extrinsèquement puisque les raisons données de leur motivation sont l'utilisation de l'informatique, le côté ludique et le changement par rapport aux mathématiques plus classiques. Les enseignants ont également relevé ces raisons dans leurs questionnaires. Cela montre bien que les élèves sont davantage motivés par des facteurs extérieurs que par la programmation en elle-même. Nous avons également pu observer que les élèves demandent de faire de la programmation dans le but de réussir aux examens de fin d'année. Ils sont donc motivés positivement puisqu'ils veulent programmer et progresser dans le but d'obtenir une bonne note à leur examen ou à leur contrôle. Ils sont donc motivés par la programmation dans le but d'obtenir une récompense pour le travail qu'ils ont réalisé. Dans la théorie béhavioriste, on va utiliser la note pour motiver les élèves.

Les élèves motivés le sont aussi bien par les finalités que par les moyens, en effet, nous pouvons voir que certains élèves restent motivés par cela dans le but de réussir à programmer et de réussir aux examens, ces derniers vont donc tout faire pour réussir à atteindre ce but. D'autres vont en revanche être motivés uniquement par les moyens puisque la seule chose qui les motive dans la programmation est l'utilisation de l'informatique ou le côté ludique et non la satisfaction de réussir à le faire. Nous avons pu constater en classe que les élèves sont très demandeurs de séances de TP donc sur ordinateurs. Ils préfèrent réaliser des séances de TP et sont plus motivés lors de celles-ci plutôt qu'en faisant des exercices en classe. Nous avons relevé dans le cadre théorique que la réalisation d'un projet pouvait permettre aux élèves d'être motivés par les finalités tout en se donnant les moyens de réussir. Nous avons donc interrogé les élèves et les

enseignants à ce propos. Environ la moitié des élèves seraient intéressés par cette méthode d'apprentissage dans le but d'apprendre de nouvelles choses par eux-mêmes de manière ludique. Les enseignants, quant à eux, pensent que cela pourrait être utile d'en réaliser, que les élèves pourraient s'y investir et être motivés puisque cela pourra être autour d'un sujet concret et qui les intéresse. Nous pouvons également donner une note à la fin de ce projet, qui peut être un moyen de motiver davantage les élèves.

Plusieurs élèves ont répondu qu'ils restaient motivés puisque même s'ils n'arrivent pas à programmer seuls, ils pensent pouvoir y arriver avec de l'aide et un peu de temps. Nous pouvons donc constater que ces élèves se sont donnés pour « but idéal » de réussir à programmer seul. C'est ce que nous avons appelé le niveau d'aspiration. Cependant, cela risque d'être compliqué pour certains et ils vont donc devoir se fixer un but plus réaliste dans un premier temps. Cela peut être, d'abord, compléter et comprendre des programmes. Nous avons essayé, durant cette année, de nous situer dans la « Zone Proximale de Développement » (ZPD) décrite par Vygotski c'est-à-dire de proposer des exercices permettant aux élèves de réussir des nouvelles choses avec de l'aide sans être trop compliqué pour eux. En revanche, certains élèves qui ont plus de difficultés peuvent avoir du mal à suivre au même rythme que le reste de la classe et ont perdus cette motivation. Le but à atteindre qui a été fixé était trop éloigné de ce qu'ils pouvaient réellement faire. On a également parlé dans le cadre théorique de la motivation en relation avec les succès et les échecs. Lorsqu'un élève ne réussit pas à faire quelque chose malgré les efforts qu'il a fourni, il perd sa motivation. C'est le cas ici de plusieurs élèves qui ont perdu leur motivation à cause des difficultés qu'ils ont rencontrées. Cela peut également lié à l'estime de soi, en effet, s'ils ne se sentent pas capable de programmer, ils ne vont pas être motivés pour le faire.

Des raisons qui ont été données par les élèves qui ne sont pas motivés et qui ne réussissent pas sont le manque de temps et de séance de TP. Ils pensent que les séances sont trop éloignées les unes des autres ce qui fait qu'ils oublient les différentes notions d'une séance à l'autre. Il s'agit donc d'une cause externe.

Nous avons pu constater dans le questionnaire des élèves que beaucoup d'entre eux ne font pas de programmation chez eux puisqu'ils n'utilisent aucun logiciel chez eux et en font uniquement en classe. Cela montre que les élèves ne sont pas motivés par cela et n'ont pas envie d'en faire en dehors des cours. Il faudrait donc trouver des moyens pour leur donner envie d'en faire davantage.

2) Sens des apprentissages

Nous avons vu précédemment que pour que les élèves s'investissent dans les apprentissages, il fallait qu'il y ait du sens. En effet, s'ils ne connaissent pas l'utilité de ce qu'ils apprennent et s'ils ne savent pas à quoi cela va leur servir plus tard, ils ne vont pas être motivés. Dans les questionnaires remplis par les élèves, on peut constater que 68.4 % pensent que la programmation est utile. Cela montre leur investissement dans son apprentissage.

Au cours des différentes observations faites, nous avons pu voir que la plupart des élèves étaient ce qu'on a appelé dans le cadre théorique des « élèves récepteurs ». En effet, certains élèves sont très demandeurs de séances de TP pour apprendre à faire de la programmation et pour manipuler le logiciel dans le but de s'approprier ce langage qui n'est pas évident lorsque l'on commence. Ils posent des questions et vont chercher à comprendre comment cela fonctionne. Nous avons pu constater qu'ils travaillaient cela en classe mais à la maison, ils ne s'exercent pas du tout, puisque certains ont répondu au questionnaire qu'ils n'utilisaient aucun logiciel chez eux.

Les élèves ont été interrogé sur ce qu'ils ont compris et sont capables de réaliser seuls en fin d'année. Nous pouvons voir que les niveaux d'acquisition des différentes notions sont très variables selon les élèves. Les stratégies utilisées en classe ne correspondent peut-être pas à tous les élèves. Nous allons donc analyser les différentes pratiques enseignantes qui sont réalisées en classe et celles qui peuvent être envisagées que nous avons proposé dans les deux questionnaires.

3) La didactique de l'informatique

Nous avons pu, au cours de cette enquête, relever un certain nombre d'éléments au sujet des pratiques enseignantes qui sont réalisées ou qui sont envisageables. Nous pouvons donc nous appuyer sur les éléments théoriques relevés dans la première partie de ce mémoire et nous pouvons également comparer les avis des élèves à ceux des enseignants à ce propos. Nous commençons donc par analyser les différents logiciels qui sont utilisés, puis nous nous intéressons à la question de la transposition didactique et des différents contenus et nous finissons par étudier les différentes méthodes possibles pour enseigner la programmation en Python.

a) Les logiciels

Nous avons relevé dans le cadre théorique les spécificités de l'enseignement de la programmation, notamment les difficultés qui peuvent être rencontrées par rapport aux logiciels et au matériel utilisés. Nous avons conclu à ce sujet, qu'il faudrait réussir à trouver un logiciel adapté qui puisse être utilisé par tout le monde. Nous avons donc interrogé les élèves et les enseignants à ce sujet afin de déterminer le logiciel qui semble le plus adapté à cette utilisation.

