Développer une pensée informatique chez les élèves de cycle 3 à travers l'utilisation du logiciel Scratch

Blandine Follet

UNIVERSITÉ DE ROUEN

ESPE – ACADÉMIE DE ROUEN

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

Mention 1

Année 2017-2018

FOLLET BLANDINE

Développer une pensée informatique chez les élèves de cycle 3 à travers l'utilisation du logiciel Scratch

Sous la direction de : Mme NGONO

Mme HOUDEMENT

UNIVERSITÉ DE ROUEN

ESPE – ACADÉMIE DE ROUEN

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

Mention 1

Année 2017-2018

FOLLET BLANDINE

Développer une pensée informatique chez les élèves de cycle 3 à travers l'utilisation du logiciel Scratch

---

Sous la direction de : Mme NGONO

Mme HOUDEMENT

Résumé : La pensée informatique peut se développer dès le plus jeune âge, à l'école. Les programmes d'enseignement de l'école élémentaire et du collège de 2015, même s'ils ne sont pas les premiers, demandent aux enseignants d'initier leurs élèves à la programmation informatique. Dans ce mémoire, nous cherchons à savoir quelles situations on peut proposer à des élèves de cycle 3, en lien avec la construction de figures géométriques. Nous étudions aussi dans ce mémoire les compétences que vont être amenés à développer les élèves et la manière dont ils vont gérer leurs erreurs. Pour cela, en nous appuyant sur des éléments didactiques et pédagogiques, nous présentons une séquence mise en œuvre dans une classe de CM1 durant laquelle les élèves ont été amenés à manipuler le logiciel Scratch et son langage de programmation portant le même nom. Une analyse a priori et a posteriori de cette séquence nous permet de mettre en relief certains résultats marquants relatifs aux comportements des élèves et à leurs stratégies mises en œuvre.

Mots clés : Programmation – Scratch – Cycle 3 – Géométrie – Pensée informatique

---

MASTER MEEF : CHARTE DE NON PLAGIAT

Je soussigné(e),
Nom, Prénom : Follet, Blandine
Régulièrement inscrit à l'Université de Rouen
N° étudiant : 21 30 59 19
Année universitaire : 2017-2018

Certifie que le document joint à la présente déclaration est un travail original, que je n'ai ni recopié ni utilisé des idées ou des formulations tirées d'un ouvrage, article ou mémoire, en version imprimée ou électronique, sans mentionner précisément leur origine et que les citations intégrales sont signalées entre guillemets.

Conformément à la charte des examens de l'université de Rouen, le non-respect de ces dispositions me rend passible de sanctions disciplinaires.

Fait à : Rouen
Le : 25/04/2018
Signature : Blandine Follet

Remerciements

Je tiens tout d'abord à remercier madame Violaine Le Coz, formatrice en mathématiques à l'ESPE de Rouen, qui m'a fait découvrir Scratch avec passion lors de l'année de préparation au concours de professeur des écoles et qui m'a accompagnée et lancée dans les prémices de ce mémoire.

Ensuite, je voudrais remercier très sincèrement Marion Le Torriellec, ma binôme sur ce travail de mémoire, avec qui j'ai co-écrit la partie théorique mais surtout avec qui j'ai partagé beaucoup de moments de réflexion qui, je pense, nous ont toutes les deux fait avancer.

Je souhaite aussi remercier madame Catherine Houdement, professeur des universités à l'ESPE de Rouen, qui, au cours des différentes séances concernant l'UE recherche, a participé au développement des compétences liées à l'observation et à l'analyse rigoureuse de situations professionnelles.

Enfin, je tiens à exprimer toute ma reconnaissance à ma directrice de mémoire madame Bernadette Ngono pour son investissement dans l'accompagnement de ce mémoire. Je la remercie de m'avoir aidée, guidée, motivée. Merci aussi pour ces échanges qui ont fait mûrir ma pensée et mon analyse quant aux situations étudiées dans ce mémoire.

Introduction

La science informatique, et plus particulièrement l'initiation à la programmation, est de nouveau entrée dans les programmes de l'éducation nationale pour l'école et le collège de 2015. Ces notions étaient déjà présentes dans des programmes précédents, notamment ceux de 1985 qui demandent à l'enseignant, dans la partie mathématiques « d'initier l'élève à la recherche d'algorithmes et de développer ses capacités logistiques »[1], cependant, elles avaient disparu ensuite. Ainsi, depuis septembre 2016, les professeurs des écoles doivent enseigner non plus uniquement l'utilisation d'outils numériques mais l'informatique « comme un ensemble de concepts et de méthodes propres »[2] tels que l'algorithmique, la programmation, le code, la pensée informatique, etc. Cette compétence informatique est aussi présente dans le socle commun de compétences et de culture, notamment dans le domaine 1 : les langages pour penser et communiquer. Ainsi, l'objet de cette recherche se justifie par cette demande institutionnelle.

En outre, le 21ème siècle a vu l'essor du numérique dans notre société. L'informatique est maintenant présente chaque jour autour de chaque individu. Etant donné que la finalité de l'Ecole est que les enfants puissent mieux appréhender le monde qui les entoure, l'enseignement de l'informatique apparaît maintenant comme incontournable.

Il est important ici de clarifier le terme informatique. L'informatique est la science du traitement automatique de l'information. Ainsi, comme le fait noter Pierre Tchounikine, l'enseignement de l'informatique ne doit pas être confondu avec « l'enseignement de l'usage d'un ordinateur […] ou de l'usage de logiciels standards »[3]. Il s'agit bien de développer la pensée informatique, aussi appelée pensée algorithmique, et cela ne demande pas nécessairement l'utilisation d'un ordinateur.

Par conséquent, Scratch, langage de programmation visuel, n'est qu'un support parmi d'autre qui permet de travailler l'algorithmique, comme la robotique pédagogique par exemple. Il semble cependant pertinent d'utiliser un tel langage car cela permettra aux élèves de tester leurs algorithmes de manière ludique, à travers la réalisation et l'exécution de leurs programmes.

---

[1] *Programmes et instructions à l'école élémentaire* – Arrêté du 15 mai 1985
[2] Claire Calmet, Mathieu Hirtzig, David Wilgenbus, 2016, *1, 2, 3... Codez ! : Enseigner l'informatique à l'école et au collège (cycles 1, 2 et 3)*, Editions le Pommier.
[3] Pierre Tchounikine, *Initier les élèves à la pensée informatique et à la programmation avec Scratch*, Université Grenoble-Alpes.

Cependant, même si Scratch paraît être un support pertinent et que les objectifs de séances avec des élèves de cycle 3 seront de développer la pensée informatique, cela reste assez flou. Quelles vont être les compétences à développer ? Comment les développer ?

Ainsi, nous articulerons ce travail de recherche autour des questionnements suivants :

- **Quelles situations un enseignant peut-il proposer pour favoriser le développement de la pensée informatique chez des élèves de cycle 3 en utilisant le logiciel Scratch ?**
- **Comment les élèves gèrent-ils l'erreur sur le logiciel Scratch ?**

L'écrit qui va suivre tente de donner une vision de ce que doit être l'enseignement de l'informatique à l'école avec une première étude bibliographique, dans une partie théorique. Nous présenterons ensuite la méthodologie envisagée pour répondre aux questions que nous nous posons. Enfin, nous exposerons les résultats de recherche, résultats étayés par l'analyse des comportements des élèves et de leurs réalisations, avant de conclure.

La partie théorique de ce mémoire a été écrite en collaboration avec Marion Le Torriellec, elle aussi étudiante en M2 MEEF mention 1. La partie des résultats a été rédigée de manière individuelle selon nos expérimentations. Il sera donc uniquement exposé dans nos mémoires respectifs les résultats de nos propres observations.

1. Enseigner l'informatique à l'école : cadre théorique

Dans cette partie, nous présenterons d'abord les aspects institutionnels relatifs à l'enseignement de l'informatique à l'école, ainsi que les éléments pouvant y être rattachés. Nous convoquerons ensuite les résultats issus de la recherche. Enfin, nous présenterons le logiciel Scratch.

1.1. Aspects institutionnels – les programmes (2015 et rappels d'anciens programmes)

Tout d'abord, notons que ce n'est pas la première fois que les programmes envisagent l'initiation à la programmation à l'école primaire. En effet, dans les programmes de 1985[4] il est écrit dans la partie « mathématiques, instructions » : « *l'utilisation de l'informatique, à propos de la résolution d'un problème numérique ou géométrique, en particulier au cours moyen, permet d'initier l'élève à la recherche d'algorithmes et de développer ses capacités logistiques.* ».

De même en sciences et technologie en CM, « *L'importance de l'informatique justifie qu'au cours moyen cinquante heures au moins lui soient consacrées*. », « *Il [l'élève] acquiert les rudiments d'une culture informatique* ». Les « *Objets et systèmes informatiques* » étudiés sont : « *Le développement de l'informatique dans la société (transformation de l'activité professionnelle et de la vie quotidienne par la télématique, la bureautique et la productique ; problèmes sociaux et éthiques). La technologie informatique (le micro-ordinateur ; automates programmables et robots). Le logiciel (analyse et modification de logiciels simples ; début de programmation dans une perspective logistique).* ».

Dans les programmes de 1995[5] on retrouve des compétences dans le domaine de l'informatique : « *Le maître familiarise l'élève avec l'utilisation de l'ordinateur qu'il met au service des disciplines et dont il fait comprendre les différentes possibilités.* ».

Cependant, on peut observer la disparition de la programmation puisque en sciences et technologie au cycle 3, on lit :

« *Informatique*

- Quelques utilisations de l'informatique à l'école et dans l'environnement quotidien.

[4] *Programmes et instructions à l'école élémentaire* – Arrêté du 15 mai 1985
[5] *Programmes de l'école primaire* – Instructions officielles du 22 février 1995

- Utilisation raisonnée d'un ordinateur et de quelques logiciels (traitement de texte, tableur et logiciels spécifiques à l'école primaire) dans le cadre de l'enseignement des champs disciplinaires ; approche des principales fonctions des micro-ordinateurs (mémorisation, traitement de l'information, communication). »

Comme nous l'avons indiqué en introduction, les programmes 2015 proposent à nouveau d'initier les élèves au codage informatique.

Cependant, dès l'école maternelle[6], les élèves peuvent apprendre à utiliser un clavier d'ordinateur. En effet, dans le domaine « Mobiliser le langage dans toutes ses dimensions » les programmes indiquent qu'en fin d'école maternelle, les élèves doivent *« Reconnaître les lettres de l'alphabet et connaître les correspondances entre les trois manières de les écrire : cursive, script, capitales d'imprimerie. Copier à l'aide d'un clavier.* »
Dans le domaine 5, « Explorer le monde », les programmes sensibilise les enseignants à l'intégration du numérique dans les enseignements. En effet, ils indiquent que « *Dès leur plus jeune âge, les enfants sont en contact avec les nouvelles technologies. Le rôle de l'école est de leur donner des repères pour en comprendre l'utilité et commencer à les utiliser de manière adaptée (tablette numérique, ordinateur, appareil photo numérique…). Des recherches ciblées, via le réseau Internet, sont effectuées et commentées par l'enseignant. en fin d'école maternelle, l'élève doit être capable d'utiliser des outils numériques dont appareil photo, tablette, ordinateur.* »

Au cycle 2, c'est encore l'utilisation d'un clavier d'ordinateur qui est poursuivie. Ils apprennent à utiliser les fonctions simples d'un traitement de texte, ils manipulent le clavier. « *De façon manuscrite ou numérique, ils apprennent à copier ou transcrire sans erreur, depuis des supports variés » (livre, tableau, affiche…) en veillant à la mise en page.* »[7] . Les élèves apprennent aussi à s'autocorriger. Ainsi, en fin de cycle, ils peuvent utiliser un correcteur orthographique.
C'est en mathématiques qu'apparaissent les premières activités liées au codage. Dans les attendus de fin de cycle, en ce qui concerne l'espace et la géométrie, les programmes indiquent que les élèves doivent être capables de « *Coder et décoder pour prévoir*,

[6] *Programmes d'enseignement de l'école maternelle*, BO spécial du 26 mars 2015
[7] *Programmes d'enseignement de l'école élémentaire et du collège*, BO spécial du 26 novembre 2015

représenter et réaliser des déplacements dans des espaces familiers, sur un quadrillage, sur un écran ».

Comme exemples de situations, d'activités et de ressources pour l'élève, les programmes de 2015 conseillent des « *Parcours de découverte et d'orientation pour identifier des éléments, les situer les uns par rapport aux autres.* » Il s'agit aussi d'amener les élèves à : « *anticiper et effectuer un déplacement, le coder ; réaliser des déplacements dans l'espace et les coder pour qu'un autre élève puisse les reproduire ; produire des représentations d'un espace restreint et s'en servir pour communiquer des positions.* »

Par ailleurs, s'agissant du codage informatique, les programmes proposent d'amener les élèves à « *Programmer les déplacements d'un robot ou ceux d'un personnage sur un écran.* »

En ce qui concerne les repères de progressivité, il est écrit : *« L'initiation à l'utilisation de logiciels de géométrie permettant de produire ou déplacer des figures ou composantes de figures se fait graduellement, en lien avec l'ensemble des activités géométriques et le développement des connaissances et compétences géométriques. L'usage des logiciels de géométrie dynamique relève essentiellement des cycles 3 et 4.* »

Au cycle 3, on retrouve la thématique de l'initiation à la programmation dans les disciplines des mathématiques et des sciences et technologies au cycle 3. En mathématiques, elle trouve sa place dans le domaine « Espace et géométrie » et en sciences et technologie, elle est travaillée en technologie.

Dans le domaine « Espace et géométrie » des programmes 2015, il est ainsi indiqué que « *Les activités spatiales et géométriques […] constituent des moments privilégiés pour une première initiation à la programmation notamment à travers la programmation de déplacements ou de construction de figures.* ».

Les compétences visée en fin de cycle, c'est-à-dire en sixième doivent ainsi amener l'élève à « *(Se) repérer et (se) déplacer dans l'espace en utilisant ou en élaborant des représentations* ».

Dans ce domaine des mathématiques, une rubrique spécifique concerne l'initiation à la programmation. Dans cette rubrique, les programmes conseillent de proposer des « *activités de repérage ou de déplacement* » permettant de « *programmer les déplacements d'un robot ou ceux d'un personnage sur un écran* ». Ils invitent aussi les enseignants à offrir aux élèves

des « *activités géométriques* » qui amèneront les élèves à la réalisation « *de figures simples ou de figures composées de figures simples* ».
Enfin, dans cette partie des programmes, ces derniers sollicitent les enseignants à travailler « *avec de nouvelles ressources comme […] des logiciels d'initiation à la programmation* ».

Dans le domaine « Sciences et technologie » et plus particulièrement en technologie, il est précisé que « *Les élèves découvrent l'algorithme en utilisant des logiciels d'applications visuelles et ludiques.* » afin de pouvoir, selon les attendus de fin de cycle « *Repérer et comprendre la communication et la gestion de l'information* ». La base de l'enseignement en technologie est l'étude d'un objet technique. Ainsi, les enseignants sont invités à initier leurs élèves à la programmation en utilisant des « *objets programmables* ».

Puisque les programmes mettent en relation le codage et les activités géométriques, nous allons aussi présenter les attendus des programmes de géométrie au cycle 3.

Ainsi, les programmes 2015 précisent que *« Les activités géométriques pratiquées au cycle 3 s'inscrivent dans la continuité de celles fréquentées au cycle 2. Elles s'en distinguent par une part plus grande accordée au raisonnement et à l'argumentation qui complètent la perception et l'usage des instruments*. ». Les attendus de fin de cycle indiquent que les élèves doivent savoir, à la fin de la sixième « *Reconnaître, nommer, décrire, reproduire, représenter, construire des figures et solides usuels »*. Cet attendu met en exergue les différents types de problèmes à proposer : description, reproduction, représentation et construction de figures simples. En particulier, on doit développer chez les élèves les compétences suivantes : « *Réaliser, compléter et rédiger un programme de construction ; Réaliser une figure simple ou une figure composée de figures simples à l'aide d'un logiciel. »*. L'utilisation de Scratch comme logiciel de programmation permettra aux élèves d'être confrontés à un autre langage de description de figure.

Enfin, l'initiation à la programmation participe à l'atteinte des composantes du socle[8]. En effet, dans le domaine 1 du socle « *Les langages pour penser et communiquer* », une des compétences à atteindre est de « *Comprendre, s'exprimer en utilisant les langages mathématiques, scientifiques et informatiques* ». Il est écrit : « *Il [l'élève] sait que des*

8 Décret n° 2015-372 du 31 mars 2015 relatif au socle commun de connaissances, de compétences et de culture

langages informatiques sont utilisés pour programmer des outils numériques et réaliser des traitements automatiques de données. Il connaît les principes de base de l'algorithmique et de la conception des programmes informatiques. Il les met en œuvre pour créer des applications simples. ». Ensuite, dans le domaine 2, « *Les méthodes et outils pour apprendre* », il est indiqué que l'élève doit apprendre à organiser son travail personnel. Il doit apprendre à « *identifier un problème, s'engager dans une démarche de résolution, mobiliser les connaissances nécessaires, analyser et exploiter les erreurs, mettre à l'essai plusieurs solutions* ».

1.2. Présentation de résultats issus de la recherche

Dans cette partie, après avoir présenté quelques concepts, nous reprendrons certains résultats issus de nos lectures.

1.2.1. Quelques définitions des notions clés

Les notions clés qui vont être définies sont : le code informatique, l'algorithme, la programmation et la pensée informatique.

Le code informatique

D'après M. Ben Henda,[9] Le code est désormais nécessaire là où il y a une information à transformer et un système d'information à innover. D'après ce chercheur, une ambiguïté persiste encore dans le choix d'une terminologie commune, les usages diffèrant entre « littératie numérique », « informatique », « programmation », « codage ». Il rappelle la polysémie du terme « code » et relève quelques définitions. Le Littré définit l'étymologie du terme « Code » comme un dérivé, depuis le XIIIe siècle, du terme « *codex ou caudex, assemblage de planches, des planchettes ayant servi à écrire* ».

Citant d'autres chercheurs, et en particulier Fiske, Ben Henda considère que « *Le code est donc « un système de sens commun aux membres d'une culture ou d'une sous-culture. Il se compose de signes et de règles ou de conventions qui déterminent comment et dans quel contexte ces signes sont utilisés et comment ils peuvent être combinés pour former des messages plus complexes* » (Fiske, 1990:19). « *Un code peut ainsi signifier un ensemble de lois et de règlements (code pénal), un système de signes et de symboles par lequel on traduit des informations en une série de règles de bonne conduite et de conventions dans un*

[9] Mokhtar Ben Henda. L'enseignement du code informatique à l'école : Prémices d'un humanisme numérique congénital. Chaire Unesco-ITEN. Actes de la 5e rencontre annuelle d'ORBICOM, Les éditions de l'immatériel, 2017. <hal-01516577>

environnement donné (code Braille) ou une combinaison de lettres et/ou de chiffres qui autorise l'accès à certains éléments informatiques (Code ASCII). »

Ben Henda inscrit donc le code, et en particulier le code informatique dans un système de convention. En effet, d'après cet auteur, « *le concept de « convention » […] constitue la clé de voute du bon fonctionnement de tout système de codes. C'est l'accord entre les membres d'un milieu partagé sur la façon comment les signes sont combinés entre eux pour former des codes porteurs de sens. Cette règle élémentaire s'applique autant à un code de loi qu'il faut savoir interpréter dans ses différents renvois, qu'à un code sémiotique qui donne tout son sens à un idéogramme ou un symbole, qu'à un code informatique qui ordonne les fonctionnalités exécutées par un ordinateur.* »

Faisant le lien avec les mathématiques, Ben Henda indique enfin que « *Les sciences mathématiques constituent elle-aussi un exemple très concret de représentations sous forme d'énoncés et d'expressions contenant des théorèmes, des formules, des équations et des variables codés dans des signes et des symboles qui relèvent de l'abstraction du monde du réel. Elles nécessitent des connaissances et des compétences mathématiques pour savoir encoder et décoder, transposer, interpréter et distinguer les différentes formes de représentations d'objets réels et de situations.* »

L'algorithme

Selon Pierre Tchounikine, « un algorithme est un enchaînement mécanique d'actions, dans un certain ordre, qui chacune a un effet, et dont l'exécution complète permet de résoudre un problème ou de faire quelque chose »[10]. Les algorithmes sont omniprésents, allant de la recette de cuisine à la requête sur un moteur de recherche. De plus, les élèves sont déjà confrontés aux algorithmes à l'école, notamment lors de l'apprentissage des techniques opératoires de l'addition, soustraction, multiplication ou division.

La programmation

La programmation est le fait de transcrire un ou plusieurs algorithmes en un programme, dans un langage qui pourra être traité par un ordinateur ou autre machine. Un des langages possibles est le langage Scratch que nous évoquerons par la suite.

[10] Pierre Tchounikine, *Initier les élèves à la pensée informatique et à la programmation avec Scratch*, Université Grenoble-Alpes.

Jusqu'au cycle 4, il ne se sera pas « question de formaliser d'aucune manière la distinction algorithme/programme »[11]. Ainsi, l'algorithme comme le programme seront vus comme une suite d'instruction pour arriver à un résultat.

Les élèves devront peu à peu s'approprier les notions sous-jacentes à l'algorithmique et à la programmation qui sont :

- L'action / l'instruction.
- Les structures conditionnelles (une action ou une séquence d'actions se déclenche que si les conditions sont remplies).
- Les boucles (permet de faire se répéter une action ou une séquence d'actions).

La pensée informatique

On trouve le questionnement d'un « mode de pensée spécifique à l'informatique »[12] dans un article de Jeannette Wing, professeur d'informatique aux Etats-Unis. La définition de cette pensée est très large : elle permettrait de résoudre des problèmes en s'appuyant sur des concepts informatiques afin de mieux comprendre le monde qui nous entoure. Il ressort plus particulièrement de cet article que la pensée informatique est le raisonnement par étape, la décomposition d'un problème en sous-problèmes plus simples et la capacité à l'abstraction. Ainsi, la pensée informatique est marquée par un raisonnement algorithmique.

Un autre chercheur ayant beaucoup écrit et travaillé sur la notion de pensée informatique dans ses articles est Seymour Papert, un des pionniers de l'intelligence artificielle et créateur du langage de programmation Logo. Dans le dossier réalisé par Michèle Drechsler (IEN conseillère TICE dans l'académie d'Orléans-Tours), il est énoncé que selon Seymour Papert, la pensée informatique s'articule autour de « *la pensée critique, la créativité, la résolution de problèmes, la communication, la collaboration et l'informatique* »[13]. Là, encore c'est une définition très large.

Selon Pierre Tchounikine, les compétences mises en jeu dans l'élaboration d'une pensée informatique sont les suivantes : « savoir décomposer un problème en sous-problèmes plus simples ; savoir réfléchir aux tâches à accomplir pour résoudre un problème en termes d'étapes et d'actions (algorithme) ; savoir décrire les problèmes et les solutions à différents niveaux d'abstraction ».

---

[11] Eduscol, *Algorithimique et programmation*, Mathématiques, cycle 4
[12] Jeannette Wing, *La pensée informatique* (traduction, 2006), https://interstices.info/jcms/c_43267/la-pensee-informatique
[13] Michèle Drechsler, *Initier les élèves au codage et à la programmation*, Canopé Académie de Rouen, 22 Mars 2017, http://classetice.fr/spip.php?article883

Cela rejoint les compétences explicités dans le document Eduscol pour les cycle 2 et 3 qui sont : « *adopter une démarche scientifique : utilisation d'un langage spécifique, contrôle, essais-erreurs* » et « *développer l'abstraction : apprendre à anticiper l'effet de telle ou telle séquence d'instructions avant même de la faire exécuter par une machine ou un programme*»[14].

1.2.2. Eléments pédagogiques et didactiques

Démarche d'investigation

L'ouvrage *1.2.3…CODEZ !* conseille aux enseignants d'utiliser la démarche d'investigation dans la réalisation de séquence autour de la programmation. C'est une démarche chère à la fondation la main à la pâte qui est partenaire du projet « 1,2,3…Codez ! ». Ce qui est très intéressant avec cette démarche c'est qu'elle débute par une situation problème posée aux enfants. Dans cet esprit de résolution de problème, elle parait effectivement adaptée aux développements de stratégies particulières chez les élèves qui sont alors actifs dans leurs apprentissages.