Dans les réponses données, celui qui revient le plus souvent est EduPython qui est aussi bien utilisé en classe qu'à la maison. Ce logiciel est simple d'accès et très abordable pour les élèves. A partir du moment où on a accès à un ordinateur où ce logiciel a été installé, nous pouvons programmer en Python simplement et il permet de voir où sont les erreurs liées au langage dans le programme. Cependant, les élèves n'ont pas accès à un ordinateur en classe (sauf dans quelques établissements ou régions) et pour que chaque élève puisse manipuler le logiciel, il faut avoir accès à une salle informatique ce qui n'est pas toujours facile. Les dernières calculatrices sorties sur le marché proposent maintenant des modules permettant de programmer en Python ce qui pourrait faciliter son enseignement. Nous identifions tout de même quelques freins à l'utilisation de la calculatrice, en effet, il faut déjà que tous les élèves soient équipés de ce type de modèle et cela demande beaucoup de rigueur de la part des élèves pour programmer avec. L'utilisation de la calculatrice n'est pas très intuitive et prend beaucoup de temps ce qui est difficilement réalisable en classe ou très rarement.

Pour que les élèves puissent programmer en Python de chez eux, ils peuvent également utiliser des sites internet qui proposent de le faire, cela peut leur éviter de télécharger un logiciel mais c'est moins évident à utiliser et plus difficile de retrouver des programmes qui ont déjà été fait. Les élèves peuvent également utiliser des applications sur Smartphone mais cela est moins pratique que les autres outils.

D'après toutes ces données, nous pouvons conclure que le logiciel qui semble le plus adapté à l'enseignement de la programmation en langage Python est EduPython.

b) Les typologies de l'enseignement de l'informatique

Nous avions différencié six typologies de l'enseignement de l'informatique différentes décrits par Janine Rogalski. Dans notre étude, nous n'avons observé seulement trois de ces typologies. En effet, il s'agit ici d'informatique éducative, d'informatique outil puisque la programmation va être un outil pour aborder d'autres notions. Nous avons vu dans les programmes que l'algorithmique devait être réalisée tout au long de l'année au cours des différents chapitres au programme. La programmation va donc permettre aux élèves d'automatiser des calculs compliqués à réaliser à la main ou va permettre de mieux comprendre certaines notions. Il s'agit également d'une initiation préprofessionnelle puisque certains élèves vont avoir besoin de ces notions apprises au lycée pour continuer dans les études supérieures voir dans leur futur métier selon ce qu'ils vont choisir de faire.

c) La transposition didactique

Dans le cadre théorique, nous avons étudié les travaux de Fluckiger et de Viallet pour qui il était important de s'appuyer sur les contenus des enseignements et pour Viallet sur la transposition didactique. Pour rappel, la transposition didactique est l'ensemble des transformations qui permettent de passer des savoirs savants aux savoirs enseignés.

i) Les savoirs à enseigner

La première étape de la transposition didactique est la transposition didactique externe qui consiste à construire le « texte de savoir ». Elle va donc permettre de construire les programmes scolaires sur lesquels nous allons nous appuyer pour faire les cours. Nous avons, dans notre recueil de données, présenté les différents contenus dans les programmes de mathématique et de SNT des classes de seconde et de première générale. Ces derniers sont donc les savoirs que nous, enseignants, devons apporter aux élèves. Mais il va falloir rendre abordable ces savoirs aux élèves.

Nous avons relevé, pour la classe de seconde générale et technologique que les contenus à apporter sont les variables, les affectations, les séquences d'instructions, les instructions conditionnelles, les boucles bornées et non bornées et pour finir ainsi que les fonctions à un ou plusieurs arguments. Pour la classe de première générale, ces différentes notions doivent être

connues par les élèves et doivent être utilisées pour aborder la notion de listes. Pour enseigner la notion de boucles, nous pouvons reprendre la méthode énoncée par Viallet mais cela peut s'appliquer également aux autres notions au programme.

Nous nous sommes aperçus, en regardant les réponses des élèves et des enseignants aux questionnaires, que les notions qui sont le moins maîtrisées par les élèves sont les boucles et les listes. Il faudrait donc insister davantage sur ces notions et passer plus de temps dessus pour que les élèves réussissent à les utiliser.

La programmation étant une branche des mathématiques, nous pouvons reprendre les termes utilisés en didactique des mathématiques. Les notions présentées sont ce que l'on va appeler des **notions mathématiques**. En effet, ce sont des objets de savoirs bien identifiés qui sont construits et donnés par les enseignants aux élèves. Ces dernières vont ensuite pouvoir être évaluées. Lorsque ces notions seront connues par les élèves, elles pourront ensuite devenir des notions **proto mathématiques** c'est-à-dire qu'elles vont servir dans les apprentissages futurs.

Pour enseigner ces différents contenus, il va falloir trouver des méthodes pour les aborder avec les élèves de façon à ce qu'ils comprennent et qu'ils puissent apprendre à les utiliser progressivement.

ii) La programmabilité des savoirs

Cette programmabilité des savoirs va permettre d'organiser progressivement les apprentissages. Tout d'abord, nous avons vu dans les différents programmes étudiés que la programmation devait être enseignée au cours de toute l'année scolaire à travers les différents chapitres. Pour se faire, les élèves ainsi que les enseignants pensent qu'il faudrait réaliser des séances de Travaux Pratiques toutes les deux semaines et des exercices en classe, intégrés aux cours toutes les unes à deux semaines. Cela permettrait aux élèves de progresser dans ce domaine et de ne pas oublier les différentes notions d'une semaine à l'autre. Mais les enseignants rencontrent des difficultés pour réaliser aussi souvent des séances de TP, en effet, beaucoup d'entre eux n'ont pas le temps de les emmener en salle informatique aussi souvent sinon ils ne pourront pas terminer le programme de l'année. L'accès aux salles informatiques peut également être difficile dans certains établissements, ce qui ne permet pas forcément de faire beaucoup de séances de TP. La programmation en classe ne permet pas aux élèves de manipuler les logiciels

(sauf s'ils sont équipés d'ordinateurs ou s'ils ont une calculatrice permettant de le faire). Cette fréquence des séances est donc un idéal mais peut être ajusté selon les conditions dans lesquelles nous nous trouvons.

Les exercices proposés pour s'entraîner à la programmation sont généralement des exercices appliqués à des notions de mathématiques au programme, en effet, pour chaque notion décrite dans le programme, des exemples d'algorithmes sont proposés. Les élèves peuvent ainsi s'exercer à la programmation en suivant la progression de l'ensemble de l'année définie par l'enseignant.

Pour pouvoir réaliser de tels algorithmes et programmes, il faudrait tout de même commencer par faire des cours théoriques sur les nouvelles notions. Par exemple, avant de commencer à utiliser les listes dans des programmes pour répondre à des problèmes concrets, les élèves doivent connaître un minimum d'éléments sur cela. Ils doivent savoir ce qu'est une liste et avoir fait des programmes simples les utilisant avant d'en réaliser des plus complexes. C'est ce que nous avons essayé de faire avec la classe de première cette année. Nous avons en effet commencé par un cours théorique puis nous avons réalisé ou complété des programmes qui les utilisaient.

Pour les élèves, nous avons pu remarquer qu'il n'était pas évident d'écrire des programmes seul. Il faudrait donc procéder par étape, en commençant par exemple par comprendre un programme déjà écrit, et essayer de reproduire son cheminement à la main. On pourrait ensuite donner seulement une partie du programme et ils pourraient le compléter ou encore leur donner un programme à modifier. Toutes ces étapes pourraient leur permettre de finalement réussir à réaliser un programme seul.

On a pu constater dans les réponses données par les élèves que beaucoup d'entre eux n'étaient pas capables de réaliser des programmes seuls par contre avec de l'aide beaucoup comprennent comment cela fonctionne. En procédant par étape, cela pourrait leur permettre, par la suite de programmer seul ou en demandant de l'aide par moment.

Nous pouvons également reprendre la méthode énoncée par Viallet dans son mémoire en commençant par présenter le langage de programmation (ici, il s'agit de Python) ainsi que la notion à aborder. Ensuite, il faut décomposer le problème en plusieurs parties et traiter chacune d'elle une à une. Le but étant de réaliser un algorithme sur papier qui puisse être retranscrit dans n'importe quel langage. D'après les enseignants interrogés, il est important de commencer d'abord par réaliser un programme sur papier avant de le programmer en Python. Cela pourrait

permettre aux élèves de comprendre la logique de l'algorithmique sans se soucier des problèmes de langages. La transcription en Python ne pourrait se faire qu'après cette étape.

Le langage Python n'étant pas évident à utiliser pour débuter en programmation, on peut envisager de commencer cela en utilisant un langage plus simple, qui va servir d'intermédiaire entre Scratch, vu au collège, et Python. Nous avons constaté que la moitié des enseignants étaient favorables à cette idée. Cela permettrait, dans un premier temps, de comprendre les nouvelles notions apportées en utilisant un langage simple avant de le faire en Python. Nous risquons tout de même rencontrer des difficultés au niveau du temps, puisque nous passerions donc moins de temps sur Python mais si cela peut permettre de comprendre plus facilement, cela pourrait être utile.

4) Méthodes proposées pour enseigner la programmation en Python

Nous avons, au cours de l'année, utilisé différentes méthodes pour enseigner la programmation en langage Python. Nous avons, dans les questionnaires, proposé aux élèves ainsi qu'au enseignants, des méthodes qui pourraient éventuellement être envisageables afin de recueillir leur avis. Nous allons donc faire un point sur ces différentes méthodes pour savoir si elles peuvent être adoptées ou non selon les points de vus obtenus et éventuellement comment elles peuvent être améliorées.

a) L'élaboration d'un projet

Dans le cadre théorique, nous avons parlé de l'élaboration d'un projet avec les élèves qui leur permettrait d'être motivés par les finalités tout en se donnant les moyens de réussir. Les élèves pourraient ainsi apprendre des choses par eux-mêmes tout en ayant un but à atteindre. Nous avons donc interrogé les élèves et les enseignants pour avoir leur avis sur cela et sur la manière de procéder pour son élaboration.

Parmi les élèves interrogés, seule la moitié d'entre eux seraient intéressés par l'élaboration d'un projet de programmation. Cela peut peut-être s'expliquer par le fait qu'ils n'ont pas l'habitude d'une telle méthode d'enseignement et qu'ils manquent d'autonomie pour en réaliser un. Ils préfèrent donc réaliser des cours ou des TP plus classiques. Mais si l'on compare cela aux réponses de l'enseignant qui en a déjà réalisé, on peut constater que ses élèves ont apprécié ce

type d'enseignement. Pour les autres enseignants qui en n'ont pas réalisés, ils pensent que cela pourrait permettre aux élèves de s'investir et d'être motivés par la programmation. Cela leur permettrait donc d'apprendre de nouvelles choses par eux-mêmes de manière ludique tout en étant accompagné par l'enseignant.

Cependant certains enseignants ont identifié quelques freins à ce type d'enseignement. En effet, ils pensent que cela peut être difficile à organiser et ils manquent de temps pour le faire. Il ne faudrait donc pas qu'il dure trop longtemps. Le projet organisé par l'enseignant a duré environ un mois. Cela pourrait être une durée convenable. De plus l'objectif final ne serait pas trop éloigné et donc permettrait aux élèves de s'investir et de ne pas se décourager avant la fin de ce dernier. Pour réaliser un tel projet, il faudrait d'abord que les élèves suivent des cours théoriques sur les bases de la programmation pour pouvoir ensuite s'en servir seul. En effet, s'ils ne sont pas capables de réaliser ne serait-ce que des petits programmes simples, un projet n'aurait pas de réel intérêt.

Il faudrait, pour réaliser un projet, durant toute sa durée, avoir accès à une salle de TP régulièrement pour qu'ils puissent avancer et l'enseignant pourra les accompagner dans son élaboration pour débloquer des erreurs qui peuvent être commises dans le langage ou s'ils sont bloqués. Des groupes de 4 ou 5 élèves semblent le plus adapté pour qu'il y ait assez d'idées différentes et pour que chacun puisse apporter des connaissances au groupe. Les élèves pourraient s'entraider et expliquer des notions aux autres. Cela permettrait également à l'enseignant de les encadrer assez facilement de façon à ce qu'ils ne se dispersent pas trop.

Un tel projet pourrait également se faire en collaboration avec un enseignant de SNT ou de physique comme nous l'a suggéré un enseignant afin de programmer un automate. Cela peut être avec une carte Arduino qui est simple d'utilisation à partir du moment où l'on a fait un peu de programmation en Python. Les élèves pourraient ainsi appliquer ce qu'ils ont appris dans un cas concret et comprendre à quoi peut servir la programmation ce qui permettra de donner du sens à ces apprentissages et d'être motivés.

b) Les différentes méthodes proposées

Dans les questionnaires, nous avions demandé l'avis aux élèves et aux enseignants à propos de deux méthodes qui pourraient éventuellement être envisagées pour enseigner la programmation en langage Python.

La première était de mettre des petits programmes sur Pronote (ou autre logiciel utilisé en classe) pour que les élèves puissent les récupérer facilement, les comprendre et les modifier ou les utiliser dans d'autres programmes plus complexes. Les élèves étaient globalement favorables à cette méthode mais du côté des enseignants, c'était plus mitigé. En effet, d'après environ la moitié des enseignants, cela ne serait pas possible à cause du manque d'autonomie des élèves et du manque de suivi. On pourrait donc essayer de contrer ces problèmes en mettant tout de même des programmes de façon à ce que les élèves y aient accès et puissent y réfléchir de chez eux en essayant de les exécuter et de les comprendre. Ces mêmes programmes pourraient être repris et expliqués en classe en interrogeant les élèves sur ce qu'ils ont compris. Une fois que ces programmes auront été compris, l'enseignant pourra ensuite leur demander de les modifier ou de les réutiliser dans des programmes plus complexes. Les élèves ne seraient donc pas en autonomie pour les réaliser mais pourront tout de même les travailler de chez eux en aillant pour exemple des programmes qui fonctionnent. L'enseignant pourrait également mettre les programmes fait en cours sur Pronote pour que les élèves puissent les reprendre par la suite et puissent les tester même s'ils n'ont pas pu le faire en classe. Cela leur permettrait de voir le résultat final et pourrait les intéresser de voir qu'on obtient le bon résultat. Cela pourrait donc les motiver à continuer la programmation. L'enseignant pourrait ainsi faire de la programmation en cours avec eux sans forcément les emmener en salle de TP.

La seconde méthode que nous avions proposée était de donner un exercice de programmation aux élèves, à faire à la maison, et qu'ils arrivent en classe avec le programme sur clé USB. Le but serait qu'un élève montre son programme à la classe, même si celui-ci ne fonctionne pas et que la classe puisse en discuter. L'élève au tableau pourrait alors expliquer ce qu'il a fait, on a pu constater que certains élèves comprenaient mieux certaines notions lorsqu'elles étaient expliquées par l'un de leur camarade. Si l'on remarque que celui-ci ne fonctionne pas, le reste de la classe peut éventuellement apporter des modifications à celui-ci pour qu'il puisse être exécuté ou il peut même être amélioré pour réaliser d'autres choses. Pour cette méthode, les enseignants étaient globalement favorables à cette idée puisque cela permettrait aux élèves de s'exprimer à l'oral et de comprendre des programmes. Nous pouvons donc imaginer utiliser cette méthode pour enseigner la programmation même si les élèves ne sont pas tous favorables à cette idée puisque certains ne se sentent pas capables de le faire. Les élèves ne sont pas habitués à cela et peut donc leur faire peur, il faudrait y aller progressivement avec des programmes simples au départ pour leur donner confiance et pour les motiver à le faire.