C'est aussi l'avis de Seymour Papert qui préconise une pédagogie active de la part de l'enseignant. En effet, dans un entretien pour Les Cahiers Pédagogiques, il explique que « *L'important, c'est que les enfants apprennent. Pour cela, il faut qu'ils soient actifs par eux-mêmes et que cela les motive* »[15] et plus loin il précise la place que doit avoir l'enseignant : « *L'enseignant ne peut plus être celui qui sait tout et qui impose son enseignement. Il sera celui qui conseille, qui sait accompagner, stimuler.* ».

Statut de l'erreur

La programmation est un domaine scientifique où le statut de l'erreur tient un rôle spécifique. En effet, l'erreur fait partie du cheminement de l'élaboration d'un programme. Une des procédures que peut utiliser l'élève pour résoudre un problème algorithmique est le tâtonnement, la progression par phases d'essais et d'erreurs. Ainsi, cela rend l'utilisation d'un logiciel de programmation assez pertinente. En effet, comme le dit Pierre Tchounikine, on peut gérer la notion d'erreur en informatique par « l'utilisation d'une machine qui permet de constater l'effet de ce que l'on a produit […] pour amener les élèves à développer des démarches exploratoires, pour favoriser le « tâtonnement expérimental » [16](anticipation /

[14] Eduscol, *Initiation à la programmation aux cycles 2 et 3*, Mathématiques
[15] Alain Jaillet, *Un défi de taille pour l'école*, Entretien avec Seymour Papert, Les cahiers pédagogiques, n°398, septembre 2001
[16] Pierre Tchounikine, *Initier les élèves à la pensée informatique et à la programmation avec Scratch*, Université Grenoble-Alpes.

contrôle / validation) ». Cela se rapproche d'ailleurs beaucoup de la démarche d'investigation.

Le langage en géométrie

L'équipe de didactique des mathématiques ERMEL appuie l'importance du langage en géométrie dans leur ouvrage *Apprentissages géométriques et résolution de problèmes*[17]. Ce langage a plusieurs fonctions, comme le disent les auteurs de cet ouvrage : « *Les enfants ont besoin d'un minimum de langage géométrique pour : décrire des figures, expliquer leurs procédures ; argumenter sur leurs productions ou celles de leurs camarades.* ». Ainsi, ils préconisent à l'enseignant de « *proposer des situations de communication où le mot juste, plus précis, plus rigoureux apparaîtra comme le seul moyen d'établir l'accord et de savoir, sans ambiguïté, de quoi on parle* ».

Tout ceci est à mettre en parallèle avec le langage de programmation qui va être pour les élèves un nouveau langage permettant de décrire ou de construire des figures simples. De plus, le document Eduscol *Initiation à la programmation cycle 2 et 3* préconise aussi des situations de communication autour des activités de programmation en valorisant la précision du langage. Pour cela, « *l'enseignant organise des temps pour que les élèves explicitent leurs procédures ; ce qu'ils ont fait en premier, ce qu'ils ont fait ensuite, etc. Il met à disposition les supports qui facilitent ce temps de langage : étiquettes, dessins, photos, tablettes, etc.* ».

La résolution de problème en géométrie

Mettre en place des situations de résolution de problème est très important en géométrie afin que l'élève soit « *dans la position de construire ses connaissances en les faisant d'abord apparaître comme outils de solution de problèmes* » [18] comme le disent les auteurs de *Apprentissages géométriques et résolution de problèmes* (ERMEL). Dans ce même ouvrage ils indiquent les différents types de problèmes comme suit :

- *Problème de reconnaissance de relation (ex : quelle relation entre les côtés opposés de tel quadrilatère ?)*
- *Problème de production d'objet (ex : produire une perpendiculaire à une droite donnée.*
- *Problème d'existence d'objet (ex : existe-t-il une forme, prise dans un ensemble, superposable à une autre forme donnée ?)*

[17] ERMEL, *Apprentissages géométriques et résolution de problèmes (cycle 3),* Hatier, 2006
[18] ERMEL, *Apprentissages géométriques et résolution de problèmes (cycle 3),* Hatier, 2006

- *Problème de recherche de toutes les solutions (ex : le trésor est à égale distance de deux points A et de B ; où creuser ?)*

En situation de résolution de problème, la métacognition est importante. En effet, les élèves vont être amenés réfléchir en amont, planifier leur tâche, anticiper leurs actions et prévoir le résultat afin de choisir quelle stratégie sera la plus adaptée pour résoudre le problème.

Les stratégies d'autorégulation[19]

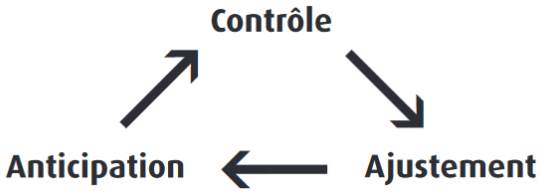
Il s'agit de processus en jeu dans la résolution de problèmes. Les stratégies d'autorégulation permettent aux élèves de :

1) Anticiper son action : c'est une phase où l'élève va planifier sa tâche.

2) Contrôler son activité : tout au long de la tâche, l'élève est capable de comparer ses résultats et de savoir s'il a atteint ou non l'objectif visé.

3) Ajuster son travail : En fonction du contrôle réalisé, s'il a atteint l'objectif, l'élève arrête son activité. Sinon, il reprend son travail : soit il le modifie, soit il le poursuit.

On peut résumer ces stratégies par le schéma ci-dessous.



Difficultés éventuelles

Il est maintenant important de noter les difficultés que pourront avoir les élèves avec les termes mais aussi la logique sous-jacente en programmation.

Les difficultés présentées dans cette partie sont issues des documents Eduscol et aussi du document « L'analyse psychologique des activités de programmation chez l'enfant de CM1 et CM2 »[20] dont l'étude se base sur l'utilisation de la tortue LOGO.

Commençons par la **notion de variable**. Comme le rappelle le document d'accompagnement Eduscol, « *la notion de variable en informatique diffère de la notion*

[19] Jérôme Focant, *Impact des capacités d'autorégulation en résolution de problèmes chez les enfants de 10 ans*

[20] *Mendelsohn Patrick. L'analyse psychologique des activités de programmation chez l'enfant de CMI et CM2. In: Enfance, tome 38, n°2-3, 1985. L'ordinateur et l'écolier (II) : recherches expérimentales. pp. 213-221*

mathématiques »[21]. En informatique, la variable peut être modifiée au cours du programme lorsque qu'il y a une instruction d'affectation. Pour les élèves, on expliquera que la variable est en fait « une étiquette collée sur une boîte qui peut contenir différentes valeurs ». Cette notion de variable reste difficilement accessible pour des élèves de classes élémentaires.

Le déroulement de l'exécution d'un programme est aussi quelque chose d'essentiel que les élèves devront assimiler. En effet, comme le précise le même document Eduscol, l'ordre des actions dans un programme a une grande importance. Deux programmes similaires ne vont pas arriver au même résultat si l'ordre des instructions est différent. La notion de temporalité se cache donc derrière l'exécution d'un programme. Pour Patrick Mendelsohn, cela correspond aux « *difficultés logiques* » qu'éprouvent les élèves. En effet, « *La compréhension de programmes suppose que l'enfant construise […] une aptitude à mobiliser mentalement des opérations qui rendent compte de la structure des programmes »* [22]et l'auteur ajoute que cela est relié *« à la capacité du sujet à pouvoir coordonner plusieurs actions élémentaires. »*

Une autre difficulté que l'on peut évoquer concerne **les types de déplacements : relatifs ou absolus**. Le déplacement relatif dépend de la position momentanée d'un personnage ou autre objet (ex : Tourner à gauche) quand le déplacement absolu n'en dépend pas (ex : Sur Scratch, on peut donner l'instruction d'aller vers une position définie par des coordonnées (x ; y)). Les élèves seront confrontés aux deux dans l'élaboration d'un algorithme et il sera donc nécessaire de bien cerner les différences, d'autant plus que c'est par cette approche des déplacements de personnages ou de robots que les programmes préconisent d'initier à la programmation. Il faut remarquer ici que les coordonnées données avec les variables x et y ne sont pas au programme de l'élémentaire. Il faudra donc contourner cette difficulté en créant des blocs adaptés sur Scratch. De plus, comme le précisent les documents d'accompagnement Eduscol, « la notion de déplacement relatif peut poser problème pour les élèves non latéralisés. Les difficultés rencontrées au quotidien pour lire des cartes papier et suivre les instruction d'un GPS l'illustrent assez bien »[23]. Cette difficulté remarquée par Patrick Mendelsohn qui met en avant la difficulté de « *La prise en compte de la position et de l'orientation finale de la tortue* ».

[21] Eduscol, *Algorithimique et programmation*, Mathématiques, cycle 4

[22] *Mendelsohn Patrick. L'analyse psychologique des activités de programmation chez l'enfant de CMI et CM2. In: Enfance, tome 38, n°2-3, 1985. L'ordinateur et l'écolier (II) : recherches expérimentales. pp. 213-221*

[23] Eduscol, *Initiation à la programmation aux cycles 2 et 3*, Mathématiques

Enfin, Patrick Mendelsohn relève aussi « *les difficultés sémantiques* » liées au fait que « *Chaque code de programmation comprend des instructions propres.* » qu'il faut que l'élève s'approprie. Par exemple, pour faire le parallèle avec le langage de programmation Scratch, les actions « Tourner » et « Avancer » sont toutes les deux nécessaires pour faire avancer le lutin vers une nouvelle direction alors que dans la vie quotidienne, « tourner » signifie généralement la combinaison des deux actions « Tourner » et « Avancer ».

Variables didactiques

L'enseignant peut utiliser des variables didactiques pour varier les difficultés des situations problèmes. Quelques variables sont énoncées dans le document d'accompagnement pour les cycles 2 et 3 : « *le nombre de pas pour le déplacement d'un robot ou d'un personnage dans une grille ou sur un écran, le nombre d'instructions nécessaires, l'environnement, les supports, les instructions disponibles, etc.* ». Cela nous amène aux différents types d'activités que peut proposer l'enseignant à ses élèves. Tout d'abord, les situations d'apprentissage peuvent se faire « en débranché ». Ce sont des activités qui permettent l'élaboration d'algorithmes, principalement concernant le déplacement absolu ou relatif d'objets ou de personnes. Ensuite, il y a beaucoup de situations à exploiter autour de la robotique pédagogique avec des robots conçus pour l'apprentissage de la programmation (ex : Bee-Bot, Thymio II, …). D'ailleurs, il est important ici de noter que les robots programmables ne sont pas nouveaux dans l'univers de l'éducation. En effet, Seymour Papert a développé la tortue Logo dont la première version date de 1967, spécialement conçue pour une première approche de la programmation à l'école [24] et permettant de proposer au élèves des situations de résolution de problème. Enfin, l'utilisation du logiciel Scratch est aussi préconisée par les documents d'accompagnement.

1.3. Le logiciel Scratch

Scratch est à la fois un logiciel et un langage de programmation. Il a été créé par le laboratoire Lifelong Kindergarten Group du Massachussetts Institude of Technology. Il a été pensé pour être ludique, visuel et pour avoir un côté éducatif.

Le langage Scratch fonctionne à partir d'instructions notées sur des blocs que les élèves sont amenés à manipuler. L'encastrement successif de différents blocs constitue le programme. Ces blocs ont différentes couleurs qui correspondent à leurs catégories d'action : mouvement, apparence, sons, stylo, données, évènements. Le principe de base est

[24] Alain Jaillet, *Un défi de taille pour l'école*, Entretien avec Seymour Papert, Les cahiers pédagogiques, n°398, septembre 2001

que l'on donne des instructions à un « lutin » qui est un petit chat orange par défaut. Le langage Scratch est basé sur de la programmation évènementielle : le programme réagira donc aux différents évènements qui se dérouleront, comme par exemple cliquer sur le drapeau vert pour démarrer l'exécution du programme.

Pourquoi utiliser Scratch à l'école semple pertinent ? Tout d'abord, c'est le langage qui est au programme du cycle 4, ainsi, il est recommandé par l'institution. Ensuite, concernant l'algorithmique, les compétences sous-jacentes peuvent se développer sans recours à un ordinateur, cependant créer le programme correspondant à l'algorithme permet de le tester et de mieux procéder par tâtonnement expérimental. La gestion de l'erreur sera plus aisée puisque l'élève lui-même pour se rendre compte s'il y en a une et ensuite chercher à la corriger. En outre, utiliser un logiciel de programmation permettra aux élèves de gérer la difficulté de l'abstraction. En effet, ils passeront de l'abstrait au concret en réalisant le programme sur Scratch. De plus, comme le disent les documents d'accompagnement Eduscol, l'explicitation de leur programme pourra leur permettre de développer abstraction et autonomie.

Il y a cependant une limite à l'utilisation de Scratch à laquelle il faudra être vigilent : la possibilité de procéder par essais-erreurs en permanence peut entrainer chez l'élève une non-réflexion préalable à l'élaboration d'un algorithme. Or c'est aussi cette réflexion qui permet de développer des compétences en algorithme.

2. Méthodologie

Dans cette partie de méthodologie, nous exposerons dans un premier temps la problématique et le dispositif mis en œuvre pour tenter d'y répondre. Ensuite, nous nous intéresserons au contexte d'expérimentation. Puis, nous détaillerons quelles données seront recueillies et comment elles seront recueillies. Enfin nous présenterons la séquence à mettre dans la classe en analysant les tâches demandées aux élèves.

2.1.1. Problématique et dispositif mis en œuvre

Nous avons vu que de nombreuses compétences sont en jeu dans le développement d'une pensée informatique : la réalisation d'algorithme, la décomposition en sous-problème, le repérage et le traitement des erreurs, l'abstraction, …

Nous nous concentrerons dans ce travail de recherche aux compétences liées aux erreurs dans les programmes et à celle de l'abstraction car il apparaît que comprendre qu'un programme est erroné confirme la bonne analyse abstraite et la compréhension d'un algorithme.

Ainsi, nous affinerons notre questionnement autour de la problématique suivante :

Quelles situations un enseignant peut-il proposer à des élèves de cycle 3 pour favoriser le développement des compétences liées à la gestion des erreurs et à l'abstraction en programmation, en utilisant le logiciel Scratch ?

Des questionnements annexes apparaissent aussi autour de cette problématique :

Dans quelles mesures les élèves vont-ils être capables d'anticiper les instructions d'un programme ?

Quelles vont-être les procédures des élèves quant à la gestion des erreurs dans l'utilisation du logiciel Scratch ?

Afin de réfléchir aux situations qu'un enseignant peut proposer à des élèves de cycle 3 pour que ces derniers développent des compétences liées à la gestion des erreurs et à l'abstraction en programmation, nous avons élaboré une séquence composée de 4 séances principalement tournées vers des situations de recherche et de résolution de problèmes. Nous détaillerons cette séquence par la suite.

Après la lecture des différentes activités proposées dans le document d'accompagnement *Initiation à la programmation aux cycles 2 et 3*, les activités de débogages semblent adaptées au développement de ces compétences. En effet, elles sont

directement liées à l'analyse de l'exécution d'un programme et aux repérages des erreurs (ou bogues). Ainsi il faudra que des activités de débogages interviennent dans la séquence envisagée.

2.1.2. Contexte d'expérimentation

Mes expérimentations ont lieu au sein d'une classe de CM1 de l'école élémentaire Ferdinand Buisson à Sotteville-lès-Rouen, commune de l'agglomération rouennaise. Cette école est composée de 11 classes : 2 CP, CP-CE1, 2 CE1, 2 CE2, 2 CM1, CM1-CM2, CM2. La classe de CM1 observée comprend 24 élèves : 11 filles et 13 garçons tous âgés de 9 à 10 ans.

Avant l'année 2017-2018, les élèves n'ont jamais suivi un enseignement de programmation informatique et n'ont jamais utilisé Scratch.

La séquence observée a lieu durant la troisième période, en janvier 2018. En première période, les élèves ont déjà découvert le logiciel Scratch (découverte libre puis réalisation de petits exercices permettant aux élèves de s'approprier les commandes de base). Les élèves ont aussi utilisé le logiciel Scratch en période 2 lors d'un projet durant lequel ils ont programmé un mini dessin animé (deux séances sur Scratch au cours du projet). Ainsi, lors de la réalisation de la séquence que nous allons proposer ensuite, les élèves étaient déjà familiarisé avec Scratch.

Les séances se déroulent en salle informatique où il n'y a que 12 postes informatiques. Souhaitant que chaque élève puisse travailler sur un poste, la classe est systématiquement partagée en deux groupes. Un groupe sera en autonomie sur un travail tout à fait indépendant de la séquence pendant que l'autre groupe sera en activité sur l'ordinateur. Ensuite les groupes sont intervertis afin que chaque élève réalise l'activité de programmation.

2.1.3. Le recueil des données

Les différentes séances sont à la fois menées et observées par moi-même. De plus, comme dit précédemment, la classe est partagée en deux groupes durant ces séances. Cela ne me permettra pas de me détacher pour observer un élève en particulier.

Ainsi, afin de mener un travail rigoureux et précis, un logiciel de capture vidéo des écrans sera utilisé pour pouvoir enregistrer chaque action de l'élève sur le logiciel.

Ensuite, les productions finales des élèves, c'est-à-dire leurs fichiers Scratch, pourront être récoltées ainsi que les productions « papiers » lorsqu'elles ont lieu durant la séquence.

Enfin, des observations générales seront menées par moi-même au cours de la séquence, que je noterai à l'écrit.

2.1.4. La séquence proposée

OBJECTIFS :

- Se repérer, décrire ou exécuter des déplacements, sur un plan ou sur une carte.
- Réaliser, compléter et rédiger un programme de construction.
- Réaliser une figure simple ou une figure composée de figures simples à l'aide d'un logiciel.

Séance 1 : Recherche du programme du carré

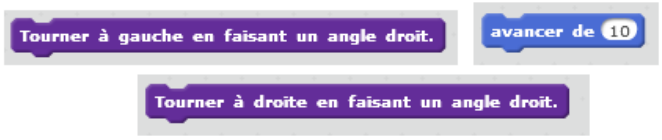
<u>Objectif :</u> Construire un programme sur Scratch permettant de dessiner un carré

L'exercice demandé lors de cette séance est le suivant :

Activité Scratch : Réaliser un carré

<u>Consigne</u> : Complète le programme pour que le petit chat dessine un carré.

Tu n'as le droit d'utiliser que les blocs suivants :



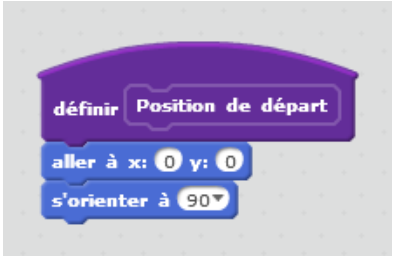
Tu as le droit d'utiliser ces blocs plusieurs fois. Tu as le droit de ne pas utiliser tous les blocs.

Nous allons maintenant apporter quelques précisions sur la consigne donnée aux élèves et effectuer une analyse à priori de la tâche à effectuer.

Le programme de départ est donné aux élèves sur le logiciel Scratch. Les élèves devront le compléter en s'aidant des blocs qu'on leur propose.

Figure 2-1 – Blocs « Tourner à gauche » et « Tourner à droite » créés par l'enseignant.

Les blocs « Tourner à gauche en faisant un angle droit. » et

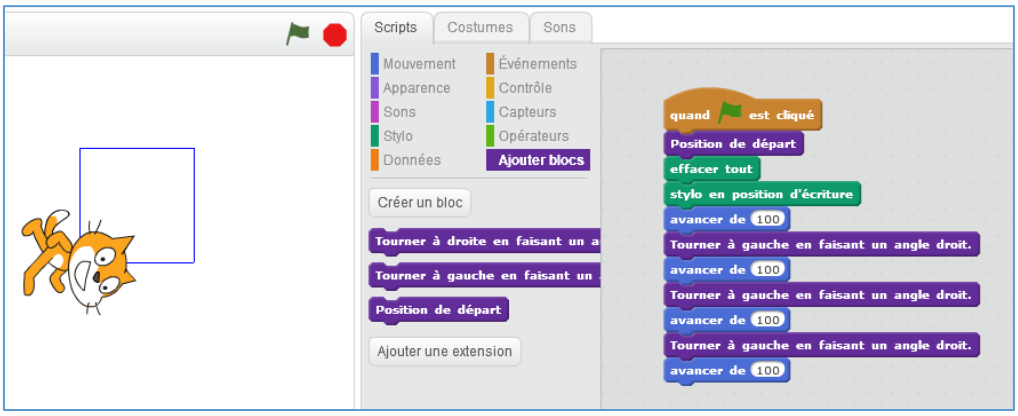


« Tourner à droite en faisant un angle de droit. » sont créés préalablement par l'enseignant comme on peut le voir sur la figure ci-contre pour éviter l'utilisation de la notion de degrés qu'ils ne connaissent pas encore.
Le bloc position de départ est défini comme cela :

Concernant la consigne, cette dernière est complétée par deux sous-consignes : « Tu n'as le droit d'utiliser que les blocs suivants : » et « Tu as le droit d'utiliser ces blocs plusieurs fois. Tu as le droit de ne pas utiliser tous les blocs. »
La première sous-consigne est un guidage de la part de l'enseignant. Elle est présente pour 2 raisons. La première raison est que les blocs créés (« Touner à gauche en faisant un angle droit » et « Tourner à droite en faisant un angles droit ») ne peuvent pas être ajoutés dans la catégorie « mouvement » des blocs et donc les élèves ne peuvent pas les retrouver intuitivement. La deuxième raison est que l'objectif de cette activité n'est pas de rechercher les blocs nécessaires pour créer le programme du carré mais plutôt de les articuler entre eux, de les lier au programme pour créer un carré. Cela permet de supprimer une difficulté préalable. La deuxième sous –consigne précise la première sous-consigne afin que la première ne les induise pas en erreur. En effet, à partir de la première sous-consigne, les élèves pourraient penser qu'ils n'ont strictement le droit que d'utiliser ces trois blocs sans pouvoir les utiliser plusieurs fois. Ils peuvent aussi supposer de la première sous-consigne qu'ils sont obligés d'utiliser les trois blocs différents.

Le déroulement de la séance va s'effectuer en commençant dans un premier temps par la présentation de l'exercice aux élèves. L'enseignant explique aux élèves où trouver les blocs violets qu'il a créés car cela n'est pas intuitif. Ensuite, l'enseignant explique les nouveaux blocs utilisés : "effacer tout" et "stylo en position d'écriture".

Dans un second temps, les élèves réalisent l'exercice. Par tâtonnement, par une démarche d'essais-erreurs, les élèves vont chercher à construire un carré.
Une production attendue est la suivante :



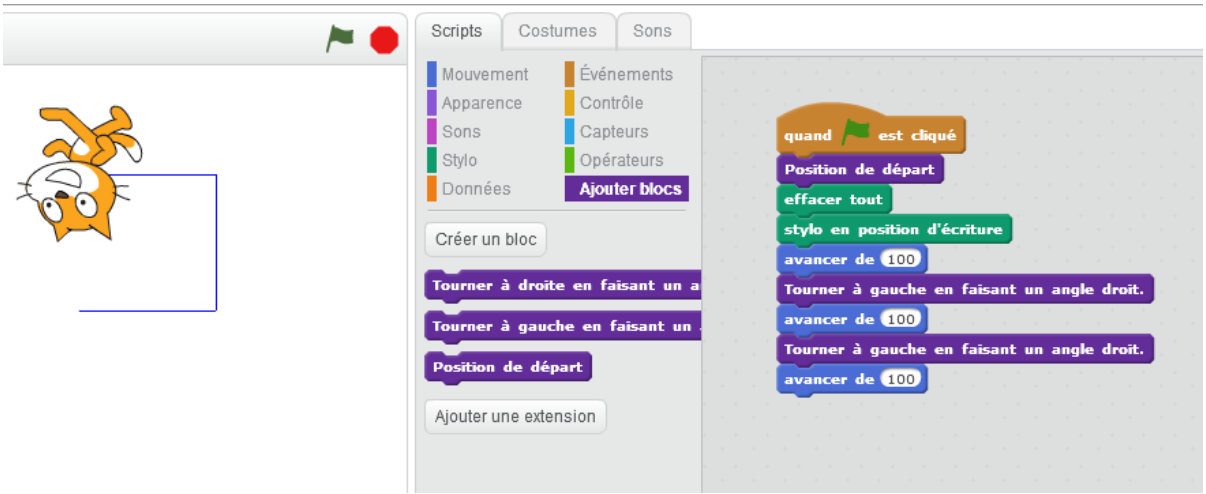
La troisième et dernière phase de la séance correspond à un temps de structuration lors duquel un programme possible du carré est proposé aux élèves (par exemple : celui présenté ci-dessus), qui le gardent comme trace écrite dans leur cahier.

Nous allons maintenant envisager les difficultés que les élèves vont éventuellement rencontrer ainsi que les erreurs possibles.

Une première difficulté serait pour l'élève de **définir de trop petites distances dans le bloc "avancer de"** : Le carré ne sera pas visible à cause de la place que prend le chat et les élèves pourront penser que c'est une erreur même s'ils élaborent un programme est correct.

Une deuxième difficulté possible, similaire à la première serait de **définir de trop grandes distances dans le bloc "avancer de"** : Le carré ne sera pas entièrement visible car une partie dépassera de la fenêtre de visualisation.

Une erreur à envisager est que l'élève **réalise un carré non terminé car toutes les étapes de construction ne sont pas présentes** comme on peut le voir sur la figure ci-dessous.



Une autre erreur possible est **l'absence de carré** car l'élève a simplement tourné sans avancer, pensant que le bloc « tourner » signifie en fait « tourner en avançant ». Son

programme ne sera alors constitué que de blocs « Tourner à [gauche/droite] en faisant un angle droit] » et d'aucun bloc « avancer de [nombre de pas] ».
Enfin, l'élève peut aussi **définir des longueurs différentes pour les différents côtés du carré**. Concrètement, cela signifie que les valeurs de nombre de pas ne sont pas les même dans tous les blocs « avancer » ajoutés par l'élève. Cela monterait alors une méconnaissance ou une difficulté à appliquer les propriétés du carré.

Concernant les variables didactiques à envisager, durant cette séance, la plus intéressante serait le nombre de blocs maximum autorisé donné aux élèves. En effet, cela pousserait les élèves à bien lire et anticiper leurs instructions afin de réaliser le programme le plus efficace possible.

Séance 2 : Activité de débogage

Objectifs :

- Corriger un programme comportant une erreur

- Réaliser une figure simple à l'aide d'un logiciel.

- Se projeter dans le temps, anticiper, planifier ses tâches.

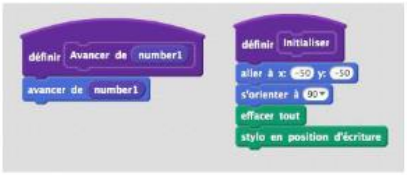
L'activité principale de cette séance est tirée du document d'accompagnement Eduscol[25] sur l'initiation à la programmation aux cycles 2 et 3 et est définie comme suit :

Activité 2 : Un carré en escalier

Description de l'activité

Les élèves doivent analyser le script suivant qui est censé permettre de tracer un carré de côté 100 pixels et le modifier en conséquence.

Pour éviter la manipulation des coordonnées et des degrés, il peut être intéressant de créer des blocs personnalisés afin d'initialiser la position initiale du lutin et d'y adjoindre trois commandes de déplacement préalablement définies : « Avancer de... », « Tourner à droite », « Tourner à gauche ». Cela suppose que l'enseignant mette à disposition des élèves un fichier dans lequel ces différentes commandes ont été créées.



L'énoncé donné aux élèves est celui présent ci-dessous. Il faut préciser qu'en dessous des lignes, les élèves disposent aussi d'un espace vide leur permettant éventuellement de dessiner à main levée la figure obtenue.

Pour tracer un carré de côté de longueur 100, Cléa propose le programme ci-contre .

Peux-tu sans utiliser Scratch prévoir ce qui sera tracé ?
Vérifie à l'aide du logiciel.
Corrige le programme pour qu'il permette de tracer un carré.

.....
.....
.....
.....
.....

[25] Esducol, *Initiation à la programmation, Annexe 5.5 : Scratch – Activités de débogage*

Nous allons maintenant réaliser l'analyse de la tâche à effectuer.

L'exercice indique que Cléa souhaite tracer un carré de longueur 100. L'environnement scratch permet d'interpréter que l'unité de longueur est le pas d'après les premières expériences qu'ont eu les élèves.

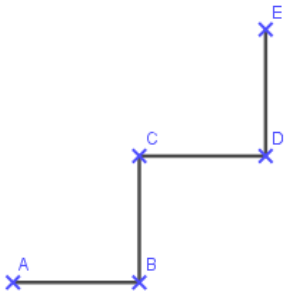
L'exercice est découpé selon trois consignes à effectuer les unes à la suite des autres.

L'objectif de la première consigne est de décoder un texte élaboré à l'aide de Scratch et décrire la figure ou de la construire à main levée, ce qui lui enlève l'obstacle de la manipulation des instruments.

Le programme proposé est constitué de 7 blocs permettant de repérer 4 longueurs identiques dont la mesure est exprimée en unités du programme (100 correspond à 100 pas) et de 3 blocs qui décrivent le sens du déplacement :

- Avancer correspond à aller vers la direction dans laquelle est orienté le lutin.
- Tourner à gauche (ou à droite) correspond à tourner de 90 degrés à gauche (ou à droite).

Le programme étant séquentiel, la lecture s'effectue de haut en bas. Les instructions s'exécutent ligne après ligne, les unes à la suite des autres. Une ligne de code commence quand la précédente est terminée.



Voici la figure que va dessiner le programme et donc celle qui est attendue si les élèves dessinent la figure à main levée. Nous avons nommé les points A, B, C, D et E sur la figure pour les utiliser plus facilement pendant l'analyse.

Pour effectuer le tracé, les élèves doivent placer un point de départ pour le premier déplacement, placer le point A et tracer le segment [AB]. Pour cela, ils peuvent mentalement envisager le tracé de manière à ne pas sortir du cadre uni. Ensuite, ils doivent tracer le deuxième segment à partir de B, et ainsi de suite afin de réaliser de proche en proche la figure attendue. Le tracé est sur support uni. Les élèves tracent à main levée, c'est donc l'allure de la figure obtenue qui est attendue et non un tracé précis.

Séance 3 : **Recherche**

<u>Objectif</u> : Reproduire une figure complexe à l'aide du logiciel Scratch (croix de lorraine, podium, lunettes).

L'élève se retrouve en situation de résolution de problème. Il aura à sa disposition la figure dessinée sur papier avec simplement un codage sur la figure. L'élève ne disposera pas de

programme de construction. Il devra reproduire cette figure sur le logiciel en élaborant un programme Scratch.

Séance 4 : Recherche

Objectif :

- **Anticiper le déplacement d'un personnage à l'écran.**
- **Construire un programme permettant de réaliser le déplacement de ce personnage.**

Un labyrinthe sera proposé à l'élève sur le logiciel Scratch. L'élève devra tout d'abord anticiper le déplacement du lutin pour sortir du labyrinthe dessiné à l'écran. Il pourra faire cette anticipation de manière mentale mais aussi en la mettant à l'écrit sur une feuille vierge qui sera à sa disposition. Enfin, l'élève va devoir programmer ce déplacement en construisant une séquence d'instructions avec les blocs de Scratch. L'élaboration de ce programme pourra se faire par « tâtonnement expérimental » c'est-à-dire par l'enchaînement de phases d'essais et d'erreurs.

2.2. Difficultés générales à anticiper liées à la programmation

Dans cette séquence, la gestion de l'erreur va être au cœur des difficultés à dépasser pour l'élève, en particulier lors de l'activité de débogage. En ce qui concerne la prévision des comportements et procédures observables durant cette activité, nous pensons que certains élèves vont supprimer tous les blocs du programme erroné pour recommencer depuis le début, que d'autres vont supprimer les blocs un par un et en déduire celui qui est responsable du bug, tandis que le reste des élèves va anticiper le programme et identifier en débranché le bloc erroné pour le supprimer et le remplacer par un bloc correct.

Ensuite, comme précisé dans la partie sur les aspects didactiques, lors de la construction de leurs programmes au cours de cette séquence, certains élèves pourraient confondre les déplacements relatifs et les déplacements absolus. De plus, lors des déplacements relatifs, la capacité de décentration des élèves, c'est-à-dire la capacité à se placer dans la perspective du lutin, sera essentielle pour mener à bien leur programme. C'est une capacité qui reste difficile pour des adultes, il faudra donc rester attentif à cela avec les élèves.

Enfin, on peut trouver d'autres difficultés à anticiper dans le dossier de Pierre Tchounikine, concernant plutôt la gestion de classe. En effet, ce dernier rappelle que l'accès aux ordinateurs est encore très difficile dans beaucoup d'école. Or ici, il s'agit de faire

travailler les élèves sur le logiciel Scratch (pour la modification du programme). Il prévient aussi quant à la tenue de la classe car l'excitation et la motivation que peuvent avoir les élèves fasse à l'utilisation du logiciel peut se transformer en une gestion de classe difficile. Il a aussi remarqué une difficulté des élèves en motricité fine lors de l'utilisation de la souris pour déplacer les blocs sur Scratch. Enfin, Pierre Tchounikine rappelle que l'utilisation de Scratch demande un guidage et une autonomisation forte des élèves.

3. Résultats

Tout d'abord, la séquence présentée précédemment n'a pas été réalisée dans son ensemble. En effet, seules les séances 1 et 2 ont été menées en classe. De plus, une séance de synthèse collective a été réalisée entre les séances 1 et 2. Cette séquence a dû être écourtée car elle est assez chronophage et que les contraintes liées au programme de mathématiques en CM1 m'ont obligé à avancer sur d'autres thématiques.

3.1. Séance 1 – Réalisation d'un carré sur Scratch

Concernant la première séance, nous allons dans un premier temps exposer les difficultés rencontrées quant à la gestion de classe et au recueil des données. Ensuite, nous étudierons de manière générale les productions finales des élèves. Enfin, nous nous intéresserons plus finement à certains constats et à certaines erreurs ou réussites dans les comportements et productions des élèves.

3.1.1. Gestion de classe et recueil des données

Tout d'abord, la gestion de classe durant cette première séance a été plutôt compliquée. En effet, la classe était partagée en deux groupes. Le premier groupe, censé être en autonomie m'a demandé beaucoup d'attention, autant dans la gestion du bruit que dans l'aide que j'ai dû leur apporter, ne me laissant pas la possibilité d'être complètement détachée pour le groupe en activité sur Scratch. Cela a notamment eu pour conséquence que sur les 22 élèves présents durant cette séance, seuls 15 ont enregistré correctement leur fichier final de Scratch. Cette gestion de classe difficile a aussi entraîné le fait que je n'ai pas pu me focaliser sur l'observation d'un ou deux élèves puisque j'étais très sollicitée.

Ensuite, je me suis rendue compte en cours de séance que le logiciel de capture vidéo que j'avais préalablement installé sur les ordinateurs n'avait pas fonctionné. Ainsi, lorsque j'en ai pris conscience, j'ai commencé à prendre en photo les écrans avec mon appareil photo mais je n'ai donc que très peu de productions intermédiaires illustrant les étapes de réalisation de l'activité par les élèves.

Ainsi, les données récoltées sont les suivantes : des observations générales notées sur une feuille, les productions finales de 15 élèves (voir annexe 1) et 9 photographies d'écran.

3.1.2. Les productions finales – résultats généraux

La figure attendue doit être semblable à la figure 3-1. Cependant, les longueurs des côtés, la position et l'orientation du lutin peuvent être différentes.

Sur les 15 productions finales enregistrées, 7 présentent la figure attendue. En voici quatre exemples :

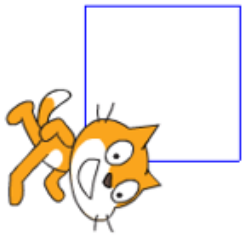


Figure 3-1 – Réalisation attendue

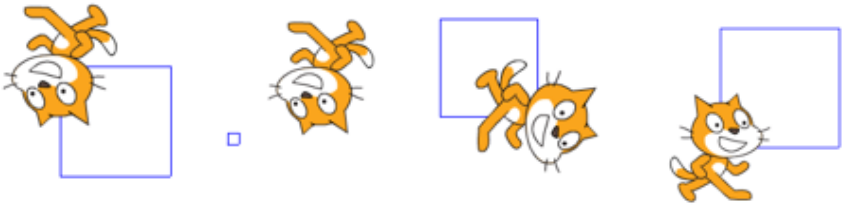
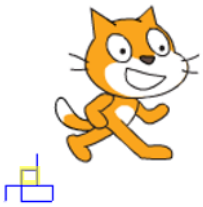


Figure 3-2 – Exemples de figures réalisées par quatre élèves (R, N, P et O).

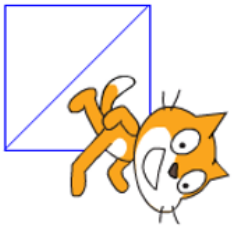
Sur deux autres figures (figure 3-3), on peut aussi observer la présence d'un carré mais la figure réalisée n'était pas celle attendue.

Figure 3-3 – Exemples de figures « non » attendues sur lesquelles un carré est présent.

(le carré est repassé en jaune)



L'élève L a identifié le carré puisqu'il me l'a montré en me disant qu'il avait réussi.



L'élève I a dessiné une diagonale en plus.

Cependant, même si sur les 15 productions finales, des carrés sont présents sur 9 d'entre elles, cela ne signifie pas pour autant que les programmes sont les mêmes. On peut d'abord très rapidement remarquer cela à travers le nombre de blocs ajoutés au programme de base par chacun des élèves. Le tableau 3-1 répertorie cette information (avec en gras les élèves qui ont réalisé la figure attendue). Il est à noter que les blocs qui n'ont pas été rattachés au début du programme n'ont pas été comptabilisés.

Nbre de blocs	0	3	4	5	6	8	9	10	11	12	16	25
Elève(s)	H	V	Q	U	X	**R**	**P**	T	I	**N/O/K**	**E/G**	L

Tableau 3-1 – Nombre de blocs ajoutés par les élèves au programme de base.

Ainsi, on peut voir instantanément que quasiment tous les programmes sont différents. De plus, deux programmes qui ont le même nombre de blocs peuvent être construits avec une succession d'instructions distinctes l'une de l'autre. Cela s'illustre bien avec les productions des élèves N et O (figures 3-4 et 3-5).

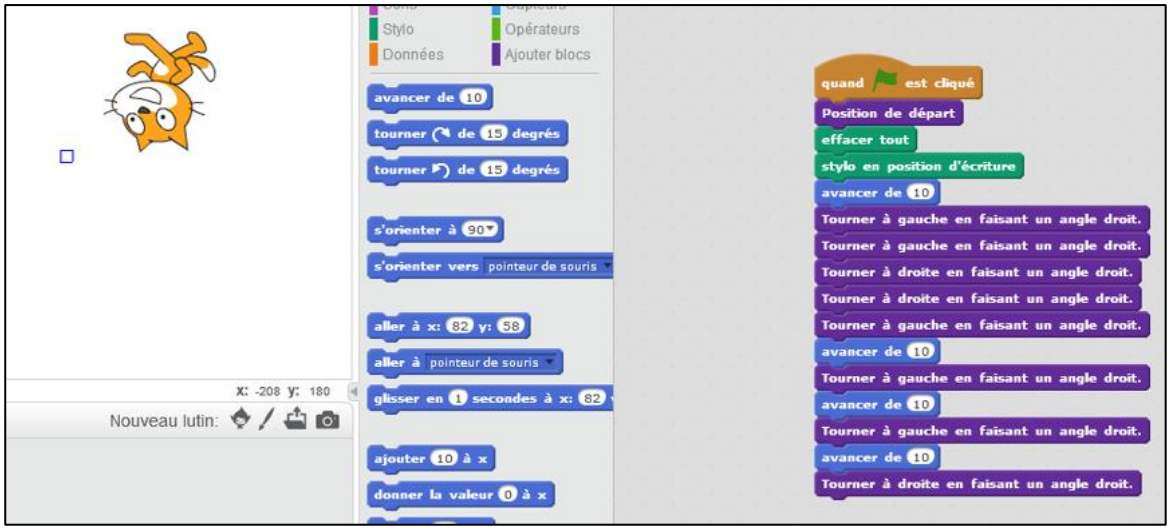


Figure 3-5 - Production élève O

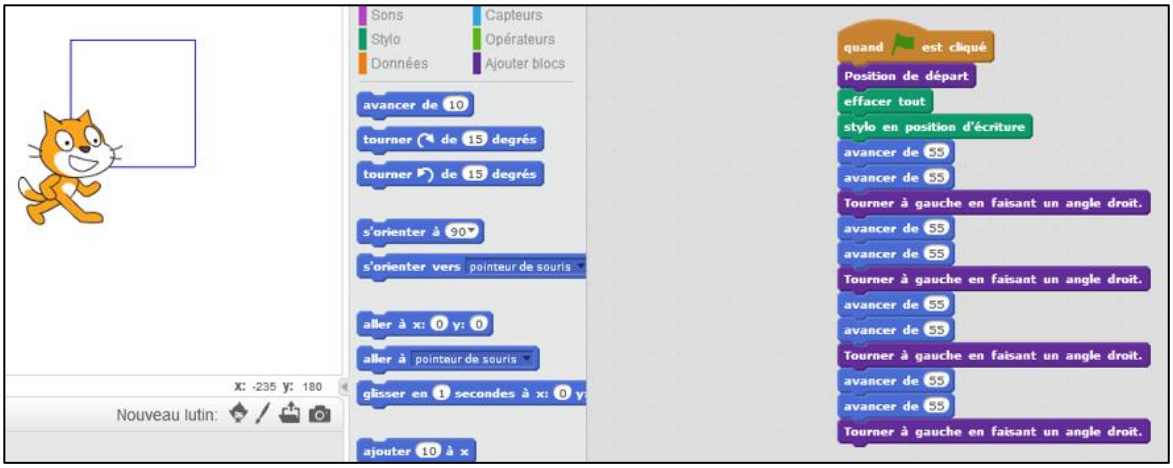


Figure 3-4 - Production élève N

En effet, l'élève N a utilisé 4 blocs « avancer », 5 blocs « Tourner à gauche » et 3 blocs « Tourner à droite » alors que l'élève O a utilisé 8 blocs « avancer » et 4 blocs « Tourner à gauche ». De plus, le nombre de pas défini dans les blocs « avancer » n'est pas le même pour les 2 élèves.

Le nombre de blocs minimum à ajouter était de 7 : 4 blocs « avancer » et 3 blocs « tourner » dans le même sens. Aucun des élèves ayant réussi la figure attendue ne la faite avec un nombre minimal de blocs. Ainsi, si la séance avait duré plus longtemps, il aurait été intéressant d'ajouter une variable didactique concernant le nombre de blocs à utiliser pour ces élèves en réussite.

Nous allons maintenant mettre en relief certains résultats et certaines observations plus précisément.

3.1.3. Analyse d'observations et de résultats marquants

Manque d'autonomie sur le logiciel

Un premier constat est que les élèves ont manqué d'autonomie sur le logiciel. Alors que les élèves s'étaient familiarisés facilement avec le logiciel et n'avaient pas hésité à « fouiller » dans celui-ci lors des premières séances en période 1, ils se sont en moyenne retrouvés bloqués en ce début de séance. En effet, j'ai été fortement sollicitée par des élèves qui me disaient **« Je n'y arrive pas. »** sans forcément avoir essayé des algorithmes à l'écran. Ils n'ont pas osé tester des procédures aussi facilement que je le pensais. Etait-ce une peur de l'erreur ? Est-ce dû au changement de statut de l'exercice sur Scratch qui était cette fois-ci explicitement caractérisé comme étant un exercice de géométrie ? Les élèves étaient-ils perdus dans l'utilisation du logiciel qui ne serait finalement pas si instinctif ?

Planification

Les élèves n'ont que très peu laissé de place à la planification de leur programme. En effet, je n'ai pas observé de temps de réflexion de la part des élèves avant qu'ils partent dans la manipulation des blocs sur Scratch. Deux hypothèses pourraient expliquer cela. La première serait que rien ne les invitait à le faire. Le support proposé avait été plastifié et ils n'avaient ni feuille ni crayon à leur disposition. La seule possibilité d'anticipation restait donc l'abstraction mentale. La deuxième hypothèse serait que l'utilisation de Scratch pourrait empêcher certains élèves d'accéder à un temps de réflexion car la tentation de tester sans cesse son programme serait trop importante.

La consigne

Dans la consigne, on demandait aux élèves de n'utiliser que certains blocs. Sur les 15 productions récoltées, 13 élèves ont respecté la consigne, et 2 élèves, les élèves I et X, ne l'ont pas respectée (figures 3-6 et 3-7).

Concernant la production de l'élève I, on peut s'apercevoir qu'il a glissé deux fois le bloc « Position de départ » dans son programme. Cela peut être dû à la zone de sélection des blocs « Tourner » qui présentait uniquement ces trois blocs comme on peut le voir sur l'image ci-contre. En effet, sur les trois blocs, deux étaient autorisés dans la consigne. Ainsi, on peut supposer que l'élève n'est pas forcément retourné voir la consigne et qu'il a pensé que ce bloc était lui aussi autorisé. De plus, la couleur identique de ce

bloc avec les deux autres peut aussi induire en erreur. Cependant, l'élève s'est adapté à l'utilisation de ce bloc supplémentaire et à tout de même réussi à réaliser un carré (avec une diagonale en plus).

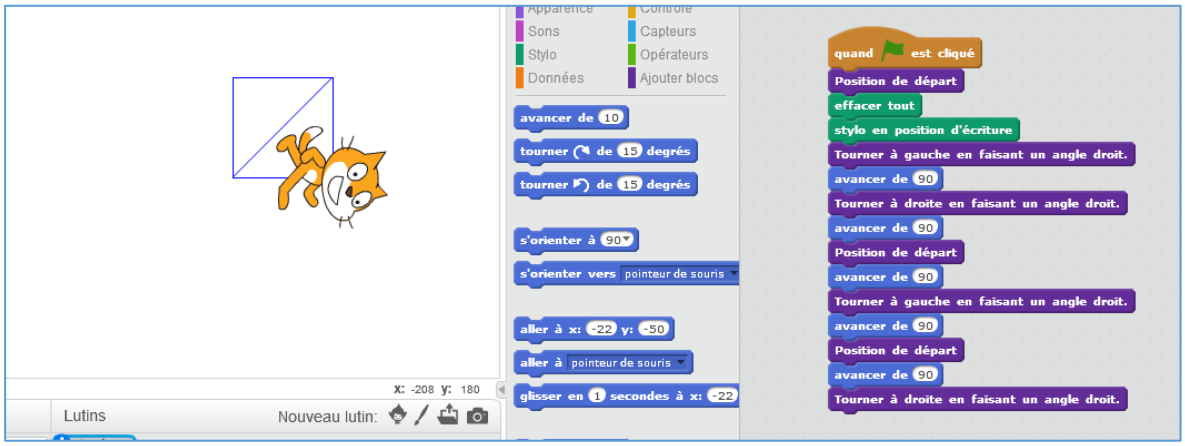


Figure 3-6 – Production de l'élève I

Pour l'élève X, c'est certainement dans la zone de sélection des blocs « mouvement » qu'il s'est laissé « submergé » par la présence de beaucoup d'autres blocs qu'il a souhaité tester. On peut remarquer qu'il utilise certains blocs sans vraiment les comprendre puisqu'il n'a encore aucune notion de coordonnées x et y à son âge.