Nous avons ainsi analysé l'ensemble des données que nous avons recueilli dans le but de répondre aux différentes questions de recherches. Nous allons donc pouvoir, à partir de cela, y répondre dans la partie discussion et nous pourrons également discuter des limites de ces recherches.

IV- Discussion

a) Réponses aux questions de recherche

Nous avions commencé par nous demander quels sont les savoirs à mobiliser pour programmer en Python. Nous avons donc réalisé une analyse des contenus dans les programmes des différentes classes qui nous permis de mener cette recherche. Nous avons différencié les contenus de la classe de seconde générale et technologique qui sont les variables, les affectations, les instructions conditionnelles, les boucles bornées et non bornées et enfin les fonctions et ceux de la classe de première générale qui sont la notion de listes et l'ensemble de leurs utilisations. Ces différents contenus permettent donc de commencer à programmer en langage Python et vont pouvoir servir pour résoudre des problèmes complexes en mathématiques ou dans d'autres matières comme les SNT, la physique ou l'économie.

Nous nous sommes ensuite interrogés sur les méthodes qui pourraient être utilisées pour enseigner la programmation en langage Python et plus particulièrement de la programmabilité des savoirs. D'après les données que nous avons recueillies, la programmation doit être réalisée en classe tout au long. L'idéal serait de faire une séance de TP toutes les deux semaines même si cela peut être compliqué à cause du manque de temps. Il faudrait également réaliser des exercices en classe toutes les unes à deux semaines. Cela permettrait en effet aux élèves de s'exercer et de manipuler le logiciel régulièrement sans oublier les différentes notions d'une séance à l'autre. Il faudrait donc apporter les savoirs progressivement en commençant pas des notions simples et en allant à celles qui sont plus compliquées. Les programmes vus en classes pourront être, dans un premier temps seulement des programmes à réaliser à la main afin de les comprendre, nous pourrons ensuite donner des programmes à compléter ou à modifier et pour finir, les élèves pourront écrire des programmes seuls. Avant d'écrire un programme en langage Python, il semble important de l'écrire sur papier en langage naturel pour comprendre la démarche puis de le retranscrire dans le langage souhaité. Nous pouvons également envisager

de commencer par un langage plus abordable que Python tel qu'AlgoBox pour commencer la programmation pour qu'ils comprennent dans un premier temps la logique de l'algorithmique.

Pour enseigner la programmation en Python, le logiciel qui semble le plus adapté est EduPython, il est simple d'usage et permet de voir où sont les erreurs commises. Les élèves peuvent y avoir accès facilement de chez eux et depuis les ordinateurs mis à leur disposition dans leur établissement. La calculatrice peut également être un outil pour l'enseigner lorsque l'on n'a pas accès à une salle informatique mais les élèves peuvent rencontrer des difficultés pour cela notamment au niveau de la rigueur demandée par ce type de matériel.

Nous avons étudié quelques méthodes qui peuvent être utilisées pour enseigner la programmation en Python. On peut par exemple élaborer un projet en classe par groupe de 4 ou 5 élèves en les accompagnant dans sa création. Il faudrait pour cela avoir accès à une salle informatique. Avant de réaliser un tel projet, il faudrait tout de même apporter quelques cours théoriques pour que les élèves apprennent les bases de la programmation. Ils pourraient ensuite commencer ce projet en posant des questions à l'enseignant s'ils rencontrent des difficultés. Ce projet ne durerait donc pas toute l'année mais peut durer seulement quelques semaines. Les élèves auraient ainsi un objectif à atteindre ce qui permettrait de donner du sens à ce qu'ils ont appris jusqu'à présent et cela peut donc les motiver. Les élèves pourraient également être notés à la fin ce qui leur donnerai une motivation supplémentaire pour s'investir dans celui-ci.

Une seconde méthode qui pourrait être utilisée serait de déposer sur Pronote (ou autre) des programmes pour que les élèves puissent les récupérer, les comprendre et les tester de chez eux. Il faudrait tout de même les retravailler en classe pour qu'ils puissent les comprendre correctement et ces derniers pourraient servir d'appui pour réaliser d'autres programmes plus complexes.

Nous pouvons également envisager de demander aux élèves de réaliser un programme chez eux et de le mettre sur clé USB afin de le montrer à l'ensemble de la classe. Ce programme pourrait être expliqué au reste de la classe, modifié si nous remarquons des erreurs et peut être amélioré pour réaliser d'autres choses. Cela servirait d'appui pour une discussion en classe et pourrait donc remotiver les élèves qui ne le sont pas ou plus.

Nous nous étions posé la question de l'évaluation de la programmation. L'évaluation de cette matière peut se faire au travers de devoirs sur d'autres notions qui comporteraient des questions ou exercice de programmation. Cela pourrait également se faire en séance de TP ou sous forme de devoir maison.

Nous nous demandions avant de réaliser cette recherche si les élèves étaient motivés par la programmation ou non et comment nous pourrions faire pour qu'ils le deviennent. Nous avons pu constater que certains élèves le sont pour l'utilisation de l'informatique et pour le côté ludique mais cela ne représente qu'une minorité des élèves. Beaucoup d'entre eux trouvent cela trop compliqués ou ils n'aiment pas cela et ne sont donc plus motivés. Les méthodes proposées précédemment pourraient permettre de remotiver les élèves en leur redonnant confiance et en rendant cet enseignement plus ludique. La note peut également leur permettre de les motiver davantage.

Toutes ceci devrait pouvoir aider un enseignant qui enseigne la programmation en essayant des méthodes différentes et en trouvant celle qui correspond le mieux à son public.

b) Limites de cette recherche

Lors de cette recherche, nous n'avons pas pu tester les différentes méthodes proposées avec l'une des classes en raison des circonstances sanitaires actuelles. Nous pouvons donc penser que celles-ci peuvent être appropriées aux vu des éléments théoriques que nous avons recueillis et des différents questionnaires remplis par les élèves et les enseignants mais nous n'en avons pas la certitude.

Dans certaines régions ou dans certains établissements, chaque élève a son propre ordinateur portable, ce qui n'est pas notre cas ici, les contraintes liées au matériel exposées précédemment n'ont donc pas lieu dans ce type d'établissement.

Nous avons exposé des méthodes possibles mais celles-ci ne sont pas les seules, il en existe une multitude d'autres. Au niveau de l'évaluation, nous n'avons pas pu faire plus de recherches et nous avons donc très peu d'éléments à ce sujet. Il faudrait donc continuer les recherches pour apporter des réponses sur cela.

Conclusion

En définitive, nous avons constaté que les professeurs pouvaient rencontrer des difficultés pour enseigner la programmation en langage Python et les élèves n'étaient pas forcément motivés pour en faire.

L'objectif de se mémoire était de chercher des moyens d'enseigner la programmation qui soient réutilisés en classe. Nous voulions en trouver qui permettent aux enseignants d'accompagner les élèves en leur apportant de nouvelles connaissances de manière ludique et qui permettent aux élèves d'être motivés.