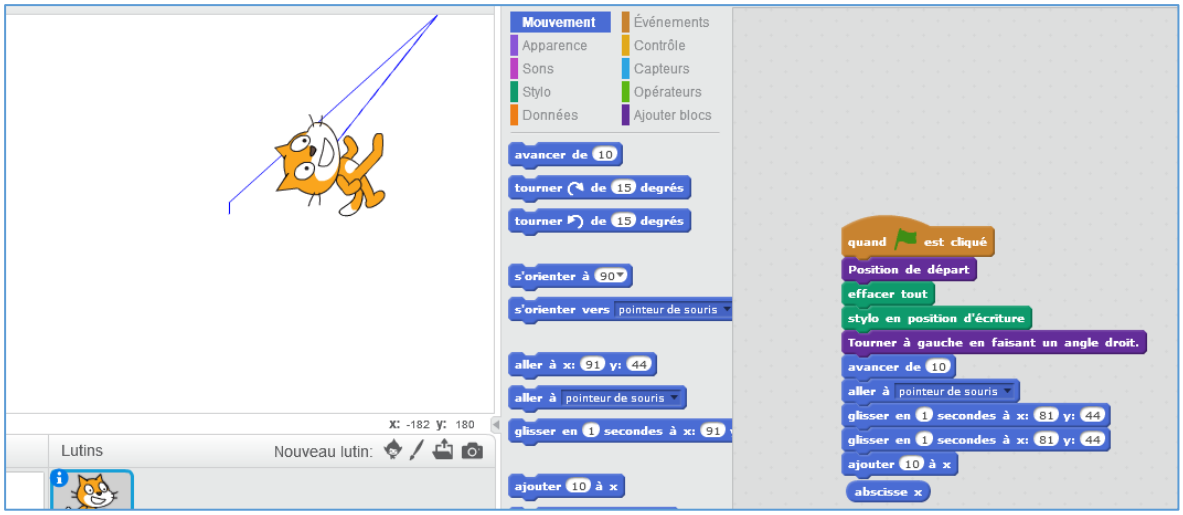


Figure 3-7 – Production de l'élève X

Concevoir le programme dans son ensemble

Une difficulté qui n'avait pas été anticipée est le fait que l'élève ne conçoit pas forcément un programme comme une seule entité qui se lance une seule fois pour exécuter toutes les instructions qui le compose. En effet, une grande majorité des élèves a commencé par cliquer successivement sur les blocs au lieu de les ajouter à la suite du programme. En fait, les élèves ont d'abord considéré les instructions comme étant indépendantes les unes des autres au lieu de créer une séquence d'instructions à réaliser d'un seul coup. Et nous

pouvons remarquer qu'envisager une séquence d'instructions est beaucoup plus difficile pour l'élève et lui prend plus de temps de réflexion.

Pour les aider à franchir cette étape de la séquence d'instruction, je leur indiquais que je souhaitais que le carré se réalise en cliquant juste sur le drapeau et les élèves entraient alors dans une dynamique de création d'algorithmes.

Pour illustrer mon propos, je vais vous présenter les productions intermédiaires et finales de deux élèves. Les deux élèves ont commencé par créer des carrés en cliquant successivement sur les blocs. Ensuite, après étayage, les deux se sont focalisés sur la réalisation du programme dans son ensemble. L'élève R a produit la figure attendue tandis que l'élève T n'y est pas parvenu. Je vais analyser le travail réalisé par ces deux élèves en m'appuyant sur leurs productions et sur les observations de leur comportement que j'ai pu noter pendant la séance.

Analyse des productions et du comportement de l'élève R :

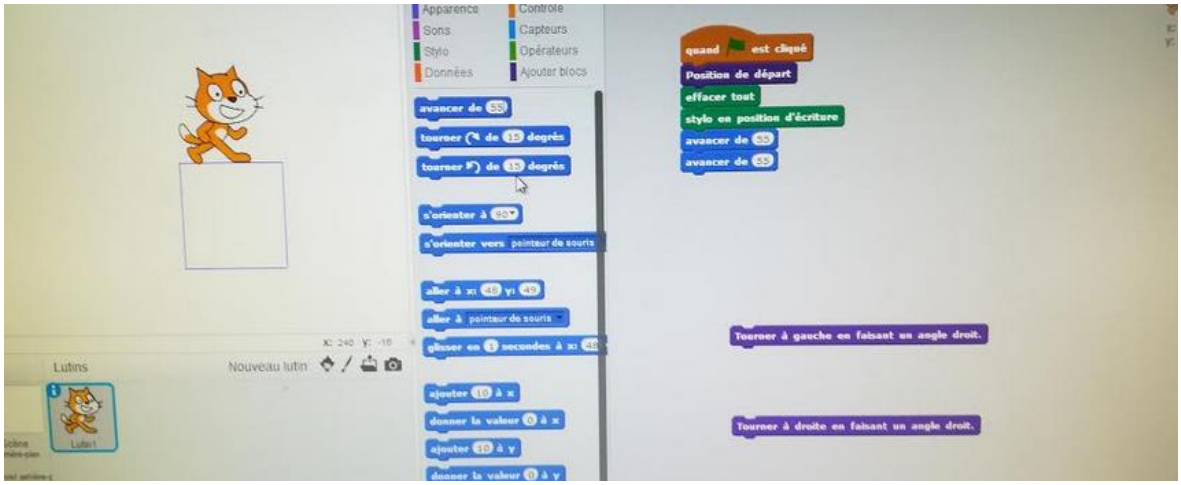


Figure 3-8 – Production intermédiaire de l'élève R

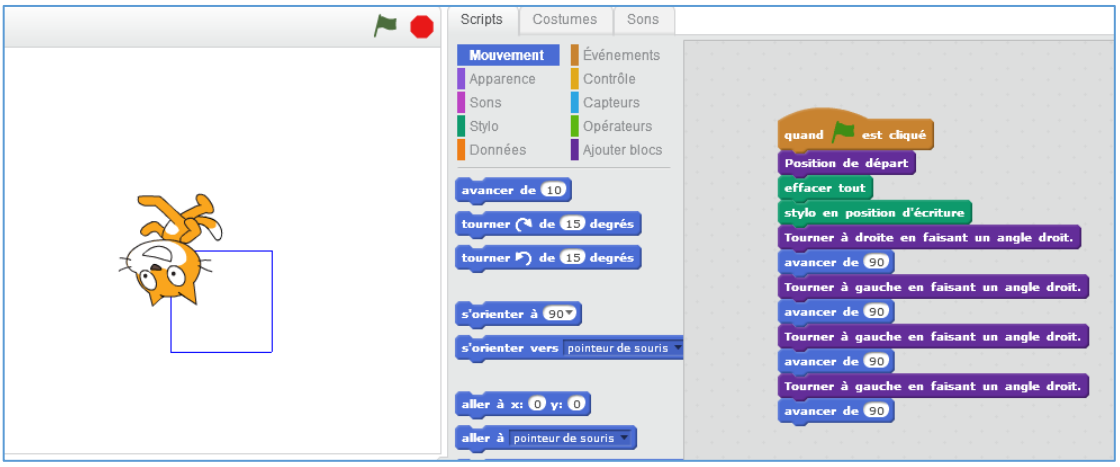


Figure 3-9 – Production finale de l'élève R

Sur la production intermédiaire de l'élève R, on peut observer que le programme écrit sur la droite de l'écran ne correspond pas aux figures tracées à gauche de l'écran. En effet, le

résultat du programme aurait dû être un segment de 90 pas, de plus, on voit sur l'écran de gauche 3 carrés. En dessous du programme, on peut observer la présence de 3 blocs : « avancer de 100 », « Tourner à gauche en faisant un angle droit. », « Tourner à droite en faisant un angle droit. ». On peut déduire de la présence des 3 blocs qu'il les a utilisés de manière indépendante en cliquant successivement sur certains de ces blocs pour créer un carré. Ensuite, pour créer un deuxième carré, l'élève déplace manuellement le lutin et recommence à cliquer sur ces blocs.

J'ai pu observer pendant la séance que pour arriver à ce résultat, cela a pris environ 5 min à l'élève. Ensuite, lorsque j'ai apporté un étayage, l'élève a eu plus de difficulté à imbriquer les unes en dessous des autres les instructions pour réaliser le carré (environ 20 min de recherche). L'élève R a tout de même fini par produire la figure attendue en procédant par essais-erreur.

Analyse des productions et du comportement de l'élève T :

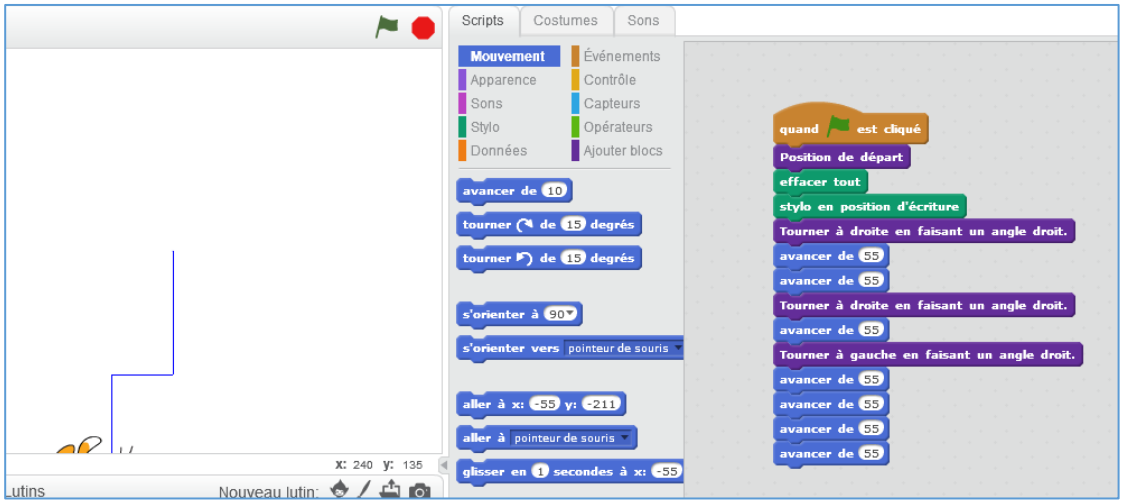


Figure 3-11 – Production finale de l'élève T

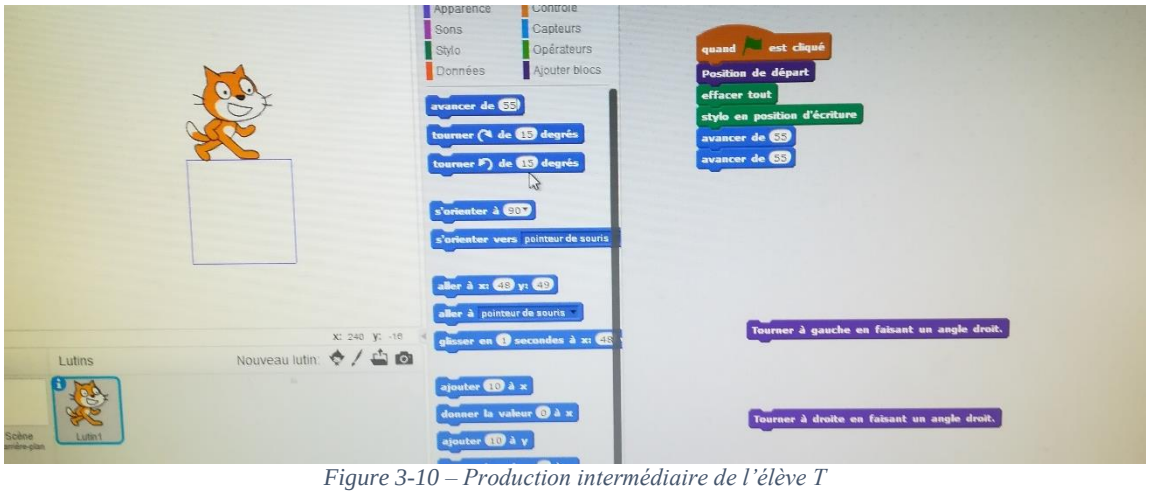


Figure 3-10 – Production intermédiaire de l'élève T

Sur la production intermédiaire de l'élève T, on peut aussi observer que la figure tracée sur

la gauche de l'écran ne correspond pas non plus au programme construit à droite de l'écran. Si on exécute le programme, la figure qui se dessine est un segment de 110 pas. Comme pour l'élève R, on peut remarquer des blocs détachés du programme (« Tourner à gauche en faisant un angle droit. » et « Tourner à droite en faisant un angle droit. ») qui indiquent que lui aussi a cliqué sur les blocs indépendamment les uns des autres pour créer son carré.
J'ai pu observer pendant la séance que pour arriver à ce premier résultat, l'élève avait mis environ 10 min. Ensuite, j'ai apporté un étayage à l'élève et l'élève s'est vraiment retrouvé en difficulté pour créer cette fois-ci une séquence d'instructions. Malgré plusieurs tentatives, l'élève n'a pas réussi à obtenir un carré avec l'exécution de son programme final. De plus, sa production finale ne présente que 3 segments et ces segments ne sont pas tous de la même longueur. Pourtant, on sait à partir de sa production intermédiaire qu'il avait compris quelle figure on souhaitait obtenir.
On peut se demander d'où vient cette difficulté. Serait-elle dans la difficulté de se repérer dans la lecture d'un programme ? de ne pas savoir où se trouve l'erreur à modifier lorsque le programme ne fonctionne pas ?

Définir de trop petites distances dans le bloc "avancer de"

Lors de l'analyse a priori de la tâche à effectuer, nous avions anticipé le fait que l'élève pourrait penser que rien ne s'affiche à l'écran lorsqu'il créait des segments trop petits, cachés par le lutin. Cette situation est arrivée. Cependant, sans l'aide de l'enseignant, les élèves ont déplacé le lutin afin de voir ce qui était dessiné en dessous. Cela atteste d'une compréhension du programme car pour penser à déplacer le lutin, il faut déjà être persuadé que son programme a bien dessiné quelque chose à l'écran.

Les caractéristiques du carré

Pour réaliser cette activité sur Scratch, les élèves devaient maîtriser les caractéristiques du carré. Il faut tout de même préciser que s'ils respectaient la consigne, les élèves ne pouvaient pas faire d'erreur sur les angles du carré car ceux-ci étaient donnés par les blocs « Tourner [à gauche ou à droite] en faisant **un angle droit**. ». Ainsi, les élèves ne devaient porter une attention particulière que sur le nombre de pas qu'ils indiquaient dans le bloc « avancer de [nombre de pas] ». Nous allons donc réfléchir aux stratégies utilisées et aux erreurs réalisées concernant la définition du carré : « Tous les côtés d'un carré sont de même longueur. ».
Dans cette analyse des résultats, nous excluons 3 productions : celles des élèves H, Q et V sur lesquelles on ne peut observer au maximum qu'un segment sur le tracé final.

Dans un premier temps, nous allons trier les productions selon le critère « Les segments tracés sont tous de la même longueur. ».

Les segments tracés sont tous de la même longueur.	Elèves G/R/N/U/O/E/P/K
Les segments tracés **ne** sont **pas** tous de la même longueur.	Elèves T/L/X/I

Parmi les élèves qui n'ont pas fait des tracés de même longueur, aucun ne l'a fait en écrivant un nombre de pas différent dans 2 blocs « avancer ». Pour les élèves I et X, l'erreur provient de l'utilisation de blocs non autorisés par la consigne. Ensuite, pour les élèves T et L, les longueurs différentes sont dues à la répétition de plusieurs blocs « avancer » pour créer un segment, le nombre de blocs répété n'étant pas toujours le même pour chaque segment.
Nous allons maintenant nous intéresser aux différentes stratégies utilisées par les élèves qui ont tous leurs segments de même longueur. On peut caractériser deux grands types de production. Pour cela, nous allons prendre l'exemple de l'élève O et de l'élève P.

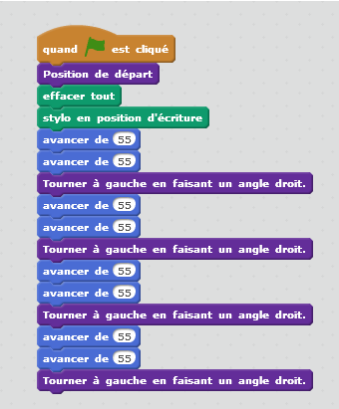


Figure 3-13 – Programme final de l'élève O

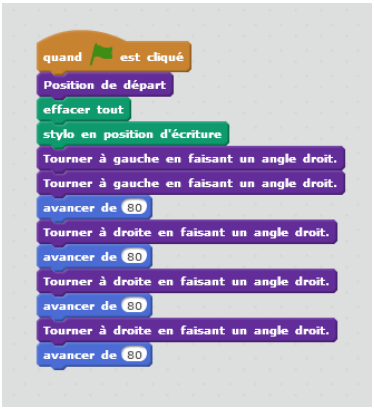


Figure 3-12 – Programme final de l'élève P

L'élève O : Pour tracer un segment plus long que celui de base (le bloc est « avancer de 10 » par défaut), l'élève a changé la valeur du nombre de pas. Il utilise ensuite systématique des blocs « avancer » avec ce même nombre de pas. Cependant, pour faire un segment encore plus grand, il décide de se faire succéder 2 blocs « avancer » identiques. Il place exactement le même nombre de blocs « avancer » entre chaque phase de rotation du lutin ce qui lui permet d'avoir toujours des segments de même longueur.
L'élève P : Il définit le nombre de pas souhaités dans un bloc « avancer » et écrit toujours cette même valeur dans chaque bloc « avancer » qu'il place dans son programme. Ici, un segment correspond à l'action d'un seul bloc « avancer ».

3.1.4. Gestion du temps

De par la gestion de classe difficile et l'autonomie très partielle des élèves face à la tâche à accomplir, je n'ai pas eu le temps de faire la mise en commun pendant cette séance.

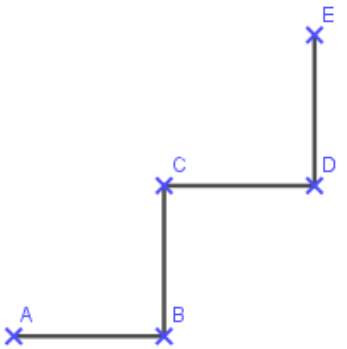
Ainsi, j'ai décidé d'ajouter une séance, en classe, avec un vidéo projecteur pour réaliser la mise en commun. Il m'a paru important d'ajouter cette séance afin de bien reprendre avec les élèves la compréhension du programme en tant que suite d'instructions qui se déroule les unes après les autres. On a repris un programme d'un élève qui avait réussi à construire le carré et nous avons décortiqué son programme afin de mettre en relief les blocs « inutile ». Les élèves, très actifs pendant cette séance, ont voulu tester de nouvelles choses (ajout de bloc(s), suppression de bloc(s)). A chaque nouvelle demande, je demandais systématiquement de prévoir ce que le programme allait faire si on réalisait ce que demandait l'élève puis on vérifiait en exécutant le programme. A la fin de cette séance, on a conclu que pour construire un carré avec Scratch de manière efficace, il ne fallait utiliser que des blocs qui tournaient dans le même sens et des blocs « avancer » avec le même nombre de pas à chaque fois. Enfin, ils ont gardé comme trace écrite l'exemple d'un programme d'un carré.

3.2. Séance 2 – Première partie de l'activité – Anticipation d'un programme

Nous allons maintenant analyser la deuxième séance. Nous commencerons dans cette partie par traiter la première phase de l'activité où il s'agissait pour l'élève de prévoir ce qui allait être tracé lors de l'exécution d'un programme donné. L'ensemble des productions des élèves se trouve en annexe 2.

Tout d'abord, pour prévoir ce qui allait être tracé par le programme, l'élève avait le choix entre décrire avec des mots la figure obtenue ou la tracer à main levée. Sur les 23 élèves présents ce jour-là, un seul a nommé la figure, tous les autres n'ont fait que des tracés à main levée.

Il convient de rappeler ci-contre la figure attendue sur laquelle nous avons nommé les segments pour faciliter l'analyse des productions obtenues.



Il est important de noter que dans l'analyse qui va suivre, certaines productions peuvent répondre à plusieurs critères puisque de nombreux élèves ont réalisé plusieurs tracés.

Les deux premiers critères que nous allons utiliser pour classer ces productions sont :

- **Critère 0** : Le tracé est une ligne non polygonale.

L'élève C semble hésiter entre segments et lignes courbes.

- **Critère 1** : Le tracé est une ligne polygonale fermée ou ouverte.

Cela concerne les 23 élèves car l'élève C a aussi produit une figure correspondant à une ligne polygonale. Nous allons différencier les élèves qui ont tracé une ligne polygonale fermée de ceux qui ont tracé une ligne polygonale ouverte.

1.1.1. Critère 1-a : Le tracé est une ligne polygonale fermée.

Au total, on peut observer des lignes polygonales fermées sur 8 productions. Les élèves A, B et D n'ont tracé que des lignes polygonales fermées.

Les élèves A, B, C, D, I, L, M ont construit **un quadrilatère ou plusieurs quadrilatères**, ce qui manifeste qu'ils ont repéré l'existence de 4 segments. Par ailleurs, leurs tracés montrent la prise en compte des 4 angles droits (les productions sont des carrés ou des rectangles). Cependant, ont-ils pris en compte dans leurs tracés à main levée la longueur égale des 4 côtés ?

L'élève D a tracé plusieurs carrés en prenant en compte l'égalité des longueurs. L'élève I a tracé 2 quadrilatères, l'un s'apparente à un carré, l'autre à un rectangle. Les élèves qui ont tracé un carré l'ont fait comme s'ils avaient a priori interprété le texte comme « tracer un carré de côté 100 », sans tenir compte de la consigne relative au programme de Cléa.

L'élève A a tracé 2 rectangles et l'élève C un. On peut être sûr que l'élève B a construit un rectangle puisqu'il l'a caractérisé ainsi à l'écrit. On peut supposer que ces élèves ont compris que Cléa s'était trompée dans son programme qui aurait dû permettre de tracer un carré (à partir de la phrase « Corrige le programme pour qu'il permette de tracer un carré »). Ensuite, ils ont cherché une autre figure connue qui n'était pas un carré : le rectangle.

Enfin, les élèves L et M ont tracé un carré en plus de lignes polygonales ouvertes. On peut se demander si la présence de ce carré n'est pas plutôt le reflet de leur recherche pour la phase de correction du programme. En effet, le tracé du carré attendu peut aider à élaborer le programme puisque l'élève peut prendre appui dessus pour identifier la séquence des étapes, même s'il ne les rédige pas.

La dernière production présentant une ligne polygonale fermée est un octogone non-régulier qui comporte des angles droits (Figure 3-14 – Elève O). On peut se demander si cette ligne polygonale fermée correspondrait à la volonté de « fermer » sa figure sur la droite composée de 7 segments où, semble-t-il, les longueurs sont voulues égales.

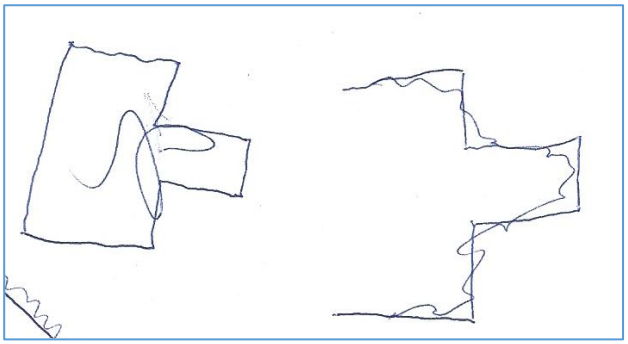


Figure 3-14 – Octogone non-régulier (Elève O)

1.1.2. Critère 1-b : Le tracé est une ligne polygonale ouverte.

Parmi les 23 élèves, 19 ont proposé au moins une ligne polygonale ouverte.

Nous allons classer ces productions selon le nombre de segments qui composent la figure.

Nombre de segments	1	2	3	4	6	7
Productions	I	F	H, I, M	E, F, G, J, O, P, Q, R, S, T, U, V, W	L, M, N?, O,	K, M, O

- Analyse des productions comportant 4 segments :

Parmi les 13 élèves ayant construit une ligne polygonale ouverte à 4 segments, 11 n'ont fait qu'un essai et 4 de ces 11 élèves ont obtenu la figure attendue : les élèves T, U, V et W.

Il n'est pas toujours aisé de savoir à partir de quel point l'élève a réalisé sa construction. Même si celui-ci peut se deviner selon la position de la figure sur la feuille, cela ne reste qu'une supposition.

Les erreurs les plus courantes portent sur le sens de la rotation, ce qui semble être caractérisé par une difficulté à se décentrer. La difficulté à se décentrer est accentuée par le fait que le lutin est représenté de profil alors que la prise de vue du tracé réalisé par le programme est aérienne. Ainsi, si on considère le lutin de profil, « tourner à gauche » peut signifier tourner vers l'écran et donc avancer ferait « plonger » le lutin dans l'écran.

Un autre type d'erreur est lié à la position de départ du lutin, ce qui semble être caractérisé par une méconnaissance ou une mauvaise interprétation du bloc « Initialiser ».

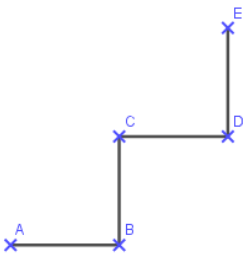
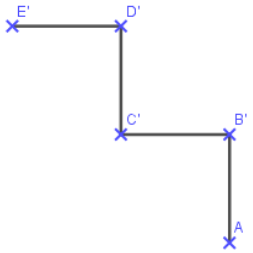
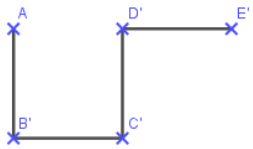
Nous allons maintenant présenter ces différents types d'erreurs en nous appuyant sur les productions des élèves.