Nous avons donc relevé des éléments théoriques et nous avons recueilli des donnés qui nous ont permis d'imaginer des méthodes qui pourraient convenir à la fois aux enseignants et aux élèves pour qu'ils réussissent à apprendre de nouvelles notions tout en étant motivé et en s'investissant dans les apprentissages. Les trois méthodes que nous avons proposées sont :

- L'élaboration d'un projet
- Laisser à disposition des élèves des programmes en Python exécutables directement qu'ils pourront comprendre, tester, réutiliser et travailler en classe
- Demander aux élèves de venir en classe avec un programme à montrer à la classe et qui servirait d'appui pour en discuter.

Ces différentes méthodes permettraient à la fois de travailler la programmation en langage Python avec la classe de façon ludique tout en favorisant la coopération entre les élèves. Cela permettrait donc de les motiver afin qu'ils s'investissent dans les apprentissages.

Ces différentes méthodes pourront ainsi être utilisées pour faciliter l'enseignement de la programmation. Cependant, ce ne sont pas les seules possibles et ne vont peut-être pas convenir à certaines classes ou certains enseignants. Il est possible qu'elles aient besoin d'être adaptées à la situation et aux élèves.

Dans ce mémoire, nous n'avons pas eu le temps de nous intéresser à l'évaluation de la programmation. Il s'agit d'une partie importante de son enseignement puisque les élèves vont en avoir dans leurs examens, il est donc utile de les préparer à cela. Il serait donc intéressant de mener des recherches dans ce sens.

BIBLIOGRAPHIE :

- Académie des Sciences. (2013). *L'enseignement de l'informatique en France*. Consulté à l'adresse https://www.academie-sciences.fr/pdf/rapport/rads_0513.pdf
- André, J. (1992). A l'origine, la relation humaine. *Cahiers pédagogiques*, (300).
- Archambault, J., & Chouinard, R. (1996). *Vers une gestion éducative de la classe*. Montréal : Gaëtan Morin.
- Bandura, A. (1986). *Social foundations of thought and action : a social cognitive theory*. New York : Prentice-Hall.
- BARDIN, Laurence (1993). *L'analyse de contenu*. 7e éd. Corrigée. Paris : PUF. 291 p.
- Brousseau, G. (2004). *Recherches en éducation mathématiques,* APMEP.
- Chevallard, Y. (1985). *Pourquoi la transposition didactique ?* In Y. Chevallard (Ed), *La transposition didactique, du savoir savant au savoir enseigné.* Grenoble : La pensée sauvage.
- Chevallard, Y. (1991). *La transposition didactique, du savoir savant au savoir enseigné.* Grenoble : La pensée sauvage.
- CNRTL. (s. d.). MOTIVATION : Définition de MOTIVATION. Consulté le 9 janvier 2020, à l'adresse https://www.cnrtl.fr/definition/Motivation
- Cordier, A. (2015). *Grandir connectés, Les adolescents et la recherche d'information*. France : C&F.
- Covington, M. V., & Teel, K. M. (2000). *Vaincre l'échec scolaire*. Bruxelles : De Broeck.
- Crahay, M. (1996). *Peut-on lutter contre l'échec scolaire ?* Bruxelles : De Broeck.
- De Vecchi, G., & Carmona-Magnaldi, N. (2004). *Faire vivre de véritables situations-problèmes* (Profession enseignant). France : Hachette Éducation.
- Fenouillet, F. (2003). *La motivation*. Paris : Dunod.
- Fluckiger, C. (2019). *Une approche didactique de l'informatique scolaire*. Lille : PU Rennes.
- Houssaye, J. (1993). *La pédagogie, une encyclopédie pour aujourd'hui (Collection Pédagogies) (French Edition)*. Paris : ESF.
- Maslow, A. (1943). A theory of human motivation. *Psychological Review*, *50*(4), 370-396.
- Meirieu, P. (1995). *Apprendre…oui, mais comment ?* (15e éd.). Paris : ESF.

- MUCCHIELLI, Roger (1998). *L'analyse de contenu des documents et des communications*. Paris : ESF éditeur, 214 p. [1974]
- Nachon, M., & Wallian, N. (2005). Les " sports co " motivent ? *Cahiers pédagogiques*, (429-430).
- Natanson, D., & Berthou, M. (2013). *Jouer en classe en collège et en lycée*. h, France : Fabert.
- Paquelin, D. (2004). *Le tutorat : accompagnement de l'actualisation du dispositif.* Distances et savoirs, 2, 157-182. En ligne https://www.cairn.info/revue-distances-et-savoirs-2004-2- page-157.htm
- Perrenoud, P. (2005). *Sens du travail et travail du sens à l'école.* Cahiers pédagogiques, 429-430. En ligne http://www.cahiers-pedagogiques.com/Sens-du-travail-et-travail-du-sens-a-l-ecole
- Perrez, M., Minsel, B. & Wimmer, H. (1990). *Ce que les parents devraient savoir : une école psychologique pour les parents, enseignants et éducateurs*. Belgique : Éd. Labor.
- *Programme de mathématique de première générale*. (2019). France. Consulté à l'adresse https://cache.media.education.gouv.fr/file/SP1-MEN-22-1-2019/16/8/spe632_annexe_1063168.pdf
- *Programme de mathématiques de seconde générale et technologique*. (2019). Consulté à l'adresse https://cache.media.education.gouv.fr/file/SP1-MEN-22-1-2019/95/7/spe631_annexe_1062957.pdf
- *Programme de sciences numériques et technologie de seconde générale et technologique*. (2019). Consulté à l'adresse https://cache.media.education.gouv.fr/file/SP1-MEN-22-1-2019/08/5/spe641_annexe_1063085.pdf
- Reuter, Y. (2007/2013). *Dictionnaire des concepts fondamentaux des didactiques* (3e éd.). Bruxelles, Belgique : De Boeck.
- Rey, A. (2000). *Dictionnaire historique de la langue française.* 3e éd. Paris : Le Robert
- Rogalski, J. (1988b). *Didactique de l'Informatique et acquisition de la programmation. Recherche en didactique des mathématiques*, 9(3), 407-426
- Rogers, C.R. (1984). *Liberté pour apprendre ?* Paris : Dunod.
- Roussel, P. (2000). *La motivation au travail - Concept et théories*. Toulouse : LIRHE.
- Sestier, D., & Hochet, Y. (2005). Jouer ou travailler : faut-il vraiment choisir ? *Cahiers pédagogiques*, (429-430).

- Tardif, J. (1992). *Pour un enseignement stratégique : L'apport de la psychologie cognitive (French Edition)*. Montréal : Logiques Editions.
- Unescopresse. (2002, mai 31). UNESCO - COMMENT PRESERVER L'INFORMATION NUMERIQUE ? Consulté le 6 novembre 2019, à l'adresse http://portal.unesco.org/fr/ev.php-URL_ID=4805&URL_DO=DO_TOPIC&URL_SECTION=201.html
- Verret, M. (1975). *Le temps des études.* Thèse d'état, Université de Paris V. Paris. Librairie Honoré Champion.
- Viallet, F. (2009). *Analyse transpositive de la boucle : Une contribution à la didactique de l'informatique*.
- Vianin, P. (2006). *La motivation scolaire : Comment susciter le désir d'apprendre ?* (1re éd.). Bruxelles : De Boeck.
- Viau, R. (1997). *La Motivation en contexte scolaire*. Bruxelles : De Boeck.
- Vygotsky, L. S. (1985). *Pensée et langage*. Paris : Sociales.
- Wikipedia contributors. (2020, janvier 10). Fichier:Maslow23.JPG — Wikipédia. Consulté le 10 janvier 2020, à l'adresse https://fr.wikipedia.org/wiki/Fichier:Maslow23.JPG

Annexe 1 : Questionnaire élève

Questionnaire Programmation en Python 1ereG

Bonjour,
Dans le cadre de mon MASTER, je dois réaliser un mémoire sur le thème de l'enseignement de la programmation en langage Python. Je vous demande donc de réaliser ce petit questionnaire qui me permettra d'analyser votre ressenti dans ce domaine ainsi que d'étudier les pratiques des enseignants.
Ce questionnaire est anonyme et je vous demande, par conséquent, de répondre honnêtement et sérieusement.
Merci d'avance
*Obligatoire

1. Etes-vous intéressé par la programmation ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Cela m'intéresse mais je n'y arrive pas
- ◯ Non

2. Etes-vous motivé par la programmation ? *

Une seule réponse possible.