❖ **Méconnaissance du bloc « Initialiser »**

Le bloc « initialiser » permet d'orienter le lutin à 90° autrement dit celui-ci sera tourné vers la droite. Ainsi, le premier bloc « avancer » fait réaliser au lutin une translation vers la droite de 100 pas. Les élèves S et E n'ont pas réalisé ce premier tracé vers la droite. Ainsi, on peut supposer que l'erreur vient d'une compréhension erronée du bloc « initialiser ». L'élève S part vers le haut et l'élève E vers le bas.

Malgré cette erreur d'orientation du lutin au départ, on peut remarquer que l'élève S ne fait pas d'erreur de rotation dans la réalisation de son tracé. Ainsi, la ligne polygonale AB'C'D'E' est l'image de la ligne polygonale ABCDE par la rotation de centre A et d'angle 90° dans le sens contraire des aiguilles d'une montre.

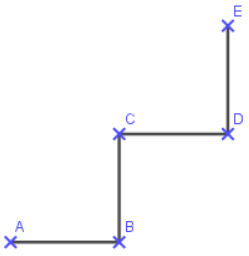
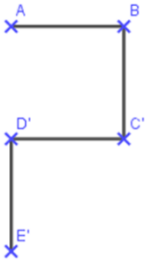
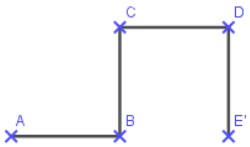
L'élève E, par contre a réalisé 2 erreurs de rotation à partir du point C'. En effet, à partir du point C', il tourne à gauche au lieu de tourner à droite et au niveau du point D', il tourne à droite au lieu de tourner à gauche.

Figure attendue	Figure construite par l'élève S	Figure construite par l'élève E
		

❖ **Erreur de rotation sur un seul segment.**

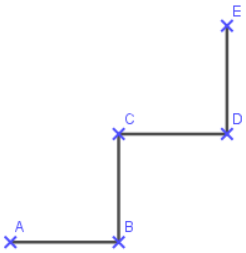
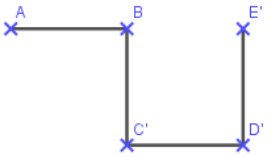
Les élèves F, P, Q et R n'ont fait qu'une seule erreur de rotation. Mais cette dernière ne se situe pas même endroit.

L'élève F fait une erreur de rotation dès le point B. Il tourne à droite au lieu de tourner à gauche. Ensuite, ses rotations sont correctes. Les élèves P, Q et R font une erreur de rotation sur le dernier segment : à partir du point D, ils tournent à droite au lieu de tourner à gauche.

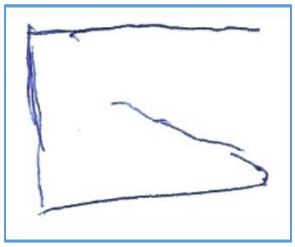
Figure attendue	Figure construite par l'élève F	Figure construite par les élèves P, Q et R
		

❖ **Erreurs de rotation sur deux segments.**

A partir du point B, l'élève G tourne à droite au lieu de tourner à gauche pour construire le point C. A partir du point C, il tourne à gauche au lieu de tourner à droite pour construire le point D. Enfin, à partir du point D, il respecte le sens de rotation en tournant à gauche.

Figure attendue	Figure construite par l'élève G
	

- <u>Cas particulier de l'élève J :</u>



La production de l'élève J présentée ci-contre présent une ligne polygonale ouverte, composée de 4 segments, cependant, tous les angles formés ne sont pas des angles droits. En effet, le dernier angle formé est un angle aigu. On peut supposer ici que l'élève avait compris à partir de la consigne que le programme présenté ne pouvait pas permettre de construire un carré (« Corrige le programme pour qu'il permette de tracer un carré. ») et donc se serait retrouvé bloqué au moment de tracer son dernier segment afin que celui-ci ne ferme pas le carré. Il aurait donc tracé un segment en suivant un autre angle.

- <u>Cas des élèves qui ont tracé plus de 4 segments :</u>

Les élèves K, L, M, N et O proposent une ligne polygonale présentant 6 ou 7 segments.

On peut se demander d'où vient cette erreur. Celle-ci pourrait par exemple être due à une difficulté pour les élèves à se repérer dans la lecture du programme. Face à cette incapacité à identifier à quelle étape ils étaient au moment de la lecture, ils ont pu être amenés à répéter une action ou à ne pas réaliser les instructions dans le bon ordre.

Cette erreur pourrait aussi venir du fait que les élèves prennent uniquement en compte le nombre de blocs. Ainsi, le bloc « tourner à gauche » pourrait signifier pour eux « tourner gauche tout en avançant ». Cela est particulièrement possible pour les productions où sont présents 7 segments. En effet, on peut voir sur le programme de Cléa (figure 3-16) qu'après le bloc « initialiser » sont présents 7 blocs de mouvement (soit tourner, soit avancer). Pour illustrer cela, on peut s'appuyer sur la production de l'élève M (Figure 3-15) car si l'élève considère que « Tourner à [gauche ou à droite] » signifie « Tourner [à gauche ou à droite] en avançant » alors toutes les actions sont réalisées par cet élève.

Figure 3-16 – Programme de Cléa

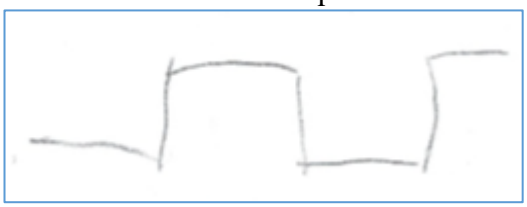


Figure 3-15 – Une des productions de l'élève M

Pour conclure concernant cette tâche d'anticipation d'un programme, on peut se rendre compte que les élèves montrent des difficultés dans la lecture d'un programme. Ces difficultés sont liées à la capacité d'abstraction mais surtout à l'habileté à se décentrer.

3.3. *Séance 2 – Correction du programme de Cléa sur Scratch*

Cette dernière partie de résultats concerne la troisième phase de la tâche de la deuxième séance. Les élèves ont alors déjà anticipé la figure qui allait se dessiner avec le programme de Cléa et ils ont vérifié leurs prévisions à l'aide du logiciel. Il s'agit maintenant pour eux de corriger le programme pour qu'il permette de tracer un carré. Pour vous présenter les résultats de cette activité, nous allons d'abord évoquer la gestion de classe et le recueil de données. Ensuite, nous évoquerons les résultats concernant les productions finales des élèves. Puis, nous nous intéresserons à la manière dont les élèves ont corrigé le programme. Enfin, nous étudierons plus en détail les différentes étapes d'élaboration du programme corrigé de deux élèves afin de mettre en lumière les stratégies d'autorégulation mises en œuvre par ces deux élèves.

3.3.1. Gestion de la classe et recueil des données

Durant cette séance, la gestion de classe pendant l'activité sur les ordinateurs a été plus aisée que lors de la séance 1 car je n'avais que la moitié de la classe. L'autre moitié était en séance en bibliothèque. Ensuite, les groupes ont été intervertis.

Concernant le recueil des données, il y a tout d'abord les productions finales des élèves que nous évoquerons brièvement. Ensuite, pour cette nouvelle séance, j'avais prévu d'installer un autre logiciel de capture vidéo que j'avais testé chez moi sur les ordinateurs de la salle informatique mais je n'ai jamais réussi à les installer avant la séance. Cependant, comme cette fois-ci j'étais au courant de ce problème avant le début de la séance, j'ai pu mettre en place un autre dispositif de recueil de données. Pendant la séance, j'ai donc suivi seulement quelques élèves en prenant très régulièrement des captures d'écrans de leurs étapes de réalisation. Pour les élèves que je n'ai pas suivre aussi précisément, je me suis concentrée sur un critère d'observation : le programme de Cléa a-t-il été corrigé ou complètement supprimé pour créer un nouveau programme ? Pour repérer cela, j'ai à la fois observé ce qu'ils faisaient mais je leur ai aussi posé la question « Comment as-tu fait pour corriger le programme de Cléa ? » et j'ai noté leurs réponses.

3.3.2. Les productions finales des élèves

Nous n'analyserons pas de manière fine les productions finales des élèves comme cela a pu être réalisé pour les productions de la première séance. Cependant, quelques remarques sont tout de même à noter.

Tout d'abord, on peut constater une réelle évolution dans la capacité des élèves à réaliser le programme du carré. En effet, sur les 21 productions finales que j'ai recueillies 19 élèves ont réalisé un programme permettant de tracer la figure attendue : un carré. En annexe 3, vous pourrez constater l'évolution d'un élève entre sa production finale de la séance 1 et celle de la séance 2.

Ensuite, on peut remarquer que élèves ont bien retenu pour que pour tracer un carré, il ne fallait « tourner » que dans un seul sens. En effet, sur les 21 productions, 16 élèves n'ont utilisé que des blocs « tourner » de même direction.

Nous pourrions évoquer d'autres résultats concernant les productions finales mais cette séance est particulièrement intéressante concernant les comportements des élèves à analyser que allons développer dans les deux parties suivantes.

3.3.3. Les stratégies des élèves afin de corriger le programme

Lors de cette séance il s'agissait d'observer le comportement des élèves face à une erreur dans un programme. Deux comportements étaient envisagés : repérer l'erreur dans le programme et corriger cette erreur ou effacer tout le programme et recommencer tout de zéro. Il n'y a pas eu de comportement plus prépondérant qu'un autre. La moitié de la classe environ a corrigé l'erreur, l'autre a tout recommencé. Concernant les élèves qui ont modifié

le programme, certains élèves ont repéré une erreur (citation de l'élève Q ci-dessous) quand d'autres en ont repéré deux (citation de l'élève A ci-dessous). En effet, soit l'élève comprenait qu'il fallait remplacer le bloc « Tourner à droite » par un bloc « Tourner à gauche », soit il changeait les deux blocs « Tourner à gauche » pour des blocs « Tourner à droite ». Enfin, des élèves ont remarqué que c'était à partir du bloc « Tourner à droite » que le programme ne permettait plus de dessiner un carré alors ils ont effacé tous les blocs à partir de celui-ci et ont réécris la fin du programme.

« Pour faire le programme du carré, j'ai enlevé [le bloc tourner à] droite et j'ai mis [le bloc tourner à] gauche à la place. » Elève Q.

« J'ai enlevé les deux [blocs] tourner à gauche et j'ai changé avec des [blocs] tourner à droite. » Elève A.

Durant cette deuxième séance, j'ai pu observer plus de réflexion en amont de la part des élèves. On peut supposer que cela était dû à la tâche demandée. En effet, puisqu'il avait déjà un programme à l'écran (le programme de Cléa), ils ne partaient pas de rien et je pense que pour certains cela les a encouragés à prendre plus de temps avant de se lancer. Par exemple, l'élève R a dans un premier temps chercher à identifier où était exactement l'erreur. Pour cela il m'explique qu'il s'est décentré afin d'appliquer le programme sui lui-même et comprendre ce qu'il se passait à chaque étape (citation de l'élève R ci-dessous).

« J'ai fait comme si j'étais sur la feuille, comme si c'était moi qui écrivait le carré à la place du petit chat. » Elève R.

Enfin, un autre exemple de réflexion préalable à l'action sur le logiciel est l'analyse de la figure erronée par l'élève K. En effet, comme elle l'a indiqué grâce à la flèche dessinée à côté de sa figure sur papier, elle a compris que la figure était bonne à une rotation près de deux segments. Elle m'a expliqué que « Si on déplaçait ça [c'est-à-dire les deux derniers segments], ça ferait un carré. ». Cependant, même en ayant bien identifié cela, elle n'a pas corrigé le programme mais a tout effacé et a recommencé de zéro. Cela peut s'expliquer par la difficulté à repérer où se

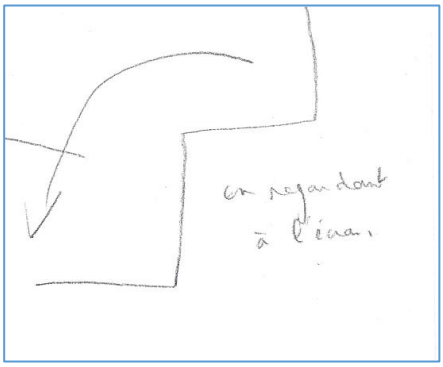


Figure 3-17 - Figure présente sur la production de l'élève K

situait cette erreur de rotation dans le programme. La difficulté est donc liée ici au nouveau langage utilisé (le langage de programmation Scratch) permettant de construire une figure en géométrie.

3.3.4. Deux exemples de réalisation – les stratégies d'autorégulation

Sur Scratch, les élèves sont amenés à travailler par essais-erreur. En effet, ils ont un retour direct sur la validité de leur production et donc sont amenés à la modifier lorsqu'il y a des erreurs.

Dans cette partie, nous nous concentrerons sur les différentes étapes de réalisation de deux élèves. Ces deux élèves sont tous les deux arrivés à la construction d'un carré en élaborant leur travail par phases d'essais-erreurs. Cependant ils ont procédé par tâtonnement expérimental de deux manières différentes. L'élève P a construit son programme d'un seul coup et ne l'a testé qu'ensuite pour vérifier s'il était correct alors que l'élève H a testé son programme au fur et à mesure de la conception. Le comportement de l'élève H met en relief une des limites de l'utilisation du logiciel que nous avions évoqué dans la partie théorique : une réflexion préalable incomplète. L'élève n'a pas de réflexion préalable à la construction d'un programme dans son ensemble. Cependant, cela reflète aussi une capacité pour l'élève H à subdiviser sa tâche en plusieurs sous-tâches.

Nous allons donc maintenant étudier leurs deux cheminements plus en détails.

- Elève P :

Vous retrouverez en annexe 4 les étapes de réalisation du programme de l'élève P sur lesquelles je vais m'appuyer maintenant. J'ai réalisé 4 captures de son écran à des moments successifs. Tout d'abord, on peut voir que l'élève ne supprime pas les blocs présents dans le programme de Cléa car il compte s'en reservir par la suite. Il les conserve donc sur le côté. Il construit le nouveau programme d'un seul coup. On peut être sûr de cela car sur la fenêtre de gauche on peut voir la dernière exécution d'un programme qui correspond à l'exécution du programme de Cléa. Sur la deuxième capture d'écran, on peut voir qu'il ajoute un nouveau bloc « Touner à gauche » qu'il place directement en dessous du bloc « Initialiser » dans son programme. Ensuite, il réutilise tous les blocs du programme de Cléa dans le même ordre. A ce moment là, lorsqu'il pense avoir terminé son programme, il le teste (capture d'écran n°3) et se rend compte que la figure dessinée par le lutin n'est pas celle attendue. A ce moment là, j'ai pu observer un temps de reflexion chez l'élève P. Ensuite, il modifie son programme en remplaçant le bloc « tourner à droite » par un bloc « tourner à gauche ». Enfin, il teste à nouveau son programme et obtient la figure attendue

alors il s'arrête. Il a donc réalisé les stratégies d'auto-régulation présentées dans la partie théoriques (anticipation, contrôle, ajustement) afin d'arriver à la construction du carré.

- Elève H :

Les étapes de réalisation du programme de l'élève H se trouvent en annexe 5. J'ai réalisé 8 captures d'écran. J'ai quasiment pris une capture d'écran à chaque phase de contrôle du programme par l'élève H, excepté au début de la réalisation de son programme. Au début, tout comme l'élève P, l'élève H ne supprime pas tous les blocs correspondant au programme de Cléa. Cependant, il ne cherche pas non plus à modifier l'erreur de Cléa puisqu'il est reparti de zéro dans la conception de son programme. On peut voir d'ailleurs que ce comportement se répète après la capture d'écran n°2. En effet, il se rend compte dès la deuxième capture que son programme ne permettra pas de dessiner un carré, alors il détache tous ses blocs et recommence. Ainsi, jusque là, l'élève ne réalise pas de phase d'ajustement après ses phases de contrôle et pas vraiment d'anticipation avant ses actions. Ceci illustre une des limites évoquées quant à l'utilisation du logiciel Scratch. Ensuite, l'élève change d'attitude, planifie sa tâche et prend la décision de la subdiviser en 4 : chaque étape dans la réalisation de sa tâche correspond à la création d'un segment. Le plus souvent, chaque étape correspond à l'ajout de deux blocs (« Tourner à [gauche/droite] » et « avancer de 100 ») sauf une fois (capture d'écran n°5) où l'élève ajoute trois blocs (deux blocs tourner et un bloc avancer). On se rend compte alors que l'élève réalise pour la création de chaque segment, une phase d'anticipation, de contrôle et d'ajustement éventuel. Par exemple, pour la création du troisième segment du carré, l'élève ajoute d'abord une séquence de trois blocs : « Tourner à gauche » - « Avancer de 100 » - « Tourner à gauche ». Ensuite, il exécute son programme et obtient la figure présente sur la capture d'écran n°5. Il se rend compte de son erreur, détache les trois blocs qu'il vient d'accrocher et ajuste son travail en ajoutant finalement une séquence de deux blocs : « Tourner à droite » - « Avancer de 100 ». Il teste à nouveau son programme et obtient la construction attendue de son troisième segment. Il peut alors passer à la quatrième étape de sa tâche, c'est-à-dire la construction du quatrième et dernier segment. Dans son travail, on peut tout de même se demander si les phases d'anticipation sont réellement présentes. En effet, la construction de chaque segment ne relevant que d'un choix entre deux blocs (« Tourner à gauche » ou « Tourner à droite »), on peut très bien imaginer que l'élève teste à chaque fois l'une des possibilités sans anticiper son résultat puis change le bloc si jamais le résultat n'est pas celui attendu. On peut aussi supposer que le grand nombre d'étapes réalisé montre une difficulté pour l'élève à se décentrer. Cela s'illustre bien au moment de la création de son dernier segment. Avant l'ajout de blocs pour créer son

dernier segment, il a obtenu la figure et la position du lutin ci-contre (figure 3-18). Ainsi, pour créer le quatrième segment, il faut que le lutin se déplace vers la gauche de l'écran. On peut donc supposer que c'est pour cela que l'élève H choisit alors d'ajouter la séquence « Tourner à gauche » - « Avancer de 100 ». Cependant, par rapport à la position relative du lutin, il faut que celui-ci tourne à droite pour construire ce dernier segment, ce qui demande une réelle capacité de décentration de la part de l'élève.

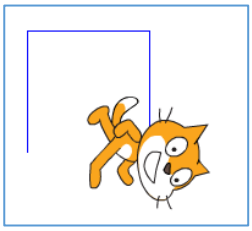


Figure 3-18 – Figure obtenue après la création du 3ème segment

4. Conclusion

L'enjeu de ce travail de mémoire était de réfléchir à des situations pertinentes permettant aux élèves de développer des compétences liées à la construction d'une pensée informatique. Un autre aspect de ce mémoire concernait l'appropriation par les élèves d'un langage de programmation, Scratch, ainsi que la gestion de l'erreur dans la conception de séquences d'instructions utilisant ce langage.

Nous nous étions demandé quelles situations un enseignant pouvait proposer pour favoriser le développement de la pensée informatique chez les élèves de cycle 3 en utilisant le logiciel Scratch mais aussi comment les élèves géraient l'erreur sur ce même logiciel.

Pour répondre à cela, nous avons d'abord exposé les aspects institutionnels puisque je le rappelle, l'initiation à la programmation est de nouveau présente dans les programmes en vigueur. De plus, ces programmes mettent en lumière un lien étroit entre l'initiation à la programmation et la construction de figures en géométrie. Ensuite, nous avons convoqué les résultats de recherche en termes de didactique et de pédagogie dans ce domaine, mais aussi dans le domaine de la géométrie et de la résolution de problème. Ces résultats de recherche nous ont permis de connaitre les éventuelles erreurs que les élèves seraient amenés à réaliser comme par exemple la difficulté de suivre un déplacement relatif à un personnage. Cela nous a aussi aidés à appréhender les différentes stratégies et comportements en jeu. Nous avons alors présenté une séquence, mise en place dans une classe de CM1 et permettant d'analyser les comportements, les stratégies et les erreurs des élèves face à des activités différentes de programmation.

Les résultats obtenus lors de la mise en place de la séquence laissent apparaitre quelques réponses quant à nos questionnements initiaux. Tout d'abord, concernant les situations pertinentes sur Scratch à mettre en œuvre, nous pouvons conclure que les activités de débogage sont très intéressantes. En effet, ce genre d'activité (comme celle réalisée par les élèves en séance 2), oblige réellement les élèves à plus anticiper leur action. Ce genre d'activité développe aussi la capacité de l'élève à décomposer la tâche à effectuer : repérage de l'erreur, modification du programme, test, etc. Ensuite, s'agissant de la gestion de l'erreur, on a pu observer que dans un premier temps (séance 1) les élèves ne progressaient pas dans la rédaction de leur programme par l'alternance de phases d'essais et d'erreur. Cela venait du fait qu'ils n'avaient pas forcément compris l'outil « drapeau vert » qui leur permettait de tester leur programme. Lors de la séance 2, on a pu remarquer que de nombreux élèves ont

cette fois-ci travaillé par phases d'essais-erreurs, en utilisant aussi les stratégies d'autorégulation (anticipation, contrôle, ajustement).

L'analyse des productions et des comportements d'élèves mise en parallèles avec les éléments de la partie théorique permettent de conclure sur des résultats marquant concernant principalement les difficultés des élèves.

Tout d'abord, une réelle difficulté de la part des élèves concerne la capacité de décentration. En effet, pour effectuer tous leurs tracés sur Scratch les élèves devaient programmer des séquences d'instructions qui provoquaient des déplacements relatifs à la position et à l'orientation du lutin. On a pu voir que de nombreux élèves, lors de la première phase de la séance 2 où ils devaient prévoir ce qu'allait tracer un programme donné, se sont exposés à cette difficulté. En effet, la grande majorité des erreurs étaient liées à de problèmes de rotations. Pour pallier cette difficulté, il peut être intéressant de travailler avec des robots programmables qui pourraient rendre les déplacements moins abstraits que sur Scratch.

Ensuite, ce travail de recherche permet de mettre ne valeur la difficulté de la tâche lorsqu'il s'agit de créer un programme dans son ensemble, en une succession d'instructions qui vont s'exécuter en une seule fois. Un élève peut très bien être capable de construire une figure en exécutant les instructions une par une et pour autant être mis en grande difficulté lorsqu'il s'agit de les assembler dans un ordre logique qu'est le programme.

Enfin un dernier résultat marquant porte sur les stratégies d'autorégulation et le manque d'anticipation de la part des élèves dans la création d'algorithmes sur Scratch. En effet, on a pu observer qu'une partie des élèves ne prenait pas le temps d'anticiper leurs actions sur Scratch, testant directement des séquences d'instructions sur le logiciel. De plus, lorsque les élèves mettent en place des stratégies d'autorégulation, s'ils découpent la tâche en des sous-tâches trop petites, cela peut empêcher une anticipation générale du programme. Alors comment les faire progresser en anticipation afin qu'ils réalisent des programmes entier qu'ils seront amenés à corriger ensuite ? Il me semble qu'une variable didactique intéressante serait le nombre de fois autorisé à exécuter son programme, c'est-à-dire un nombre limité de contrôle de son activité. Une autre variable didactique serait de proposer la construction de figures plus complexes où le contrôle étape par étape serait plus fastidieux et donc qui permettrait de faire progresser l'élève vers des stratégies plus efficaces. Cela aurait peut-être pu s'observer lors de la séance 3 qui n'a malheureusement pas pu être menée. Il me paraît donc intéressant, si une nouvelle recherche devait se faire sur ce sujet, de travailler sur ces deux variables didactiques afin de voir si elle permette vraiment à l'élève de progresser vers

des stratégies plus efficaces, qui a fortiori, constitueraient un développement de leur pensée informatique.

Un autre élément important sur lequel je souhaite conclure concerne mes « a priori » sur le logiciel Scratch. Avant la réalisation de cette séquence j'étais persuadée que Scratch était un logiciel de programmation très attrayant et instinctif pour les élèves. Au cours de ce mémoire, je me suis rendue compte que l'utilisation de Scratch n'était pas si naturelle pour les élèves, bien que ce logiciel reste très bien adapté pour initier les élèves en cycle 3. Finalement, maitriser quelques éléments du logiciel peut demander un temps d'appropriation important de la part de certains élèves. Par contre, il est nécessaire de préciser que le logiciel reste très attrayant aux yeux des élèves, surtout lorsqu'ils le découvrent. Lors des premières confrontations avec le logiciel, les élèves n'ont pas hésité à tester de nombreux blocs inconnus car l'interface de Scratch est ludique. Il est à noter aussi que lors des séances de Scratch, les élèves ont toujours montré une grande motivation dans la réalisation de leur tâche et l'envie d'utiliser le logiciel.

Enfin, je suis persuadée que ce travail de mémoire a beaucoup apporté à ma pratique professionnelle. Tout d'abord, l'élaboration de cette séquence m'a permis de comprendre l'importance de la maîtrise des savoirs didactiques avant de se lancer dans un apprentissage avec les élèves. Ensuite, travailler en pédagogie active m'a montré à quel point les élèves peuvent progresser lorsque l'enseignant prépare des situations propices à une forte activité des élèves et lorsqu'il sait laisser la parole et les échanges se faire par les élèves pendant ses séances. Enfin, j'ai beaucoup apprécié travailler avec Scratch cette année car ce logiciel a aussi été une entrée pour moi vers la pédagogie de projet puisque les élèves ont pu, grâce à Scratch réaliser un mini-dessin animé.

5. Bibliographie

Algorithimique et programmation (2016) Eduscol, Mathématiques, cycle 4.

Calmet C., Hirtzig M., Wilgenbus D. (2016). *1, 2, 3... Codez ! : Enseigner l'informatique à l'école et au collège (cycles 1, 2 et 3)*. Editions le Pommier.

Drechsler M. (22 mars 2017). *Initier les élèves au codage et à la programmation*, Canopé Académie de Rouen, http://classetice.fr/spip.php?article883

ERMEL. (2006). *Apprentissages géométriques et résolution de problèmes (cycle 3).* Hatier

Focant J. (2003). *Impact des capacités d'autorégulation en résolution de problèmes chez les enfants de 10 ans.* Association Canadienne d'éducation de la langue française.

Initiation à la programmation aux cycles 2 et 3. (2016) Eduscol, Mathématiques.

Jaillet A. (2001) *Un défi de taille pour l'école*. Entretien avec Seymour Papert, Les cahiers pédagogiques, n°398.

Mendelsohn P. (1985) L'analyse psychologique des activités de programmation chez l'enfant de CMI et CM2. In: *Enfance*, tome 38, n°2-3, 1985. L'ordinateur et l'écolier (II) : recherches expérimentales. pp. 213-221.http://www.persee.fr/doc/enfan_0013-7545_1985_num_38_2_2881

Ministère de l'Education nationale, de l'enseignement supérieur et de la recherche (2015) *Programmes d'enseignement de l'école maternelle*, *BO spécial du 26 mars 2015*

Ministère de l'Education nationale, de l'enseignement supérieur et de la recherche (2015), Décret n° 2015-372 du 31 mars 2015 relatif au socle commun de connaissances, de compétences et de culture.

Ministère de l'Education nationale, de l'enseignement supérieur et de la recherche (2015) *Programmes d'enseignement de l'école élémentaire et du collège*, BO spécial du 26 novembre 2015

Mokhtar Ben Henda. (2017) *L'enseignement du code informatique à l'école : Prémices d'un humanisme numérique congénital*. Chaire Unesco-ITEN. Actes de la 5e rencontre annuelle d'ORBICOM, Les éditions de l'immatériel.

Programmes et instructions à l'école élémentaire (1985) – Arrêté du 15 mai 1985

Programmes de l'école primaire (1995) – Instructions officielles du 22 février 1995

Tchounikine P, (25 mars 2017). *Initier les élèves à la pensée informatique et à la programmation avec Scratch*, Université Grenoble-Alpes.

Wing J. (2006) *La pensée informatique* (traduction), https://interstices.info/jcms/c_43267/la-pensee-informatique

6. Annexes

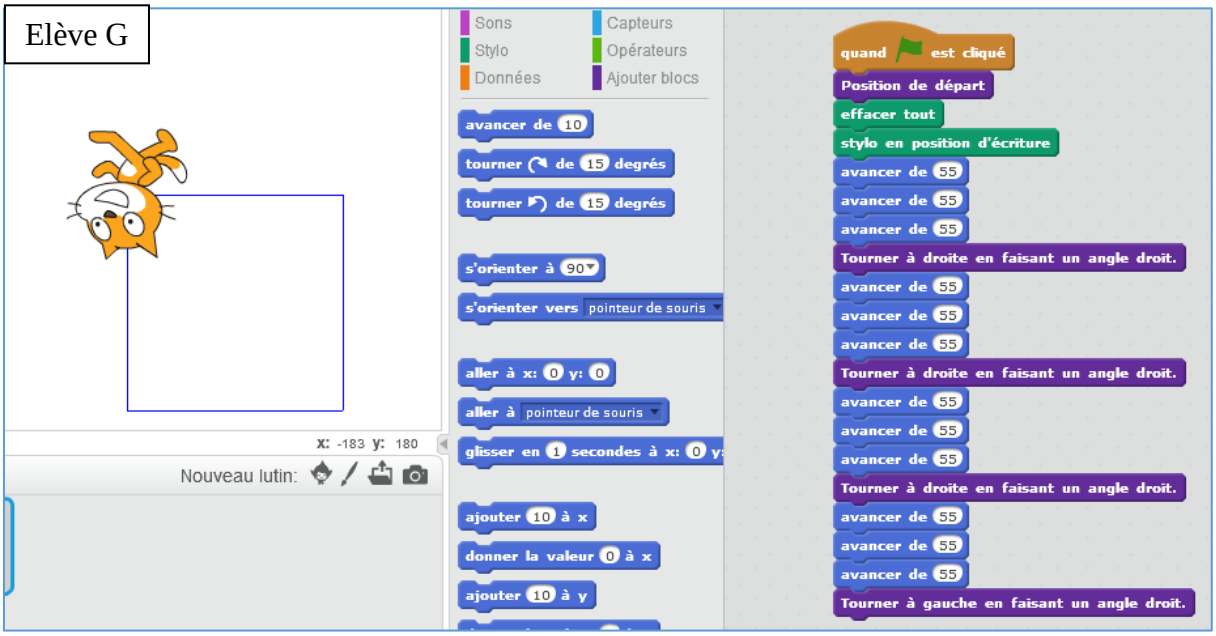
6.1. Annexe 1 – Productions finales de la séance 1

Chaque programme est exécuté en cliquant sur le drapeau vert avant la capture. Le lutin est éventuellement déplacé avant la capture pour bien voir le tracé lorsqu'il était caché.