- ◯ Oui *Passer à la question 6*
- ◯ Non *Passer à la question 3*

3. Pourquoi trouvez-vous que la programmation n'est pas motivante ? *

Plusieurs réponses possibles.

- [ ] J'étais motivé au début mais comme je n'y arrive pas, j'ai arrêté
- [ ] Je n'aime pas ça
- [ ] Je ne comprends pas à quoi ça sert
- [ ] Je trouve ça trop compliqué
- [ ] J'ai jamais réussi en maths donc la programmation c'est pareil
- [ ] Je ne comprends rien
- [ ] Je me sens nul dans cette matière
- [ ] C'est la faute du prof, il ne rend pas le cours intéressant
- [ ] Autre : _____

4. Étiez-vous à un moment motivé par la programmation ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non *Passer à la question 7*

5. Pourquoi vous ne l'êtes plus ?

Plusieurs réponses possibles.

- [ ] J'y arrivais pas donc j'ai arrêté
- [ ] Cela ne correspond pas à mes attentes
- [ ] Autre : _____

Passer à la question 7

6. Qu'est-ce qui vous motive le plus dans la programmation ? *

Plusieurs réponses possibles.

- [ ] Utilisation de l'informatique
- [ ] Cela change des maths "classiques"
- [ ] La programmation en elle-même
- [ ] Le côté ludique
- [ ] J'aime ça donc ça me motive à en faire
- [ ] Le prof donne envie d'en faire
- [ ] Je trouve ça facile
- [ ] J'y arrive pas encore mais je suis sûre qu'avec un peu d'aide, ça va venir
- [ ] Autre : _____

7. Quels logiciels utilisez-vous en classe ? *

Plusieurs réponses possibles.

- [ ] EduPython
- [ ] Python calculatrice
- [ ] Python en ligne
- [ ] Application sur Smartphone
- [ ] Autre : _____

8. Quels logiciels utilisez-vous à la maison ? *

Plusieurs réponses possibles.

- [ ] EduPython
- [ ] Python calculatrice
- [ ] Python en ligne
- [ ] Application sur Smartphone
- [ ] Aucun
- [ ] Autre : _____

9. Quels sont les outils qui vous semblent les plus adaptés pour programmer en Python en classe ? *

Plusieurs réponses possibles.

- [ ] Calculatrice
- [ ] EduPython sur ordinateur
- [ ] Application sur Smartphone

Autre : [ ] _____

10. Quels sont les outils qui vous semblent les plus adaptés pour programmer en Python à la maison ? *

Plusieurs réponses possibles.

- [ ] Calculatrice
- [ ] EduPython sur ordinateur
- [ ] Application sur Smartphone
- [ ] Python en ligne

Autre : [ ] _____

11. Que préférez-vous ? *

Une seule réponse possible.

- ( ) Faire des séances de TP
- ( ) Faire des exercices en classes incorporés au cours
- ( ) Faire un peu des deux

12. Que préférez-vous ? *

Une seule réponse possible.

- ( ) Travailler la programmation uniquement en classe
- ( ) Travailler la programmation en classe et à la maison sous forme d'exercices

13. A quelle fréquence voudriez-vous faire des TP ? *

Une seule réponse possible.

- ◯ Une fois par mois
- ◯ Une fois toutes les deux semaines
- ◯ Une fois par semaine
- ◯ Autre : _____

14. A quelle fréquence voudriez-vous faire des exercices de programmation en cours ? *

Une seule réponse possible.

- ◯ Une fois par mois
- ◯ Une fois toutes les deux semaines
- ◯ Une fois par semaine
- ◯ Deux fois par semaine
- ◯ Autre : _____

15. L'enseignement de la programmation vous semble-t-il utile ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non

16. Seriez-vous intéressé par l'apprentissage de la programmation sous forme de projet à réaliser tout au long de l'année ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non *Passer à la question 20*

17. Qu'est ce qui vous intéresse dans l'élaboration d'un projet ? *

Plusieurs réponses possibles.

- [ ] Travail en groupe
- [ ] L'occasion d'apprendre de nouvelles choses par soi-même
- [ ] C'est plus ludique que les cours "classiques"
- [ ] Autre : _____

18. Par groupe de combien voudriez-vous réaliser ce projet ? *

Une seule réponse possible.

1	2	3	4
◯	◯	◯	◯

19. Pour la réalisation d'un projet, pensez vous que des cours théoriques avant de démarrer seraient utiles ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non

Passer à la question 21

20. Pourquoi la réalisation d'un projet ne vous intéresse pas ? *

Plusieurs réponses possibles.

- [ ] Je n'aime pas travailler en groupe
- [ ] Je préfère faire des cours/TP classiques
- [ ] Autre : _____

21. Voudriez-vous avoir accès sur Pronote à des petits programmes que vous pouvez directement exécuter de chez vous pour mieux comprendre et pour pouvoir les utiliser dans des programmes plus complexes ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non

22. Voudriez-vous pouvoir arriver en classe avec un programme sur clé USB (sur un exercice demandé) et pouvoir en discuter avec le reste de la classe ? *

Une seule réponse possible.

- ◯ Oui ça pourrait être enrichissant pour tout le monde
- ◯ Cela me plairait mais je ne me sens pas capable de le faire
- ◯ Non

23. A la fin de cette année, pensez-vous être capable de réaliser seul des petits programmes ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non, mais je comprends les programmes qui sont faits en classe
- ◯ Non et je ne comprends pas les programmes qui sont faits en classe

24. Cochez les notions que vous pensez maintenant être capable d'utiliser seul

Plusieurs réponses possibles.

- ☐ Boucles For
- ☐ Boucles While
- ☐ Listes
- ☐ Fonctions
- ☐ Aucune

25. Cochez les notions que vous comprenez lorsque l'enseignant les explique mais que vous ne vous sentez pas capable d'utiliser seul *

Plusieurs réponses possibles.

- [ ] Boucles For
- [ ] Boucles While
- [ ] Listes
- [ ] Fonctions
- [ ] Aucune

26. Cochez les notions que vous ne comprenez pas du tout et que vous ne vous sentez pas capable de réaliser. *

Plusieurs réponses possibles.

- [ ] Boucles For
- [ ] Boucles While
- [ ] Listes
- [ ] Fonctions
- [ ] Aucune

27. A quoi la programmation vous a t-elle servi ? *

Plusieurs réponses possibles.

- [ ] A mieux comprendre certaines notions mathématiques
- [ ] A améliorer ma logique
- [ ] Faciliter certains calculs compliqués à réaliser à la main
- [ ] J'ai pas l'impression que cela me sert

Autre : [ ] _____

28. Si cela vous a permis de mieux comprendre certaines notions mathématiques, lesquelles ?

29. Avez-vous des remarques particulières ou des choses que vous voudriez qui vous permettraient de mieux réussir en programmation ?