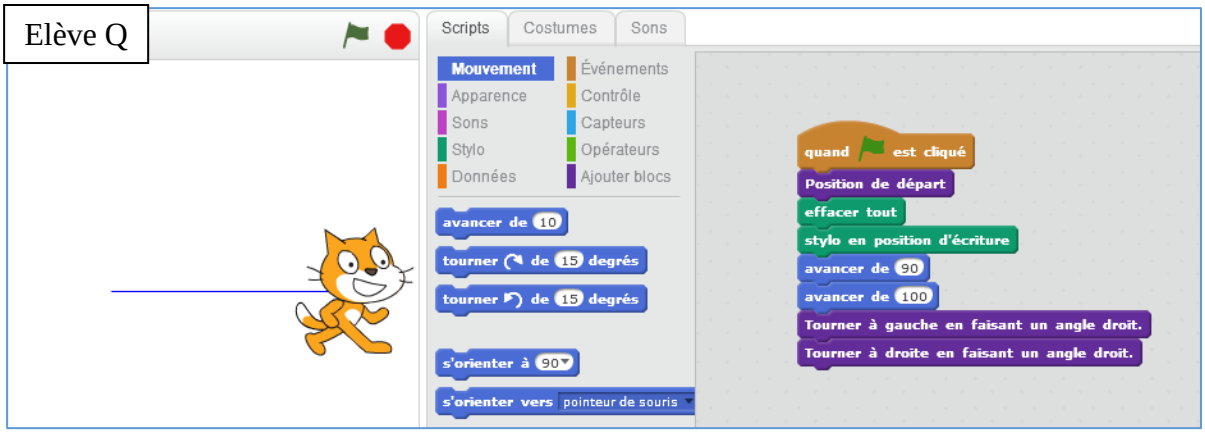
Elève T

Elève V

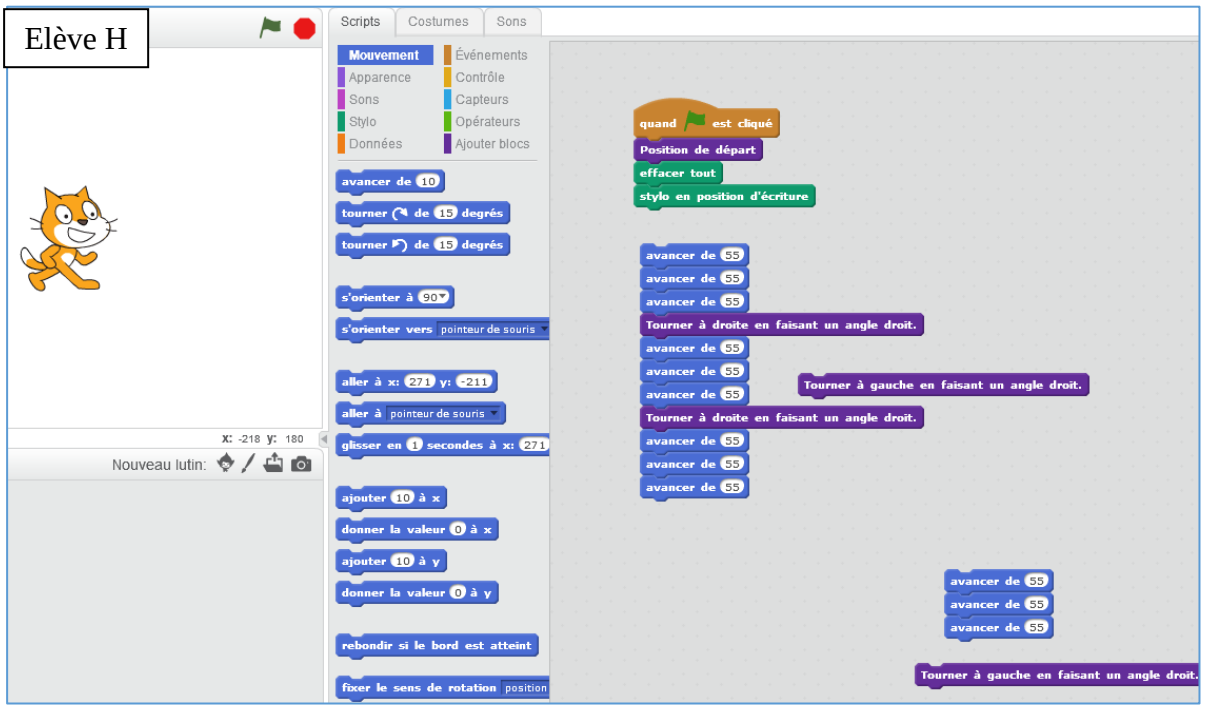
Elève G



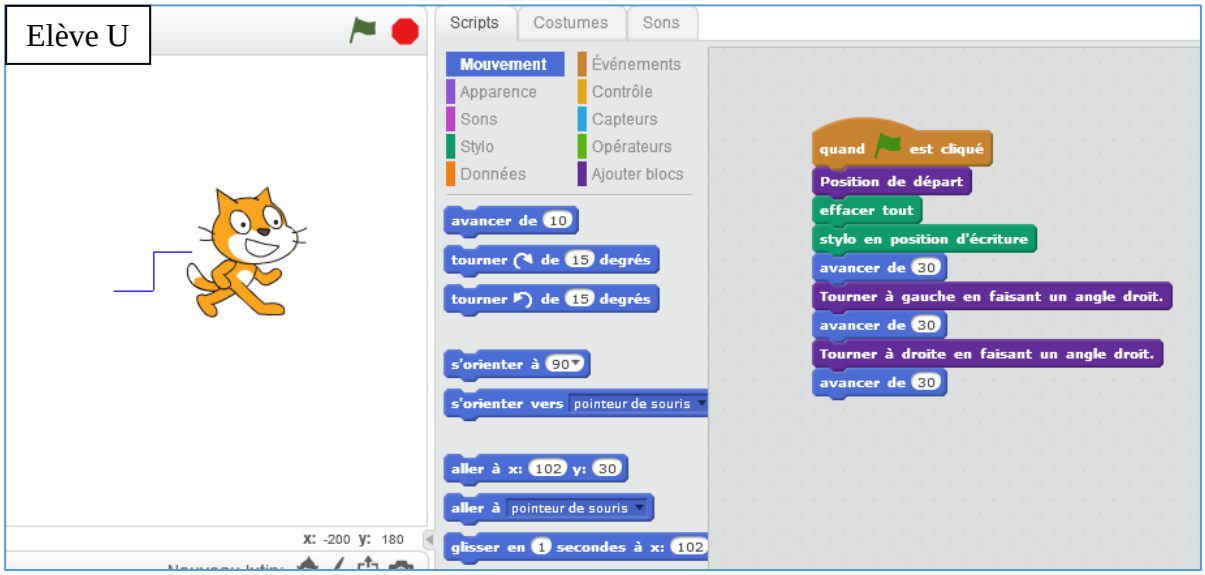
Elève Q



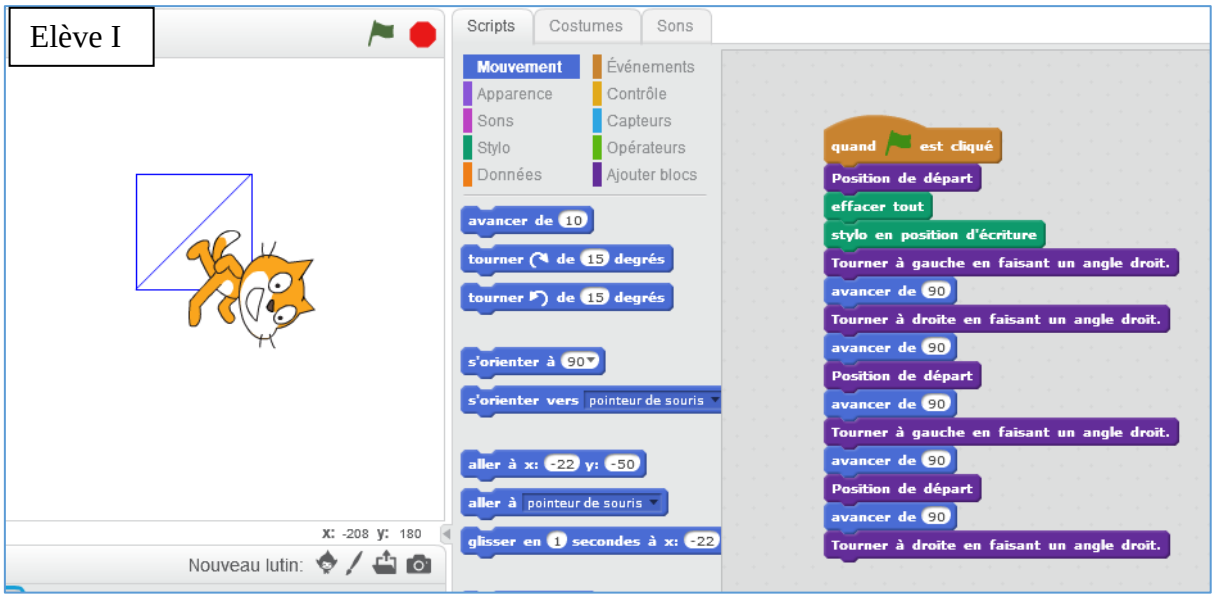
Elève H



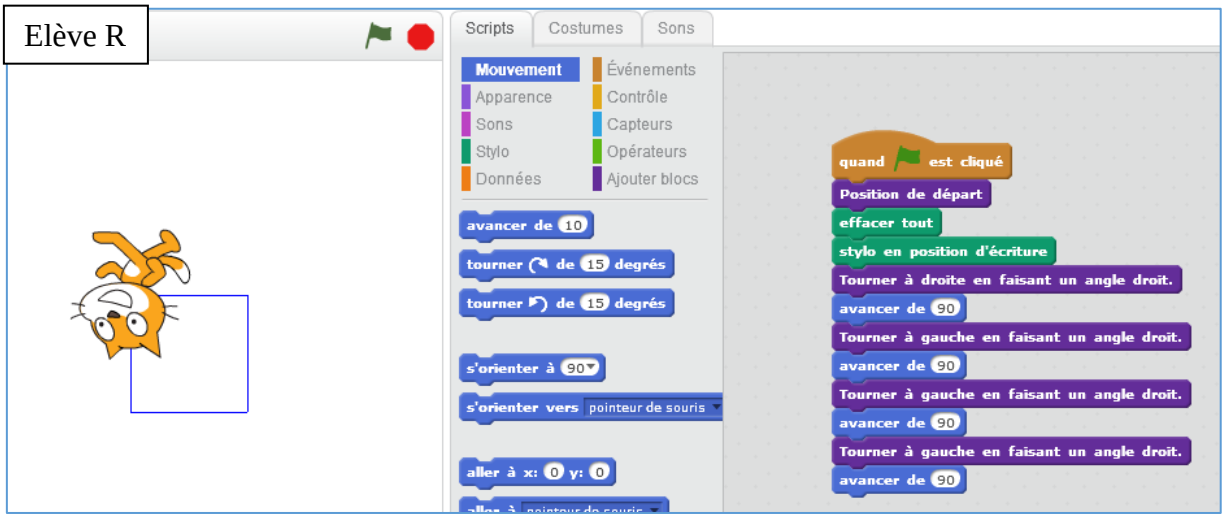
Elève U



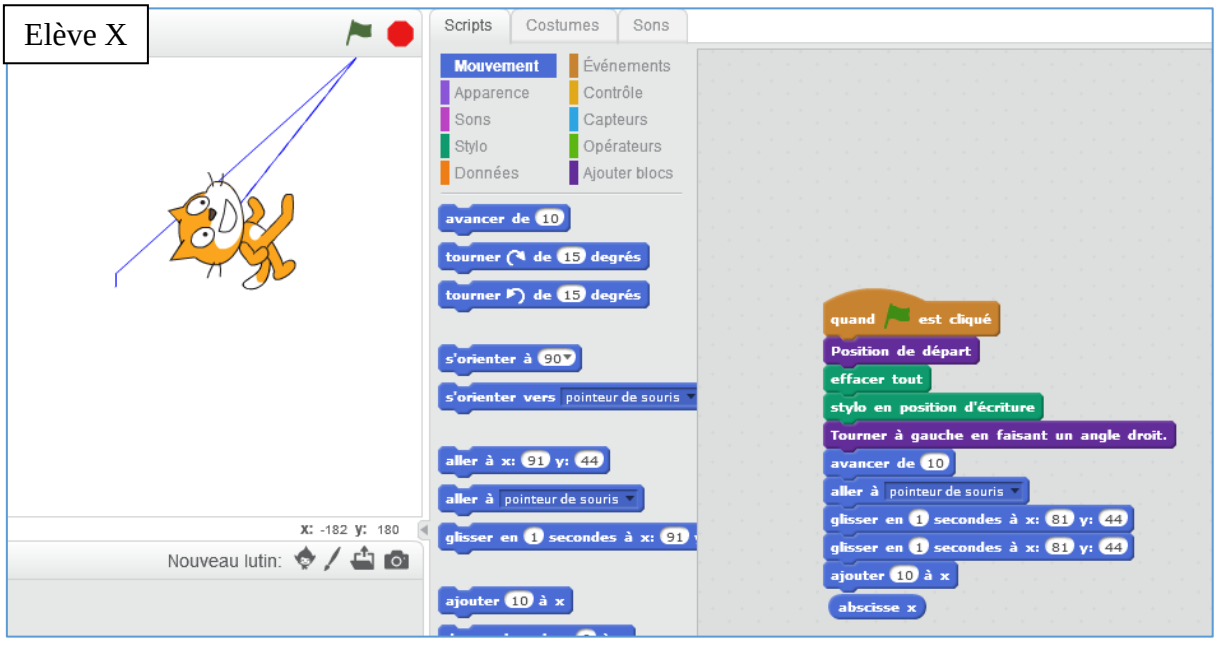
Elève I



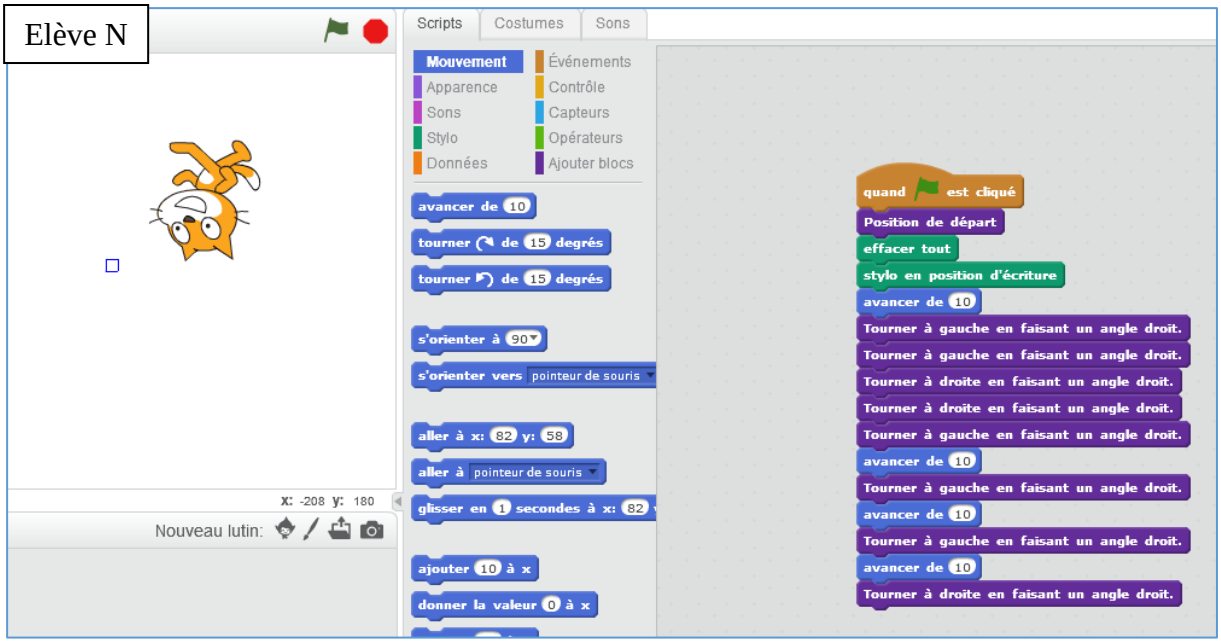
Elève R



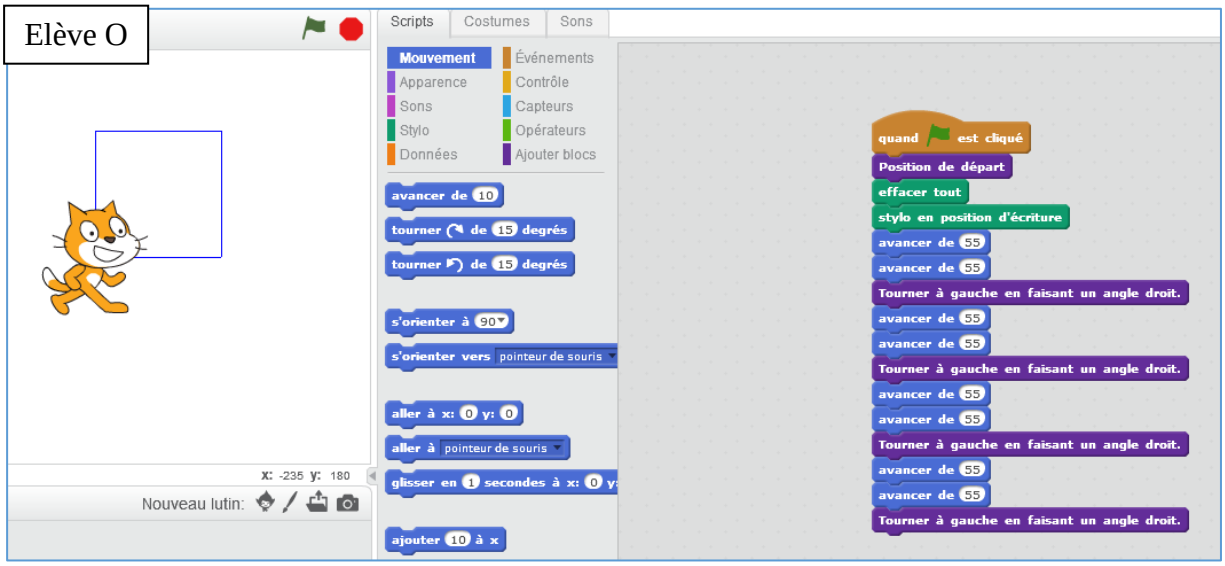
Elève X



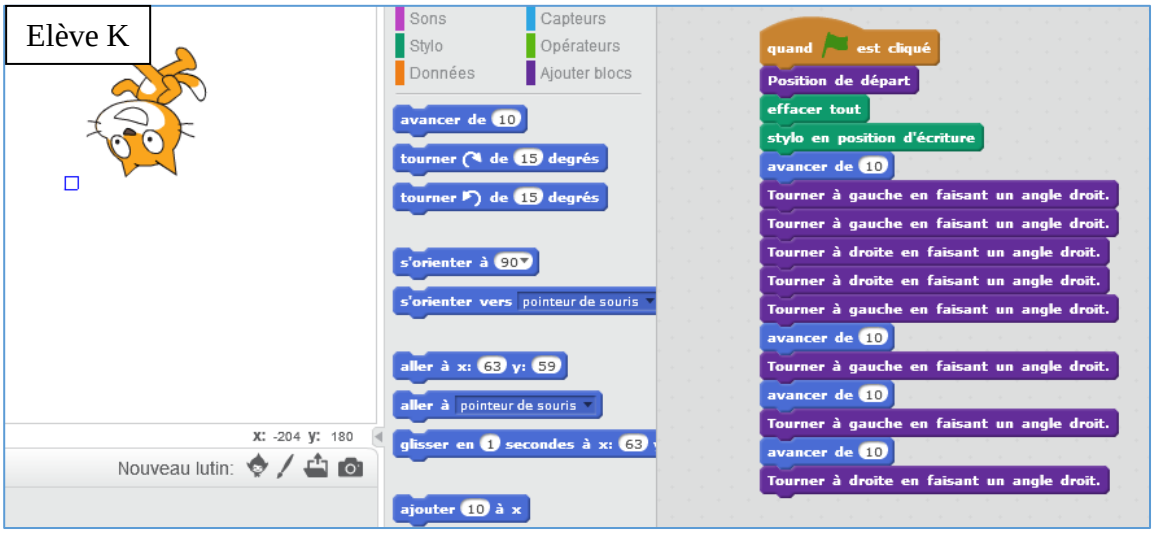
Elève N



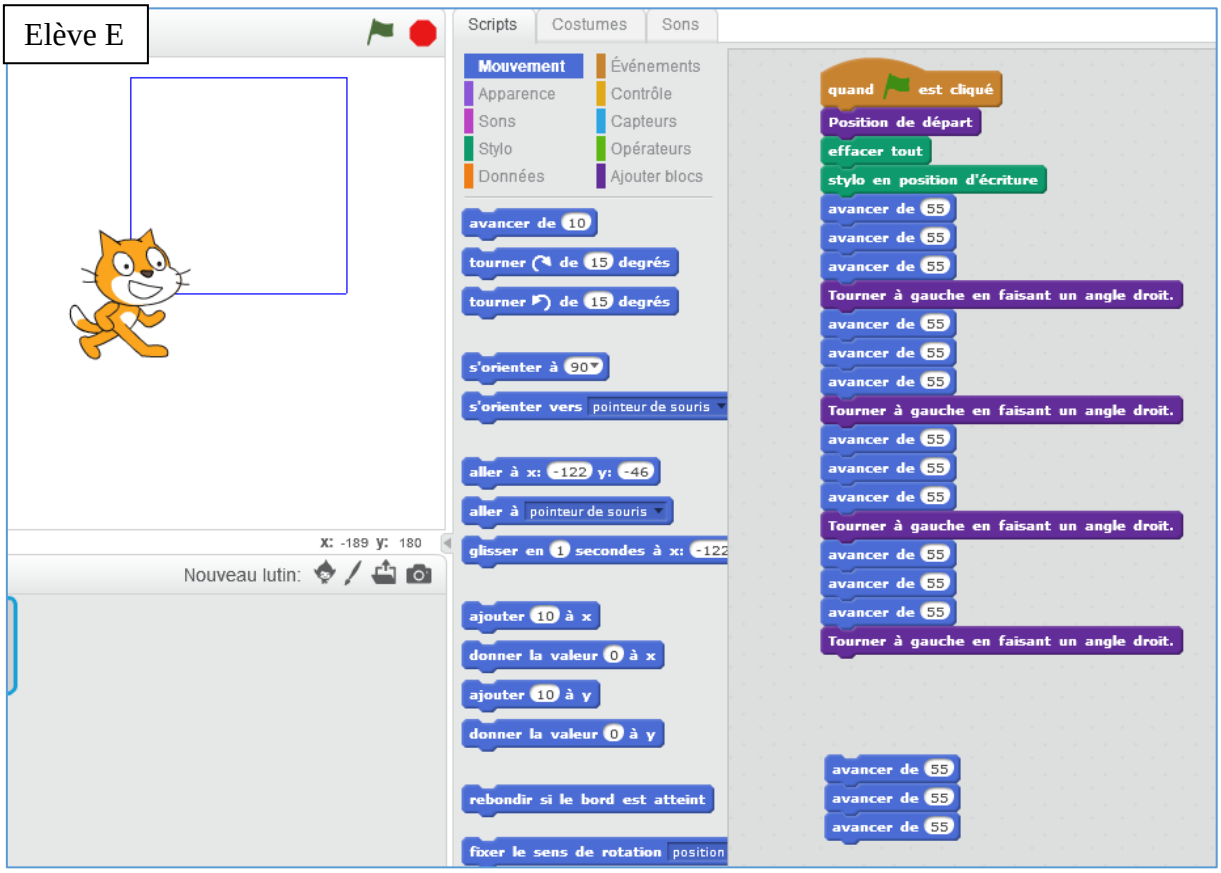
Elève O



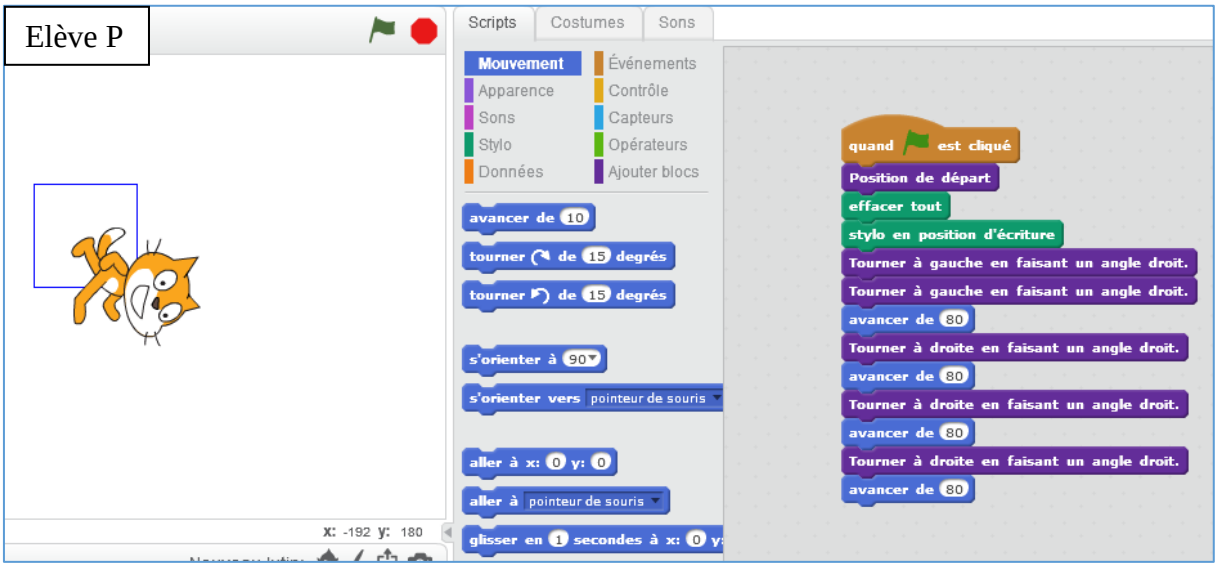
Elève K



Elève E



Elève P



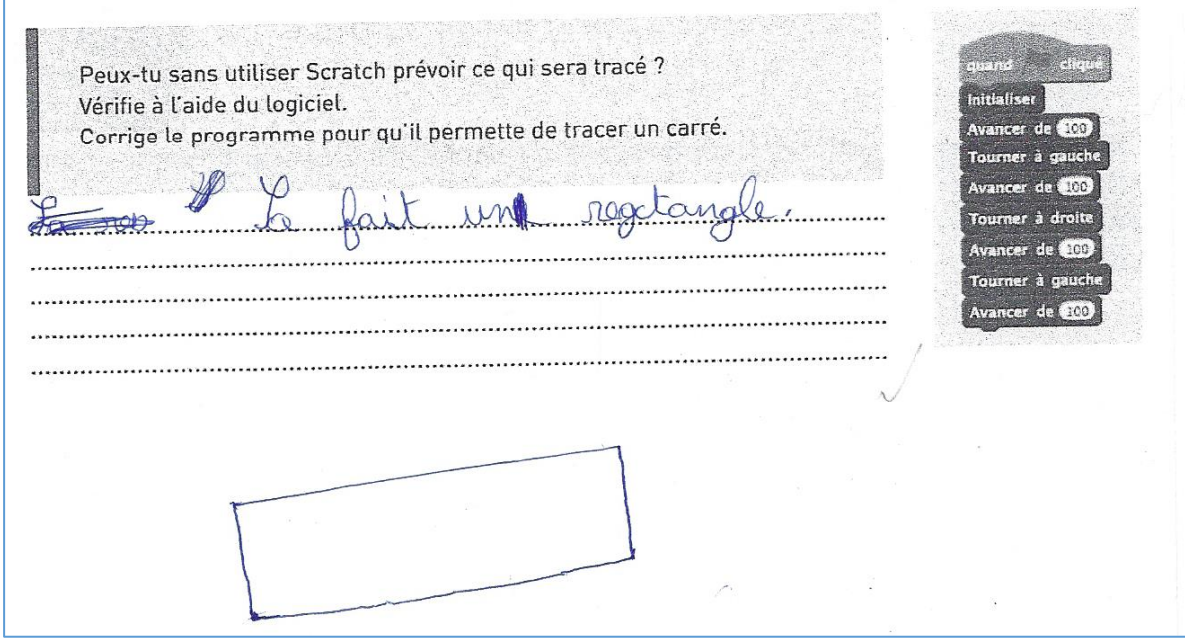
6.2. *Annexe 2 – Productions des élèves sur l'anticipation du programme de la deuxième séance.*

Elève B

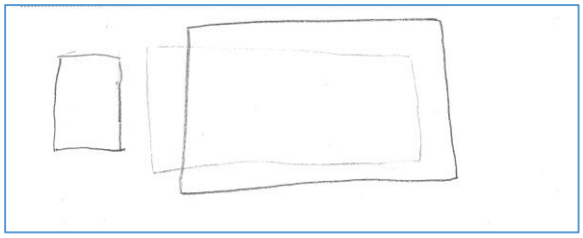
Pour tracer un carré de côté de longueur 100, Cléa propose le programme ci-contre .

Peux-tu sans utiliser Scratch prévoir ce qui sera tracé ?
Vérifie à l'aide du logiciel.
Corrige le programme pour qu'il permette de tracer un carré.

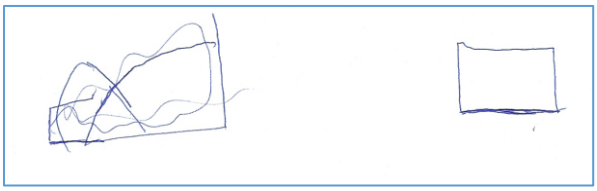
Il fait un rectangle.