Annexe 2 : Questionnaire enseignant

Questionnaire enseignement de la programmation en Python

Bonjour,
Dans le cadre de mon MASTER "Métiers de l'Enseignement, de l'Education et de la Formation", je réalise un mémoire sur le thème de l'enseignement de la programmation en Python. Je vous sollicite ainsi pour répondre à ce petit questionnaire qui me permettra de mettre en perspective des éléments théoriques et des éléments pratiques relatifs à cet enseignement.
Je vous remercie par avance pour votre aide et votre disponibilité.
*Obligatoire

1. Etes-vous enseignant de... *

Plusieurs réponses possibles.

- [ ] Maths
- [ ] SNT
- Autre : [ ] _____

2. Avec quelles classes faites vous de la programmation en Python ? *

Plusieurs réponses possibles.

- [ ] Seconde
- [ ] Première Générale
- [ ] Terminale S
- [ ] Première STAV
- [ ] Terminale STAV
- Autre : [ ] _____

3. Trouvez-vous que vos élèves sont motivés par la programmation ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non
- ◯ Ça dépend des classes
- ◯ Ça dépend des élèves

4. Selon vous, qu'est ce qui motive les élèves dans la programmation ? *

Plusieurs réponses possibles.

- ☐ Utilisation de l'informatique
- ☐ Changement par rapport aux maths "classiques"
- ☐ La programmation en elle-même
- ☐ Ils aiment cela
- ☐ Ils trouvent cela assez facile
- ☐ Ils n'y arrivent pas encore mais ils restent motivés puisqu'ils pensent pouvoir y arriver avec de l'aide
- Autre : ☐ _____

5. Selon vous, pourquoi certains élèves ne sont pas motivés par la programmation ?

Plusieurs réponses possibles.

- ☐ Ils trouvent cela trop compliqué et ont donc arrêté
- ☐ Ils n'aiment pas ça
- ☐ Ils en comprennent pas à quoi ça sert
- ☐ Ils se sentent "nul" en maths et donc pour eux en programmation, c'est la même chose
- ☐ Ils ne comprennent pas
- ☐ Ils étaient motivés au début mais cela ne correspond pas à leurs attentes donc ils ont arrêté de faire des efforts
- Autre : ☐ _____

6. Comment procédez-vous pour enseigner la programmation ? *

Plusieurs réponses possibles.

- [ ] Séances de TP
- [ ] Exercices en classes
- [ ] Exercices à faire à la maison
- [ ] Autre

7. A quelle fréquence faites vous des séances de TP ? *

Une seule réponse possible.

- ◯ Une fois par mois
- ◯ Une semaine sur deux
- ◯ Une fois par semaine
- ◯ Autre

8. A quelle fréquence faites vous des exercices (incorporés au cours) *

Une seule réponse possible.

- ◯ Une fois par mois
- ◯ Une semaine sur deux
- ◯ Une fois par semaine
- ◯ Autre : _____

9. Avez-vous déjà réalisé un projet de programmation avec l'une de vos classe ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non *Passer à la question 17*

10. Combien de temps a-t-il duré ? *

Une seule réponse possible.

- ◯ Toute l'année
- ◯ Un semestre
- ◯ Un trimestre
- ◯ Un mois
- ◯ Autre : _____

11. Les élèves étaient-ils seuls ou par groupe pour le réaliser ? *

Une seule réponse possible.

- ◯ Seul
- ◯ Par groupe de 2
- ◯ Par groupe de 3
- ◯ Par groupe de 4
- ◯ Autre : _____

12. Comment avez-vous organisé ce projet ? Fréquence des séances ? Avez vous réalisé des cours théoriques en amont ? Les élèves ont-ils choisi le sujet eux-mêmes ou venait-il de vous ? *

_____

_____

_____

_____

_____

13. Les élèves ont-ils apprécié cette méthode d'apprentissage ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non *Passer à la question 15*

14. Qu'ont-ils apprécié ?

Plusieurs réponses possibles.

- ☐ Travail en groupe
- ☐ Le côté ludique du projet
- ☐ Pouvoir apprendre des choses d'eux-même
- Autre : ☐ _____

Passer à la question 16

15. Qu'est-ce qu'ils n'ont pas apprécié ? *

Plusieurs réponses possibles.

- ☐ Travail en groupe
- ☐ Ils préfèrent réaliser des cours/TP classiques
- Autre : ☐ _____

16. Etes-vous prêt à en refaire un avec l'une de vos classes ? Pourquoi ? *

_____

_____

_____

_____

_____

Passer à la question 23

17. Quels sont les freins que vous identifiez dans la réalisation d'un projet ? *

Plusieurs réponses possibles.

- [ ] Difficile à organiser
- [ ] Manque de temps
- [ ] Les élèves ne sont pas réceptifs à ce type d'enseignement

Autre : [ ] _____

18. Qu'est-ce qui permettrait de faciliter un tel projet ?

Plusieurs réponses possibles.

- [ ] Faire régulièrement des séances de TP
- [ ] Accompagner les élèves dans son élaboration : aider individuellement les groupes
- [ ] Contact par mail qui permettrai aux élèves de pouvoir avancer entre les séances de TP

Autre : [ ] _____

19. Pensez-vous que cela pourrait permettre aux élèves d'être davantage motivés et qu'ils vont s'investir dans le projet ? *

Une seule réponse possible.

- ◯ Oui
- ◯ Non

20. Seriez-vous prêt à en faire avec vos classes ? *

Une seule réponse possible.

- ◯ Oui *Passer à la question 21*
- ◯ Non *Passer à la question 23*

21. Par groupe de combien pensez-vous que ce type de projet doit être fait ? Pourquoi ? *

_____

22. Cette organisation vous semble-t-elle appropriée : Réaliser quelques séance sur les bases de la programmation puis démarrer le projet en aidant individuellement chaque élève/groupe ? Si non, pourquoi et comment voudriez-vous faire ? *

23. Quels sont les logiciels que vous utilisez en classe ? *

Plusieurs réponses possibles.

- [ ] EduPython
- [ ] Python en ligne
- [ ] Python sur calculatrice
- [ ] Application du Smartphone

Autre : [ ] _____

24. Lequel vous semble le plus adapté ? *

Une seule réponse possible.

- ( ) EduPython
- ( ) Python en ligne
- ( ) Python sur calculatrice
- ( ) Application sur Smartphone
- ( ) Autre : _____

25. Pensez-vous qu'il faut... *

Une seule réponse possible.

- ◯ Commencer directement à programmer en Python
- ◯ Procéder par étape en commençant pas programmer avec un langage plus facile du type Algobox puis passer au langage Python

26. Pensez-vous qu'il vaut mieux pour les élèves de... *

Une seule réponse possible.

- ◯ Commencer par faire un algorithme sur papier puis le programmer en Python
- ◯ Directement programmer en Python

27. Quels sont les savoirs que vous enseignez ? *

Plusieurs réponses possibles.

- ☐ Boucle While
- ☐ Boucle For
- ☐ Fonctions
- ☐ Listes

Autre : ☐ _____

28. Quels sont les savoirs que la majorité de vos élèves sont capables d'utiliser en cette fin d'année ? *

Plusieurs réponses possibles.

- ☐ Boucle While
- ☐ Boucle For
- ☐ Fonctions
- ☐ Listes

29. Evaluez-vous vos élèves en programmation ? (Cela peut être simplement un programme à compléter dans un DM/DS sans avoir besoin de le programmer sur ordinateur) *

Une seule réponse possible.

- ◯ Oui
- ◯ Non *Passer à la question 31*

30. De quelle manière ? *

Plusieurs réponses possibles.

- ☐ Devoir maison
- ☐ Séance de TP
- ☐ Exercice ou question incorporé à un devoir de maths
- Autre : ☐ _____

Passer à la question 32

31. Pourquoi ? *

Plusieurs réponses possibles.

- ☐ Manque de temps
- ☐ Les élèves ne sont pas encore prêts pour cela
- Autre : ☐ _____

32. Pensez-vous que de mettre des petits programmes sur Pronote que les élèves pourront utiliser pour réaliser des programmes plus complexes pourrait être un moyen d'enseigner la programmation en Python ? Si non, pourquoi ? *

_____

_____

_____

_____

_____

33. Pensez-vous que de laisser un élève arriver avec un programme (donné en exercice) sur clé USB et de le montrer au reste de la classe afin d'en discuter serait un bon moyen pour que les élèves s'investissent ? Pourquoi ? *

34. Rencontrez-vous des difficultés pour enseigner la programmation en Python ? Si oui, lesquelles et qu'est-ce qui pourrait vous aidez à les surmonter ? *

Annexe 3 : Document distribué aux élèves lors de la séance sur les listes

Listes : Python

Définition :
Une liste est une collection ordonnée d'objets.
La liste contenant $C_0, C_1, \ldots, C_n$ est notée $[C_0, C_1, \ldots, C_n]$
Si L est une liste, l'élément d'indice i de L sera noté L[i]
Attention : Le premier indice est 0

Exemples :

```
>>> A=[1,5,8]
>>> A[1]
5
```

```
>>> B=["Lundi","Mardi","Mercredi","Jeudi"]
>>> B[0]
'Lundi'
```

```
>>> C=[True,False,True]
>>> C[-1]
True
```

```
>>> D=[]  #liste vide
```

Opération	Python	Exemples
Créer une liste L en extension	L=[variable 1, variable 2,...]	`>>> L=[5,12,-3,"python",8]`
Afficher une liste L	print(L)	`>>> print(L)` `[5, 12, -3, 'python', 8]`
Longueur d'une liste L	len(L)	`>>> len(L)` `5`
Afficher le $k^{ème}$ élément de L	print(L[k-1])	`>>> print(L[2])` `-3`
Liste des entiers compris entre deux entier a et b	list(range(a,b+1))	`>>> list(range(3,10))` `[3, 4, 5, 6, 7, 8, 9]`

Ajout de l'élément a à la fin de la liste L	L.append(a)	`>>> L.append("maths")`
		`>>> print(L)`
		`[5, 12, -3, 'python', 8, 'maths']`

Concaténation de deux liste L et L'	L + L'	`>>> L1=[3,4,7]`
		`>>> L2=[5,"python"]`
		`>>> L1+L2`
		`[3, 4, 7, 5, 'python']`

Suppression de l'élément d'indice i de la liste L	L.pop(i)	`>>> L=[2,6,9,"vert"]`
		`>>> L.pop(2)`
		`9`
		`>>> L`
		`[2, 6, 'vert']`

Suppression de la première occurrence de l'élément c dans la liste L	L.remove(c)	`>>> L.remove(6)`
		`>>> L`
		`[2, 'vert']`

Inverser les termes de L	L1=list(reversed(L))	`>>> L=[2,6,9,"vert"]`
		`>>> L1=list(reversed(L))`
		`>>> L1`
		`['vert', 9, 6, 2]`

Nombre d'occurrences d'une valeur a	L.count(a)	`>>> L.count(2)` `1`

Savoir si un terme a figure dans la liste L	print(a in L)	`>>> print(2 in L)`
		`True`
		`>>> print(12 in L)`
		`False`

Reproduire n fois les termes de la liste L	print(L*n)	`>>> L=[1,2]`
		`>>> L*3`
		`[1, 2, 1, 2, 1, 2]`

Ranger par ordre croissant les nombres de la liste L	L1=sorted(L)	`>>> L=[12,14,9,18,3,5]`
		`>>> print(sorted(L))`
		`[3, 5, 9, 12, 14, 18]`

Insérer au rang i le terme a dans la liste L	L.insert(i,a)	`>>> L.insert(3,15)`
		`>>> L`
		`[12, 14, 9, 15, 18, 3, 5]`

Insérer au rang i les termes de la liste L2 dans L	L[i : i] = L2	`>>> L[2:2]=[11,13,6]`
		`>>> L`
		`[12, 14, 11, 13, 6, 9, 15, 18, 3, 5]`

Afficher le type de l'élément de rang i de L	print(type(L[i]))	`>>> L=[2, 3.2, True, "python"]`
		`>>> print(type(L[0]), type(L[1]))`
		`<class 'int'> <class 'float'>`

Afficher le nombre le plus grand/petit de la liste L	print(max(L))
	print(min(L))
		`>>> L=[11,14,3,20,15,5,7]`
		`>>> print(max(L))`
		`20`
		`>>> print(min(L))`
		`3`

Afficher le rang du terme a dans la liste L	print(L .index(a))	`>>> L.index(20)` `3`

Liste formée des éléments f(x) pour x parcourant L	[f(x) for x in L]	`>>> L=[2,4,5]`
		`>>> [2*x for x in L]`
		`[4, 8, 10]`

Liste formée des éléments f(x) pour x parcourant L et vérifiant le test T	[f(x) for x in L if T]	`>>> [2*x for x in L if x%2 == 0]` `[4, 8]`

Liste formée des éléments f(k) pour k entier variant de a à b (de 0 à n-1)	[f(k) for k in range (a,b + 1)]
	[f(k) for k in range (n)]
		`>>> [3*k for k in range(3,8)]`
		`[9, 12, 15, 18, 21]`
		`>>> [2*k for k in range (6)]`
		`[0, 2, 4, 6, 8, 10]`

Exercice 1 :

Soit la liste A= [1, 5, 2, 3, 9]

a) Quelle est la valeur de A[0] ? De A[2] ?
b) Quelle est l'indice de l'élément 3 de cette liste ?
c) Quelle est la longueur de A ?
d) Quel est le plus grand élément de A ? Le plus petit ?
e) Ajouter l'entier 8 à la fin de A
f) Ajouter l'entier 2 au rang 1 dans la liste A

Exercice 2 :
On considère l'ensemble des carrés des entiers naturels inférieurs ou égaux à 20.
Construire une liste L de ces nombres.

Exercice 3 :
Soit A la liste des entiers compris entre 3 et 15. Générer cette liste de 4 façons différentes.

Exercice 4 :
Soit la liste X=[3,4,5]
Compléter les instructions ci-contre afin que la liste X devienne [3,4,5,10,11,12,13,14]

```
for i in range(.....) :
    ..........
```

Exercice 5 :
Une association met en place une cagnotte avec un premier dépôt de 50€. Cette cagnotte est alimentée chaque semaine avec un versement de 30€.
Compléter la fonction cagn pour qu'elle retourne la liste des contenus de la cagnotte pendant 11 semaines (cagnotte de départ comprise)

```
def cagn():
    c=...
    L=...
    for i in range(...):
        c=...
        ...
    return (L)
```

Exercice 6 :
Créer une fonction f(L) qui affiche les multiples de 5 d'une liste L de nombres entiers.
On pourra la tester avec la liste [6 ,4 ,50, 38, 75, 63, 0, 4, 40, 12]

Exercice 7 :
On peut calculer la moyenne pondérée d'une série de valeurs numériques. Pour cela, on définit la liste L de ces valeurs et la liste E des effectifs correspondants.
Recopier et compléter le programme de la fonction moy ci-contre afin qu'elle retourne la moyenne pondérée de cette série.

```
def moy(L,E):
    s=0
    N=0
    for i in range (...):
        s=...
        N=...
        return(s/N)
```