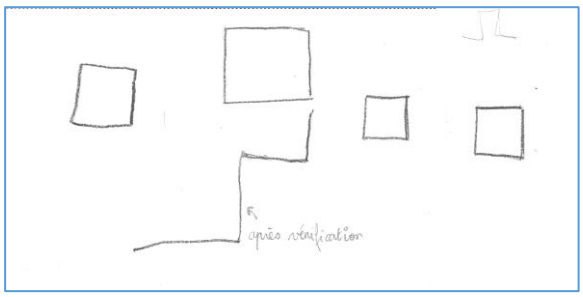
Elève A



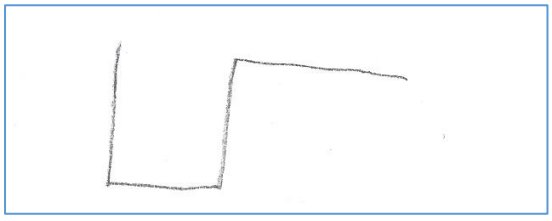
Elève C



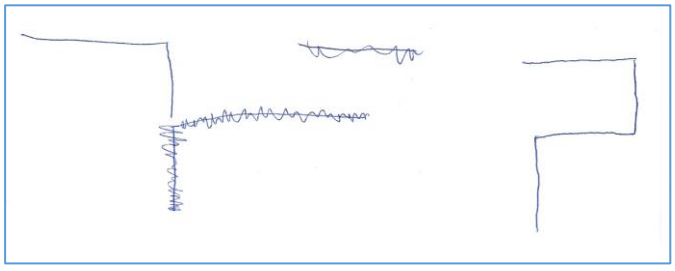
Elève D



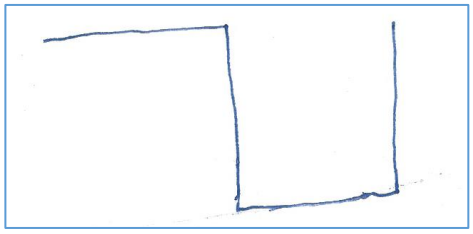
Elève E



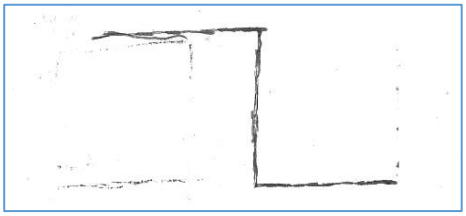
Elève F



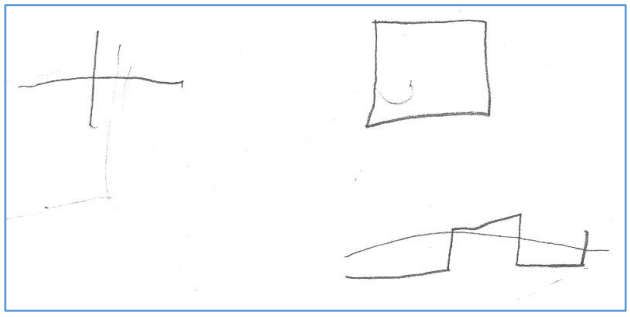
Elève G



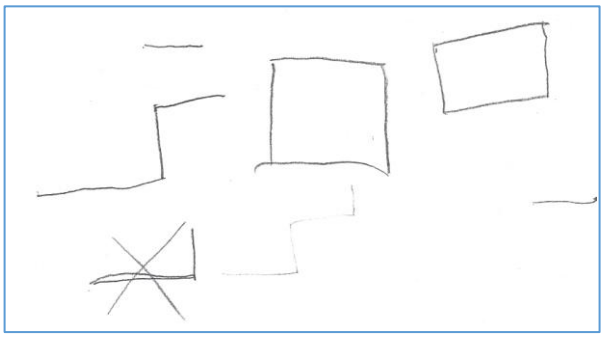
Elève H



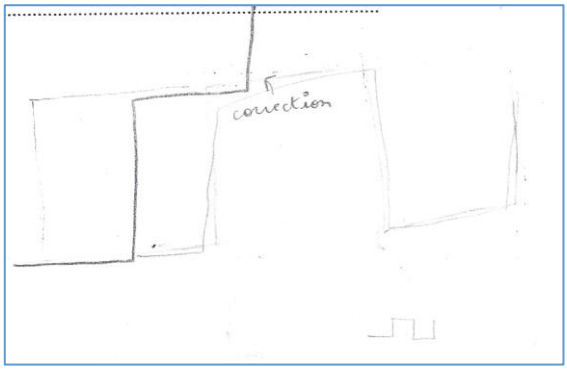
Elève L



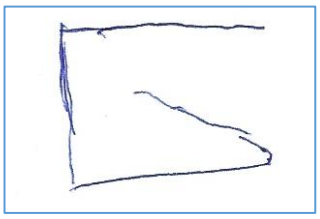
Elève I



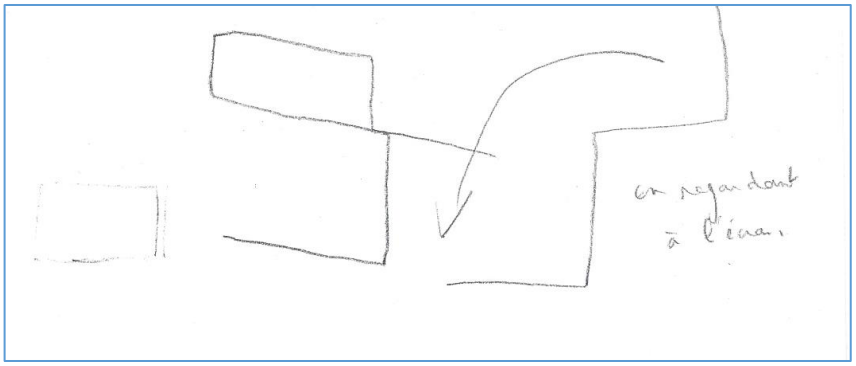
Elève N



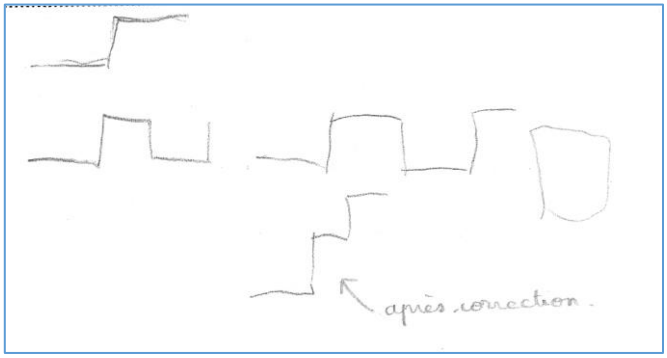
Elève J



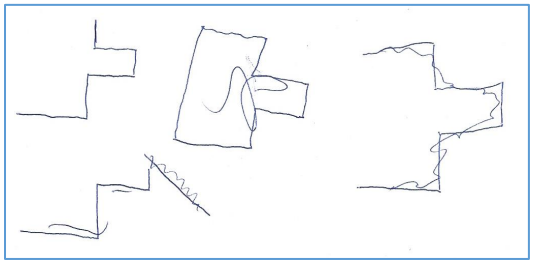
Elève K



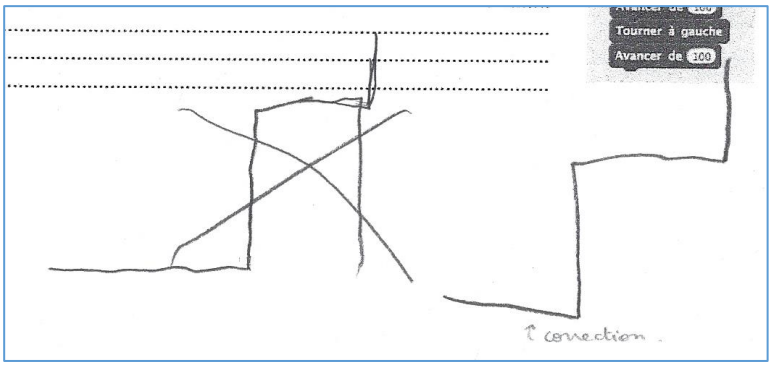
Elève M



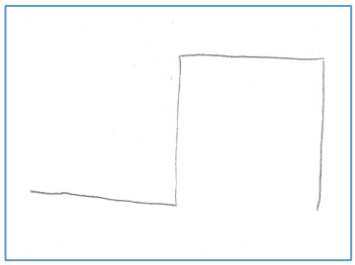
Elève O



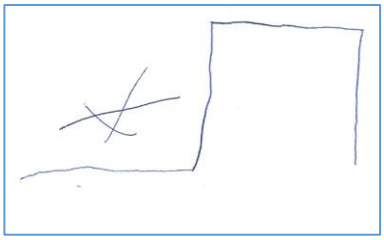
Elève P



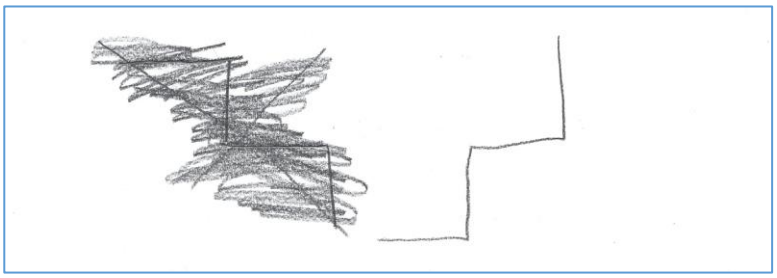
Elève Q



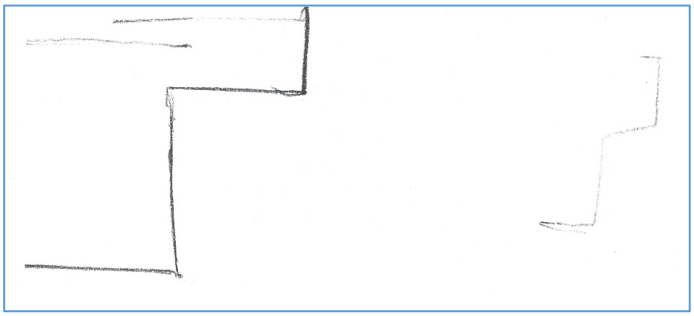
Elève R



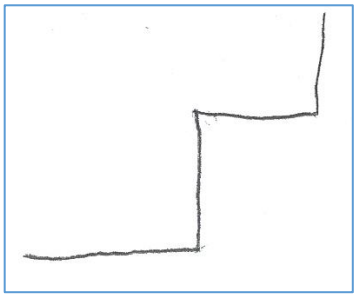
Elève S



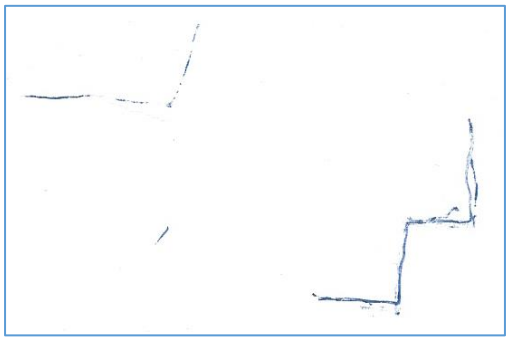
Elève T



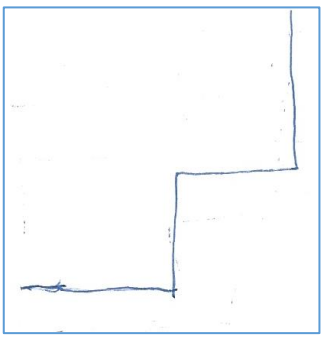
Elève U



Elève V

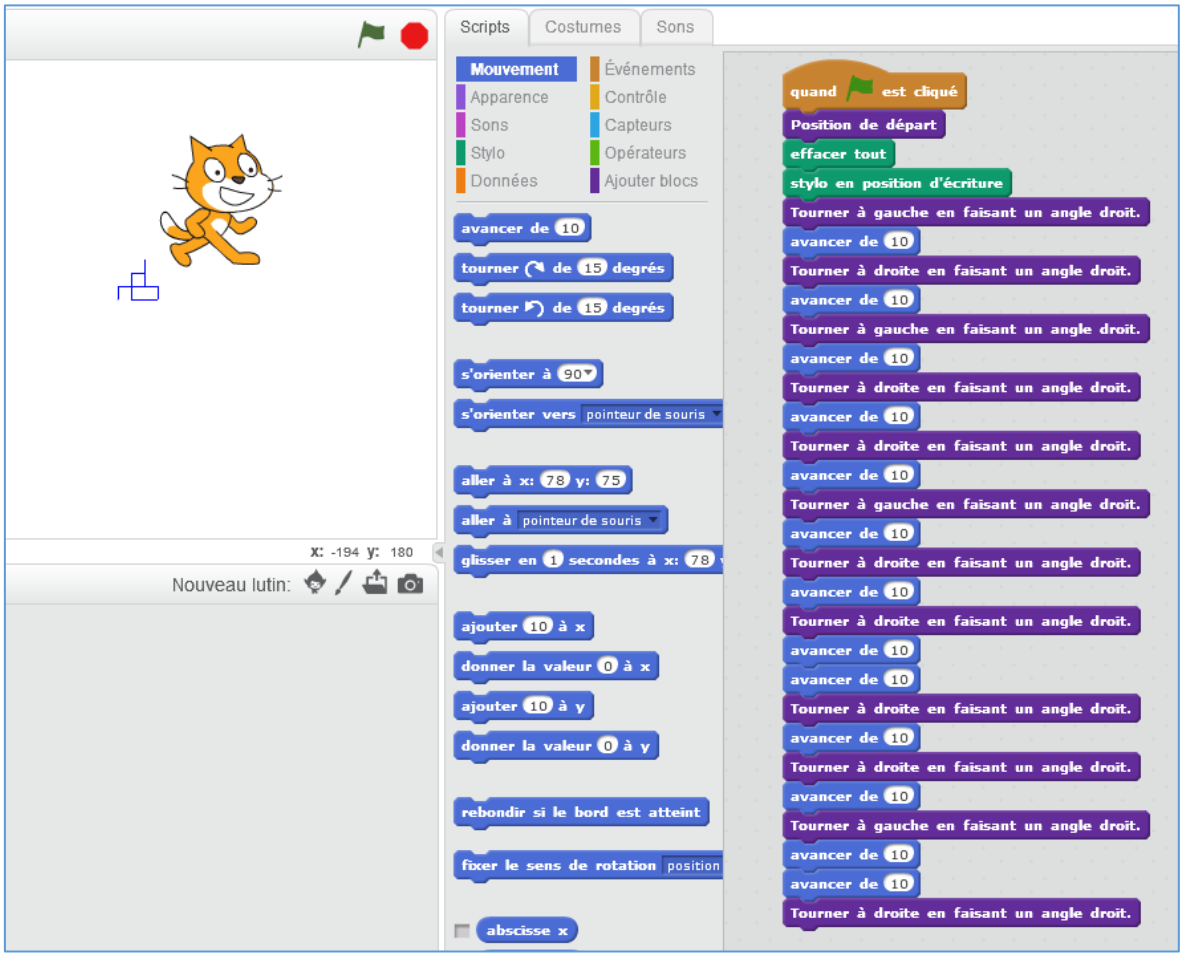


Elève W

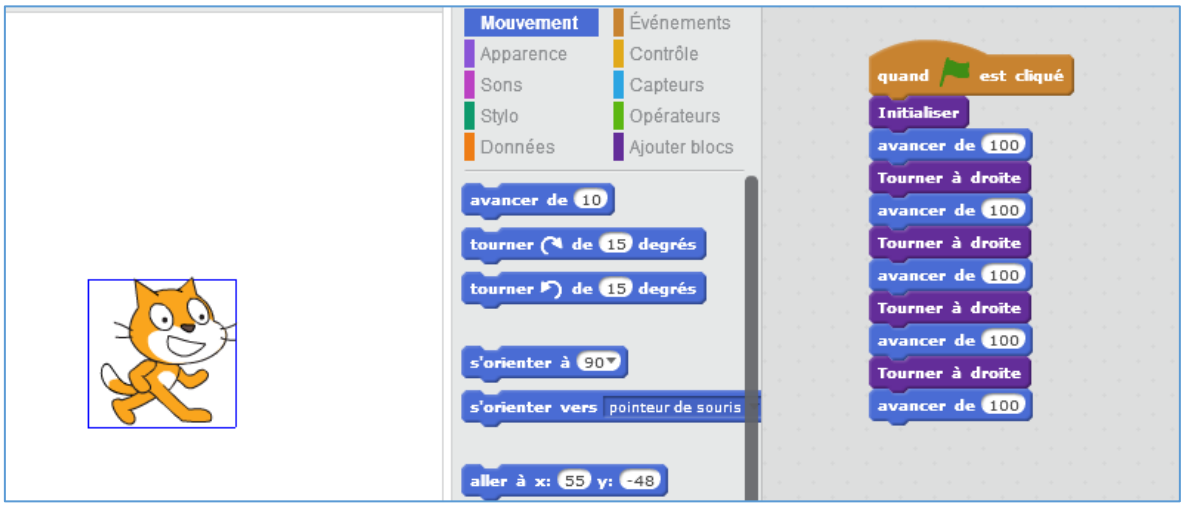


6.3. Annexe 3 – Productions finales de l'élève L (séances 1 et 2)

Séance 1 :

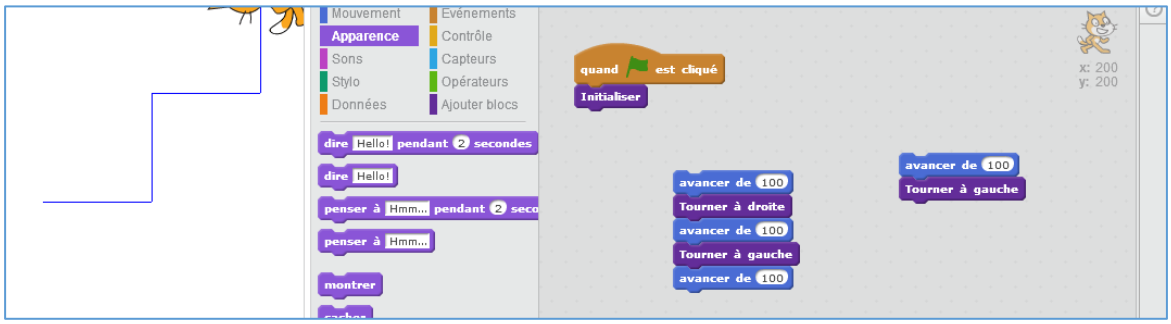


Séance 2

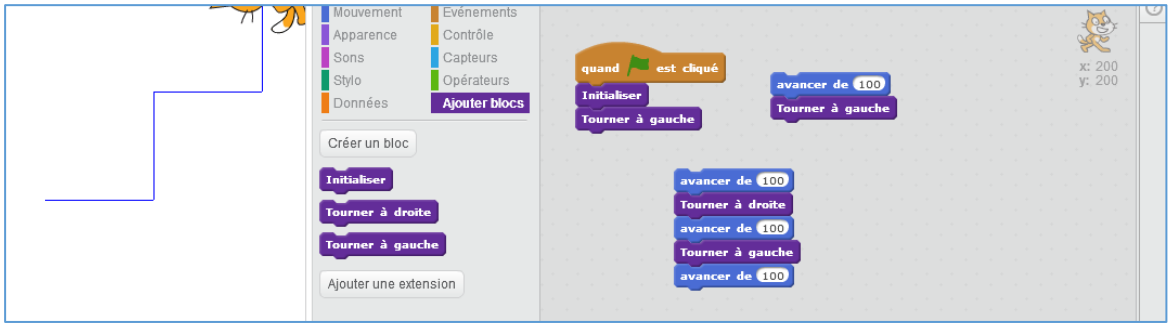


6.4. Annexe 4 – Séance 2 - Etapes de conception du programme corrigé du carré par l'élève P (4 captures d'écran)

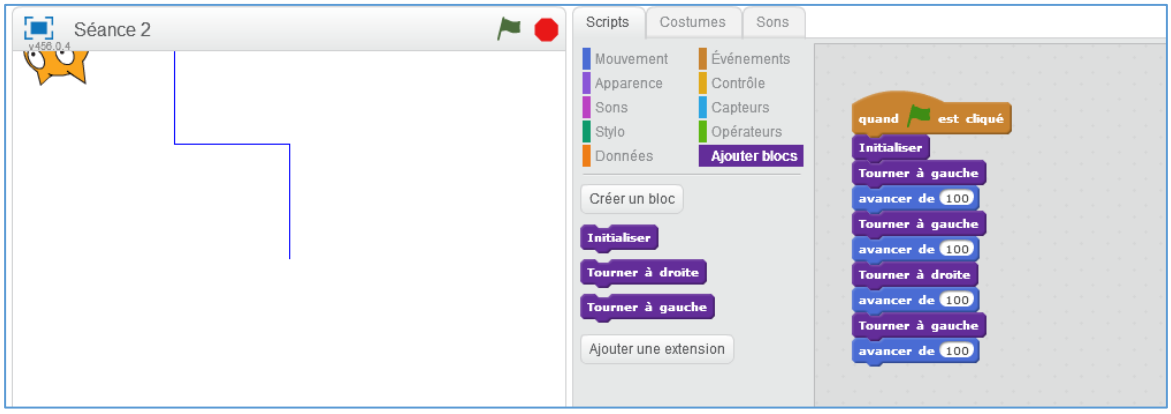
Capture d'écran n°1 :



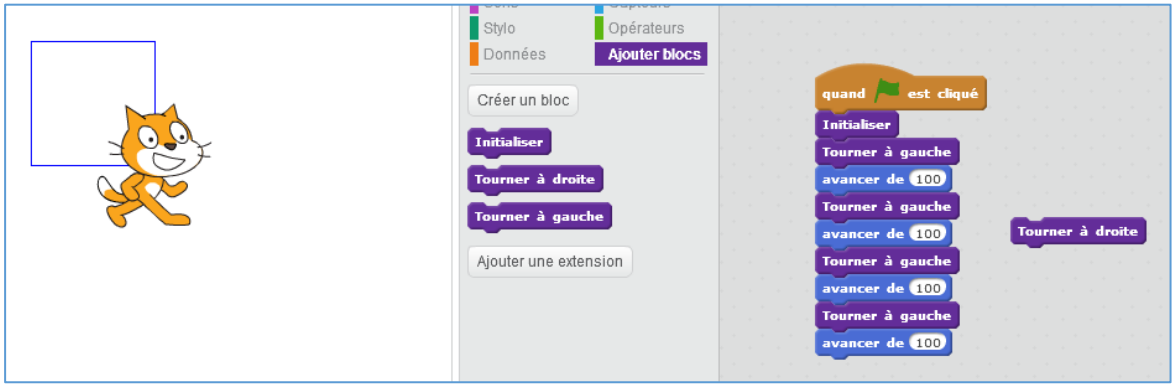
Capture d'écran n°2 :



Capture d'écran n°3 :

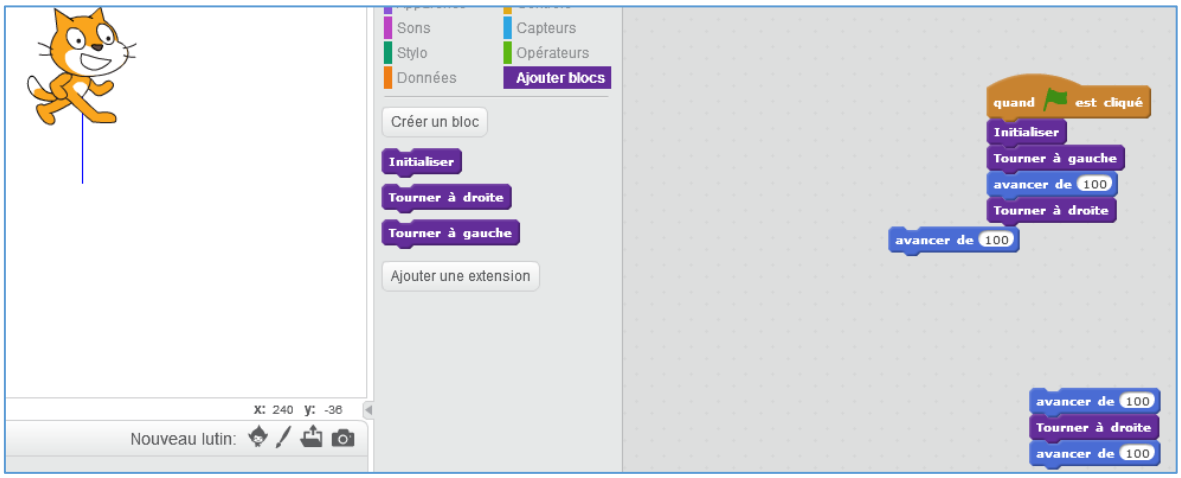


Capture d'écran n°4 :

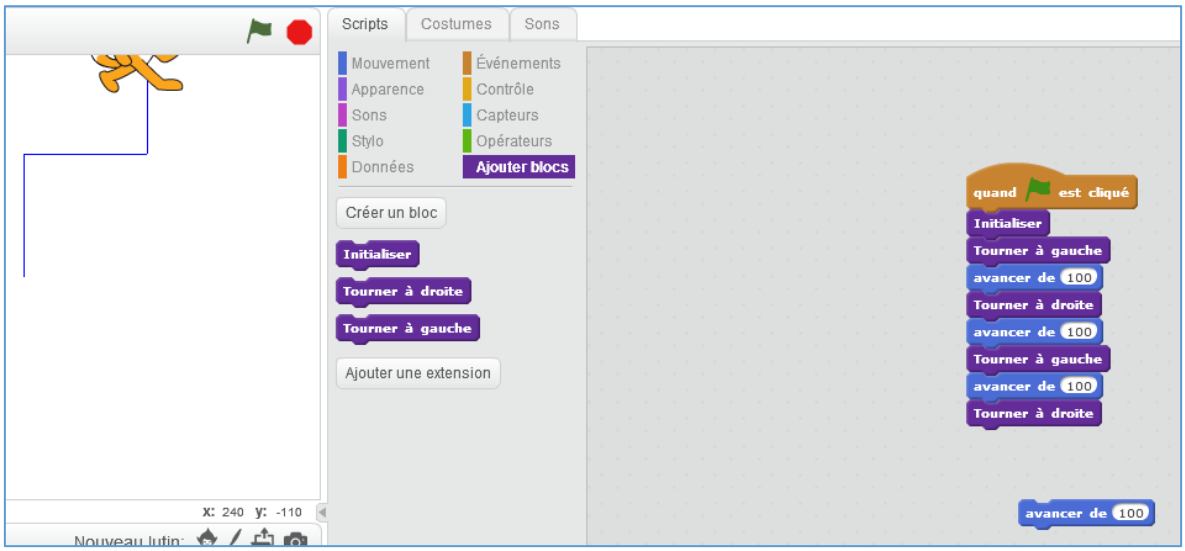


6.5. Annexe 5 – Séance 2 - Etapes de conception du programme corrigé du carré par l'élève H (8 captures d'écran)

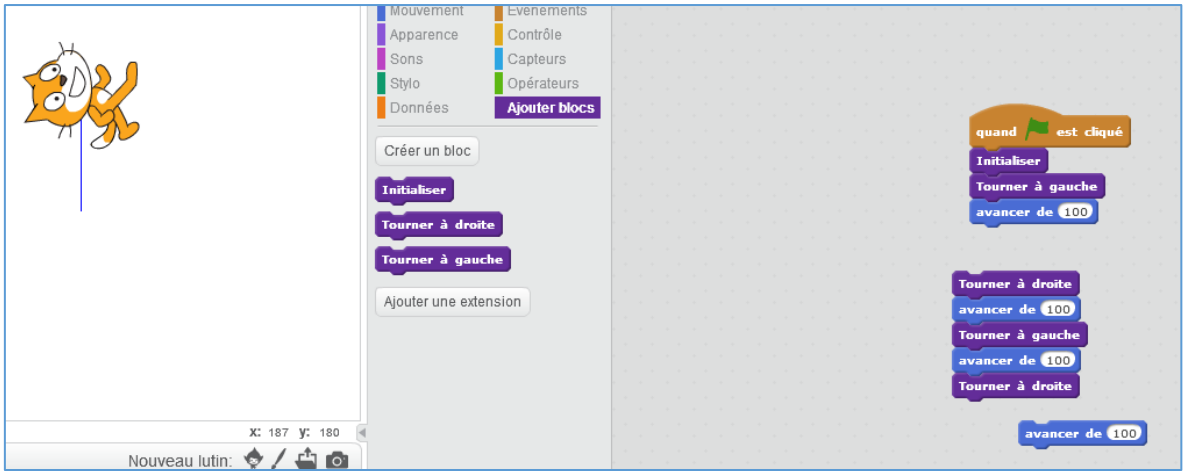
Capture d'écran n°1 :



Capture d'écran n°2 :

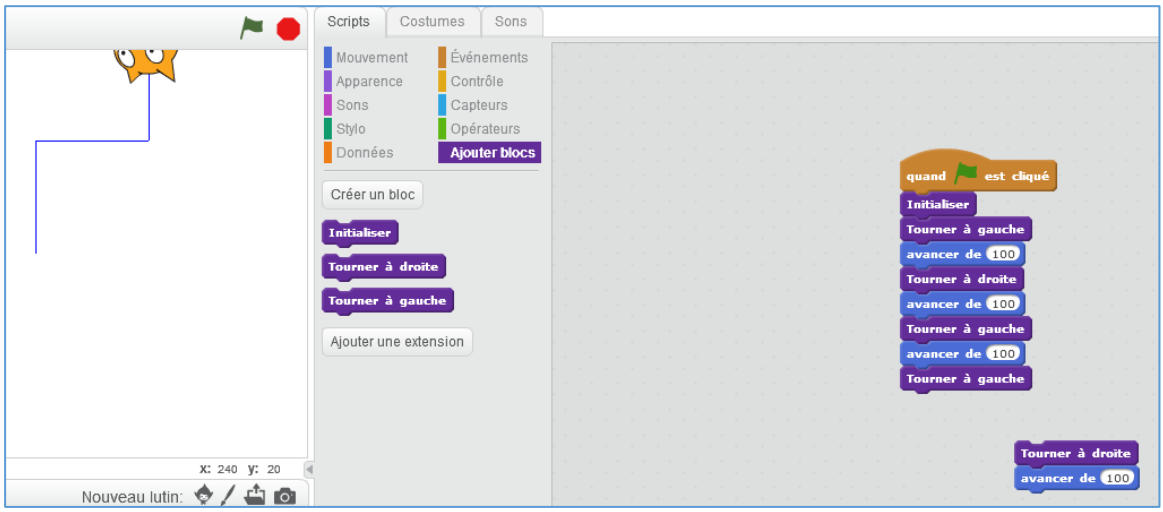


Capture d'écran n°3 : (L'élève recommence)

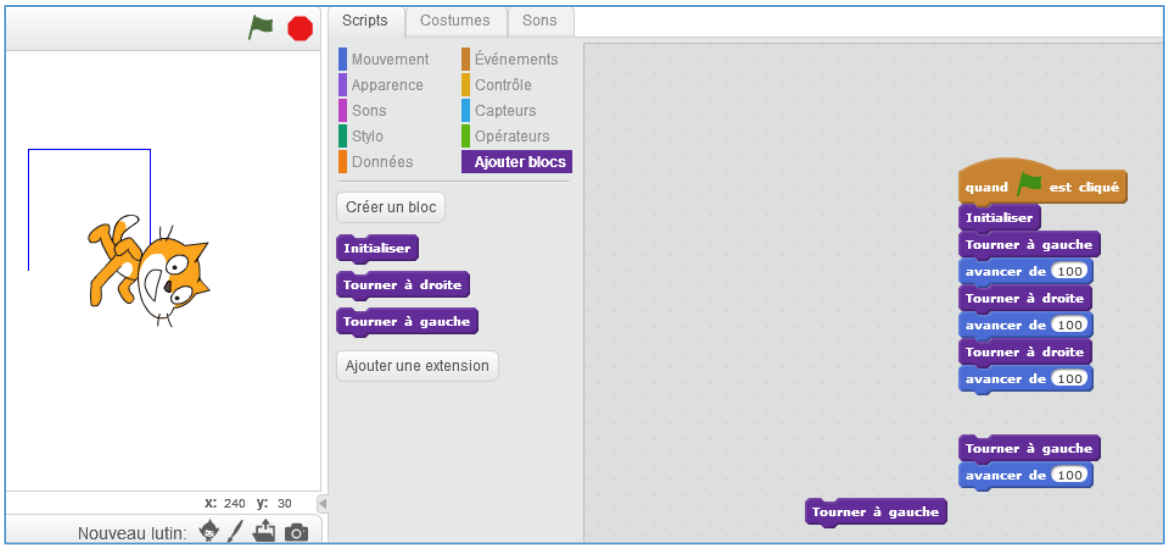


Capture d’écran n°4 :

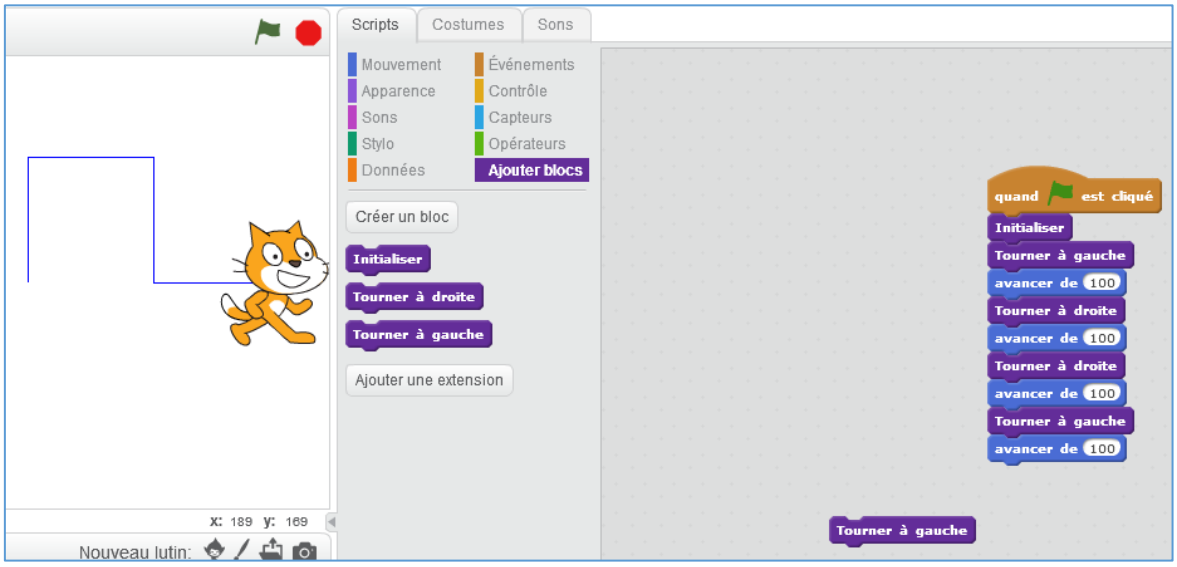
Capture d’écran n°5 :



Capture d’écran n°6 :



Capture d’écran n°7 :



Capture d’écran n°8 :

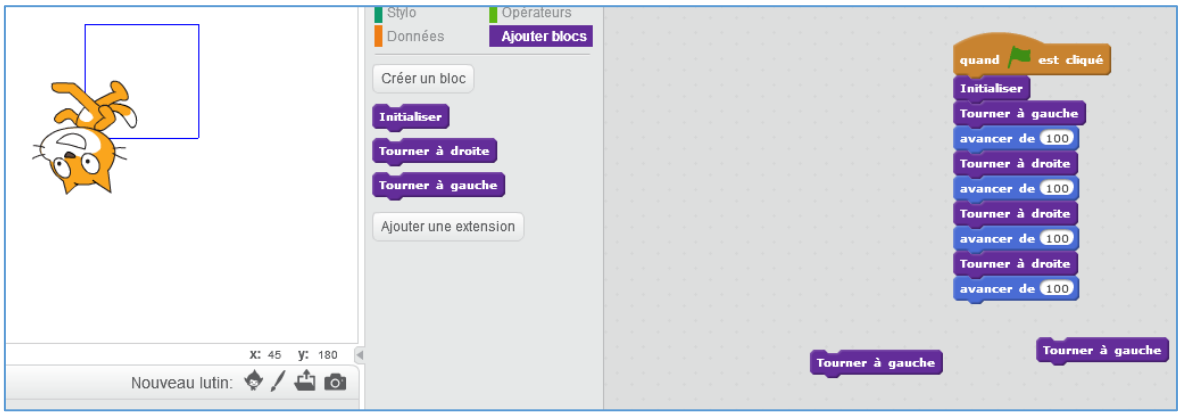


Table des matières